4.1 INTRODUCTION

Representation of knowledge concerns the formulation of symbol structures which can be used by a computer to represent the external task domain. The study of representation involves choosing the right kind of symbol structures for a certain application. The work of Ernst and Banarjee on relationship between strong and weak problem solver [54] gives some interesting views on the philosophy of representation. It presents some arguments for believing that the power of a problem solver is tied in with knowledge representation and its control structure. For example, Fortran is particularly useful for engineering computation because its symbol structures allow mathematical formulae to be expressed easily. Similar languages like Lisp, Algol, Pascal have their individual application areas because of their distinct symbol structures.

The popularity of Fortran in the engineering domain is partly due to the fact that in this field there are many problems which can be modeled by mathematics. Mathematics itself a powerful representation language and it is well understood. Hence, the designers of Fortran had a clear idea of the knowledge they wished to represent. The users of Fortran know the tasks and have the ability to express this knowledge in mathematical terms. In artificial intelligence, the situation is not as easy. First of all, we do not have a vehicle (except English and logic) to express the knowledge.
Secondly, the domain experts sometimes have difficulties articulating their own reasoning process and providing an explanation for their actions.

Various knowledge representation structures exist. Each type was designed by a group of workers who were interested in their particular applications. A good representation enables one to express the problem directly and to use the expressed knowledge in a way that one is accustomed to in practice. As Bobrow has pointed out in [55] "one can certainly express Maxwell's equations in English but the expression is of little use for solving the problem". There is also another attitude which tends to believe that one type of representation is suitable for all the problems. Though no one has yet proved such an attitude as wrong, it seems that such a viewpoint is naive since all the researches reported so far do not support it. There are different representation methods. Each one has been conceived and implemented by different groups of workers who have their own special interest. Consequently, the power of a method is biased towards that particular application.

This chapter discusses the representational requirements of the legal knowledge and presents a knowledge representation technique that uses modified production rules in unison with frame like constructs called FORMS [56]. The discussion also provides an introduction to the general form of rule-based systems. Along the way the need for modifying the production rules in order to fulfill one of the vital requirements of the legal domain i.e., to provide knowledge sources reference, and to eliminate limitations associated with knowledge-based systems when only 'production rules' type of knowledge representation technique is employed have been discussed. Towards the end of the chapter is presented a discussion regarding how the use of "FORMs and modified production rules" in unison facilitate in properly organizing
large number of domain specific rules as well as in having the required focused reasoning during the problem solving stage of the system.

4.2 REPRESENTATIONAL REQUIREMENTS OF THE LEGAL DOMAIN

4.2.1 The Shifting Legal Knowledge

The industrial Disputes Act as enacted in 1947, was a piece of legislation mainly concerned with providing machinery for investigation and settlement of industrial disputes. The Act, as it stood up to October, 1953, had no provision for the payment of 'lay-off' or 'retrenchment' compensation to the workmen who were 'laid off' or retrenched in certain contingencies. Though some progressive employers used to voluntarily pay the relevant compensations and the Industrial Tribunals used to award such compensation, the situation was far from satisfactory. In the absence of any norms laid down by law, adjudicators had to take various factors into consideration in awarding compensation and determining its quantum. The resulting adjudication was, therefore, neither certain nor uniform. The situation was precipitated in 1953, when as a result of accumulated stocks in textile industry, textile mills were threatened with the consequences of closure of one or more shifts entailing lay-off or retrenchment of large number of workers employed in the industry. In order, therefore, to avoid industrial unrest in the country, the President of India promulgated the Industrial Disputes (Amendment) Ordinance, 1953 (Ordinance 5 of 1953) with effect from 24th October 1953, making provision for Compensation for lay-off retrenchment, setting a common standard for all employers [57]. Besides making other relevant and necessary amendments in the Act, Section 3 of the amending Act engrafted Chapter V-A in it. Even these amendments made by the amending act, did not make provision for liability of an employer to pay compensation to workmen for termination of their service.
on transfer of an industrial undertaking or closure thereof. This lacuna was noticed by the Supreme Court of India, in [58]. Consequently, the Industrial Disputes (Amendment) Ordinance (Ordinance 4 of 1957) was promulgated by substituting new Sections 25-FF and 25-FFF for the old Section 25-FF which make provision for notice or payment of wages in lieu of notice and compensation being given to workmen discharged from service on transfer or closing down of an industrial undertaking as if they have been retrenched.

Original Section 25-C enacted along with the chapter V-A in 1953 was once again amended by the Industrial Disputes (Amendment) Act, 1956 (Act 41 of 1956) in order to provide the laid off employees, who had completed not less than one year of continuous service, a lay-off compensation for a maximum period of forty-five days during the course of any twelve months. Also as per this ordinance, after the expiry of the first forty-five days, if the workers are laid-off for a continuous period of one week or more, the effected employees were to be paid compensation for all the days comprised in every such subsequent period of lay-off, unless there was an agreement to the contrary. This provision was open to abuse inasmuch as workmen could be denied lay-off compensation by being made to work for some days in each week after the first forty-five days of lay-off. With a view to prevent such abuse, it was considered necessary to make provision that lay-off compensation would become payable for all the days of lay-off beyond the first forty-five days regardless of the fact whether the period of lay-off was continuous for a week or not. With this end in view, the new Section 25-C was substituted for the old Section 25-C during 1965.

As pointed out by Houghteling in [20] as well as the preceding discussion reveals, it is apparent that the chief characteristic of the legal domain is that: the knowledge keeps changing or the legal knowledge is dynamic in nature.
The statutes and the laws get amended depending upon