CHAPTER 3
RELATED WORK

3.1 INTRODUCTION

There is a substantial amount of literature concerned with the application of computers to law. In this chapter some important projects, generally referred to as legal consultation systems, which primarily deal with legal reasoning based on Artificial Intelligence techniques are discussed. A review of two of the most acknowledged Rule-Based Knowledge-Based systems, MYCIN and R1, is also presented. The discussion on MYCIN and R1 is limited only to the use of contexts which was employed to alleviate the problems associated with the unstructured rule bases.

3.2 CONSULTATION SYSTEMS

3.2.1 The TAXADVISOR

The TAXADVISOR system [43] was constructed by Robert Michaelson in his doctoral research at the University of Illinois. It has been designed to assist lawyers with tax planning for clients with large estates (valued at more than $175,000). The implementation has been done by using EMYCIN. The TAXADVISOR collects data about clients and makes planning recommendations and suggestions for organizing their estates. Entering into life insurance contracts and retirement schemes, making wealth transfers, gifts, purchases and will are all things which can improve the client's tax position. TAXADVISOR is concerned particularly with the strategic
planning aspects of this problem, rather than with the analysis of facts to determine tax results. It has been assumed that the tax effects on any recommendations would be calculated by a more conventional program.

The knowledge represented in TAXADVISOR has been chiefly drawn from practicing lawyer's experience and the different strategies they use, especially the criteria experts use to guide them in formulating financial plans. The author notes that for cases which fit the special area treated, validation by several practicing accountants indicated that about seventy percent of TAXADVISOR's recommendations were identical to those the experts would have made, while the remaining thirty percent were different but were still acceptable planning alternatives. It should be noted that in TAXADVISOR the effort has been to model how to use the legal knowledge rather than the law itself.

3.2.2 The TAXMAN project

The TAXMAN Project Thorne McCarty's TAXMAN [12] project is the most widely known and sustained project that deals with the application of artificial intelligence techniques to law. The work on this began as early as in 1972. The work done during the first few years is referred to as TAXMAN I and the work that has been done afterwards is referred as TAXMAN II.

McCarty has described the work on TAXMAN as 'an experiment in the application of artificial intelligence techniques to the study of legal reasoning and legal argumentation'. Thus the primary aim of the TAXMAN project was theoretical. The US corporate tax law has been taken as the domain of experimentation. The TAXMAN I system was constructed to represent the statutory rules and concepts which define corporate reorganizations according to the US Internal Revenue Code. TAXMAN I was initially an entirely
rule-based model and was written using MICRO-PLANNER [44] which MacCarty modified so as to attach time variables to assertions and thereby to represent the state of the world at various times within a single database. Subsequently it was reimplemented in AIMDS [45], a frame-based language.

Representation in TAXMAN begins by constructing a model of the domain. For corporate tax law, the domain will contain objects like corporations, individuals, stocks, shares, securities, transactions, etc. Each class of object is described by a template: a particular object being represented by a template instance, which can be viewed as a collection of assertions describing the object's properties (name, address, size, etc.). The domain is further described by a system of relations which express the possible relationships between objects. The facts of a particular case are described by a set of template instances and the relations between them. Concepts (like 'tax free reorganization') which are potentially applicable to a set of facts are also represented by templates and relations. The statutory rules which define these concepts are represented by logical templates.

The two reasons because of which the topic of 'corporate reorganizations' was selected are (1) the prospect of being able to entirely cover the domain with a small set of primitives and (2) the interestingness of what the program could not do. The investigation considers three specific cases, all decided by the Supreme Court during 1930's, in which the literal statutory requirements for a tax reorganization were met, but the income in question was held to be taxable nevertheless. In each case the Court introduced a new requirement for non-taxability and found that this additional requirement was not met. The concepts like "continuity of interest", "business purpose", "step transactions", etc., that were used by the court have been referred to by MacCarty as amorphous concepts, something
similar to the open textured concepts of Hart. The TAXMAN I project was incapable to handle such amorphous concepts successfully.

The main theme of McCarty's work on TAXMAN II was to handle the open textured or amorphous concepts which the TAXMAN I was unable to handle. The nature of the legal knowledge in the domain treated by TAXMAN has made it possible to investigate the representation of the deontic and Hohfeldian concepts and reasoning with the concepts of permission and obligation particularly. In the course of these investigations McCarty has proposed a system of deontic logic [46].

In TAXMAN II the amorphous concepts have been represented using conceptual structures called prototype-and-deformation [47]. The idea behind this representation scheme is that a legal concept whose meaning has evolved with the accumulation of case law can be represented in terms of a structure with three components:

1. There is an invariant core which expresses necessary but not sufficient conditions for the concept to hold. This component can be optional.

2. There is a set of exemplars each of which matches some but not all instances of the concept. These would presumably include cases in which the concept has been decided.

3. There is a set of transformations which express the relationship between the exemplars by stating how one exemplar can be mapped into another.

Deciding whether an open textured concept or amorphous concept does or does not apply to the facts of a new case
needs argumentation. In TAXMAN II, to support a claim that the concept holds for a new case, it is necessary to demonstrate how a coherent representation of the concept can be constructed with in the prototype-and-deformation framework in such a way that a new case is included. A coherent representation is one in which a sequence of permitted transformations maps the new case into one for which the concept is decided. This process generates hypothetical cases where there is agreement whether the concept holds or not. However, the legal argument being adversarial in nature, one (possibly the opponent) can construct another representation of the concept in which the new case is decided differently.

3.2.3 Meldman's Project

Jeffry Meldman's doctoral work carried out at M.I.T [48] is one of the earliest attempts to employ the artificial intelligence techniques for the analysis of cases or certain fact situations in the legal realm. Meldman's domain of investigation was 'assault and battery' and his main interest was to produce a plausible design of a program that draws its domain knowledge from more than one source of legal knowledge. Though Meldman's system was designed to be executed by a system called OWL [49] it was never implemented. However, some time later a modified version of this design was implemented by J.J. King [50].

The analysis of a new fact situation or case is based on a fictitious body of legal sources referred to as Corpus Juris Mechanism, extracted from a standard textbook. This body of domain knowledge sources is made up of a collection of legal rules and a set of cases and their decisions. The question here is how a case can be authority for anything beyond its own particular facts. This question has been resolved in a very simple way by representing the case itself, along with its decision, as a general rule. A general rule
which expresses the decision in a case corresponds to what often referred to as the ratio decidendi of the case. As the decisions of previous cases may not have the explicit reasons, depending upon which those decisions were taken, recorded or available the ratio decidendi of previous cases have to be reconstructed starting from the scrap, in general. In Meldman's system the rule expressing the decision of a case is constructed by him and hence this system can be regarded as an example of a system based on a single, fixed interpretation of the legal sources.

The input to the program is a hypothetical set of facts. During the analysis the system attempts as its top goal, to establish the presence of one or more intentional torts. Subgoals are established in several ways: by type, by element, by example, and when all fails by querying the user. The system determines the extent to which the input facts 'fall within legal doctrines' by syllogism (i.e., by applying its rules to the facts by deduction) or 'near to the legal doctrines' by analogy.

Meldman himself has remarked that the version of analogy used is far too simple. The analogy mechanism used by him is based on a hierarchical classification of the concepts represented in the system. This classification of the concepts is made use of in the contexts when the facts of a case do not exactly match the conditions of a rule in the system in order to extend the notion of matching. A fact in a given case is considered to match a condition in a rule when both of them are instances of a more general concept in the hierarchy. As pointed by Gardner on page 75 in [14] this method of analogical reasoning bears noticeable resemblance to Austin's discussion of analogy, with which Austin himself was never satisfied.
3.2.4 Gardner's Program

The main aim of Anne Gardner's doctoral project [14] was to address directly the treatment of open texture in law. The research uses materials classically taught by the case method in law schools and the specific legal topic is the formulation of contracts by offer and acceptance. The overall objective of the work was to develop a program that recognizes the issues that lay abstrused in a case and to distinguish between those cases which have enough information to resolve (referred to as easy cases) and those on which competent human judgments might differ (referred to as hard cases). Gardner's system has been implemented in MRS [51] and provides definitions for both technical legal concepts as well as non-technical concepts.

The feature that mainly distinguishes Gardner's work from the previous approaches is her recognition that most legal questions do not have a single 'right' answer. Gardner's System (having no name) has been primarily designed to provide alternative analysis whenever it comes across a hard case. The input to the program is hand generated internal representation (called target representation) of the given problem. The output of the program is a tree of all the solutions which can be reached for a given case. Branches in this tree are introduced by the presence of open textured concepts which can not be decided definitely one way or the other. The tree of possible solutions corresponds to a set of conditional answers, where each answer is qualified by the 'hard' conditions which remain to be determined. The system has a set of rules marked competing rules that aid in resolving 'hard' cases. When the system encounters a condition which has several rival definitions, every such rule will be tried. If the choice of a rule makes a difference to the outcome then there is a hard legal question and the condition becomes one of the qualifications on the final
conclusion reached. If all the competing rules agree, then the condition is taken as decided and does not become a binding or qualification on the final conclusion. Also there exist a set of rules referred to as complementary rules which provide alternate paths for moving towards the same conclusion.

In Gardner's system the most general knowledge structure is a version of an augmented transition network. The nodes of the network correspond to legally distinct types of states of the world having labels like "no relevant legal relations exist", "a contract exists" and "an offer is pending". The arcs of the network correspond to events that can change the legal relations between the parties involved in a contract and have labels like "offer", "counter offer", "rejection" and "acceptance". All states of the network are in effect final states i.e., a sequence of events may or may not lead to the formation of a contract. As Gardner points out on page 16 in [17] the control structure of the program is simply a set of nested loops with the main loop attempting to find the best legal characterization of the next event considering the events that are yet to be examined, the conclusions reached about any events already processed and the current configuration of the transition network. Because of the presence of a hard question it may not be possible to settle on one best characterization.

3.3 KNOWLEDGE-BASED SYSTEMS APPROACH

3.3.1 The MYCIN system

MYCIN [3] [52] is a rule based system. Characteristics of rule based systems are described in chapter 4. MYCIN'S performance task is diagnosis of blood infections and the recommendation of drug treatment. MYCIN conducts a consultation (in English) with a physician-user about a patient case, constructing lines-of-reasoning leading to the
diagnosis and treatment plan. The line of reasoning is accomplished by backward chaining of the rules. Its overall design is shown in figure 3.1.

![Diagram of MYCIN system](image)

**Figure 3.1 Design of the MYCIN system.**

MYCIN's knowledge is stored as fact-triples with associate degrees of certainty and as conditional rules relating facts, as shown in figure 3.2.
The rules describe the various relationship between the parameters. Depending on the values of the parameters, the rules make appropriate conclusions and suggest the drugs for the illness. There are about five hundred rules, about half of which are for blood infections and the remaining for meningitis infections. Rules are subject to categorization in accordance to the types of objects which they are most appropriately invoked. A type of object is called a context in MYCIN. For example, some rules deal with organisms, some with cultures and some with patients themselves. There are twelve types of contexts in total. Each rule in MYCIN belongs to one context type. Figure 3.3 gives an example of a diagnostic rule.
RULE 0072

IF 1. The stain of the organism is GRAMPOS and
2. The morphology of the organism is COCCUS and
3. The growth confirmation of the organism is CHAINS

THEN There is suggestive evidence (0.7) that the identity of the organism is STREPTOCOCCUS

Figure 3.3 A MYCIN rule.

In MYCIN, there is a way for the expert to state to what extent, on a scale from 0.0 to 1.0, he or she believes that preconditions imply the conclusion. 1.0 is certainty; 0.9 is "very strong evidence"; 0.6 is "suggestive evidence" and so on. These indices are combined in MYCIN with a very simple and easily explained function, yielding an index called a "cumulative certainty factor", an index of "strength of belief" in the line of reasoning.

RULE 0072 shown in figure 3.3 belongs to the context CURORGRULES. The rules in this class are only used in inference about current organism. STREPTOCOCCUS is a parameter in context CURORG (current organism). With every parameter declarations about the relevant rules will be stored. These declarations are stored in three lists and they are:

LOOKAHEAD: A list of rules where its name occurs on their left hand side, i.e., their premises.

UPDATED-BY: A list of rules where its name occurs on the right hand side, i.e., their conclusions.
CONTAINED-IN: A list of rules where its name occurs on the right hand side but which don't change its value.

An example of parameter and its declarations about the rules is given in figure 3.4.

**Figure 3.4 An example of a single value parameter.**

One can see that the representation structure in MYCIN is partitioned with respect to contexts. This helps to contain the search space to the rule which are only relevant to the current context or the current object (e.g., organism, culture or patient) of interest. In addition, each parameter has all the relevant rules recorded categorically into three types. These three types are labeled as CONTAINED-IN, LOOKAHEAD and UPDATED-BY. This provides a further improvement in the search process.
3.3.2 The R1 System

R1 [6] is a system for configuring Digital Equipment Corporation's VAX-11/780 systems. R1 is a rule based system and at present it configures almost all the VAX orders. Where as in most of the systems tasks to be performed are analytic in nature, the configuring task of R1 is synthetic i.e., given a set of components, R1's task is to impose relationships on those components and in so doing to form a complex object. R1 has 772 rules that enable it to perform the configuration task. Out of these only 480 rules contain
knowledge that is directly related to the configuration task and the remaining 292 rules contain more general knowledge. The data base contains descriptions of 420 components all of which have been grouped into fifteen different classes like Bundle, Cabinet, Unibus module, etc. On an average, each rule refers about 6.1 components or objects in the data base (the number of referencing can go up to as high as seventeen objects). On the data base, each component description consists, on the average, of eight attribute/value pairs. However, a component may have as many as fifty distinct attributes, several of which are common to most of the component classes. Figure 3.5 shows two of the entries on the data base. RK711-EA represents a bundle of components where as RK611* represents a unibus module.

R1 has been implemented in OPS4. OPS4 is a forward chaining production language. The OPS4 interpreter operates with in a control frame work called the "recognize-act" cycle. During the recognize phase, it collects all the instantiated rules. During the act phase, it performs the actions in the satisfied rule. Whenever more than one rule is instantiated in a cycle, 'conflict resolution' is used to execute the most appropriate rule. An OPS family conflict solution strategy [53] is given in figure 3.6.

However R1 heavily relies on the third strategy, mentioned in figure 3.6, which is known as 'the special case strategy'. The recognize-act cycle is repeated until either no production rule can be instantiated further or an action explicitly stops the process.

At the top level, a configuration task in R1 has been viewed as a sequence of six major sub tasks that have strong temporal dependencies, Figure 3.7 gives a sequence of these sub tasks in a chronological order.
Discard from the conflict set the instantiation that have already fired. If there are no instantiations that have not fired, conflict resolution fails and no instantiation is selected.

Order the instantiation on the basis of the currency of the working memory element, using the following algorithm to compare pairs of instantiation: First, compare the most recent elements from the two instantiations. If one element is more recent than the other, the instantiation containing that element dominates. If the two elements are equally recent, compare the second most recent element from the instantiations. Continue in this manner either until one element of an instantiation is found to be more recent than the corresponding element in the other instantiation or until no elements remain for one instantiation. If one instantiation is exhausted, before the other, the instantiation not exhausted, dominates; if the two instantiations are exhausted at the same time, neither dominates.

If no one instantiation dominates all the other under the previous rules, compare the dominant instantiation on the basis of the specificity of the LHSs of the productions. Count the number of tests (for constants and variables) that have to be made in finding an instantiation for the LHS. The LHSs that require more tests dominate if no single instantiation dominates after the previous rule, make an arbitrary selection the dominant instantiation.

Figure 3.6 LEX Conflict resolution strategy.
Determine whether there is anything grossly wrong with the order (e.g., mismatched items, major prerequisites missing).

Put the appropriate components in the CPU and CPU expansion cabinets.

Put boxes in the unibus expansion cabinets and put the appropriate components in those boxes.

Put panels in the unibus expansion cabinets.

Lay out the system on the floor.

Do the cabling.

Figure 3.7 R1's sub-tasks.

DISTRIBUTE-MB-DEVICES-3

IF : the most current active element is distributing massbus devices AND

there is a single port disk drive that has not been assigned to a mass bus AND

there are no unassigned dual port disk drives AND

the number of devices that each massbus should support is known AND

there is a massbus that has been satisfied at least one disk drive AND

that support additional disk drives AND

the type of cable needed to connect the disk drive to the previous device on the mass bus is known

THEN: Assign the disk drive to the massbus.

Figure 3.8 A R1 rule.
In accordance with the major sub tasks, mentioned in figure 3.7 on page 50, rules in R1 have been classified into different "contexts". Figure 3.8 gives an example of the English translation of a R1's sample rule.

In the working memory, among descriptions of various components, there is a "context element" the value of which indicates the current active context. In other words, the value indicates the current active sub task. The first condition in rule DISTRIBUTE-MB-DEVICES is to test if this rule is applicable to the current context, such as 'distributing mass bug device'. By so doing, R1 manages to exclude the rules which are not applicable to the current sub task and reduce the amount of search. In addition to improving the search efficiency, such an arrangement also reduces the trouble of having too many conflicting rules. This is the case because now the rule base is effectively partitioned.

3.4 SUMMARY

One of the important requirements of the legal realm (discussed in chapter 4) is that any argument put forth before a court of law as well as any judgment pronounced by a court of law need to quote the source references or 'citations' of either the statutory provisions or the precedents. None of the projects reviewed in this chapter, including the most sustained project TAXMAN and the work of Anne Gardner that addresses the problem of 'opentexture' associated with the legal concepts, have dealt this requirement. The work reported here addresses this issue and proposes a modified version of the production rules which facilitate in extending the much required citations.
With rule based systems having no facility to handle 'contexts', during every recognize-act phase of the problem solving process each and every rule on the rule base will be tried, and whenever more than one such rule is triggered a conflict resolution strategy is employed to fire an appropriate rule. Regardless of the domain there exist implicit groups of rules that apply in specific situations or contexts. The MYCIN and RI systems have used this 'context' information by encoding them into the rules themselves, and hence avoid the unnecessary 'trying' of irrelevant rules. However, the rule base still remains unstructured and flat. In the legal realm rules can be explicitly grouped into contexts as shown in figure 3.9. In figure 3.9 row headings correspond to chapters of the ACT and column headings correspond to the sections of the ACT. In this work every context (corresponding to a square in figure 3.9) has been treated as an 'object' and has been represented by a frame like structure called FORM. Use of FORMS has not only
allowed the proper organization of the knowledge base and improved the efficiency but has extended the much required facility for shifting attention.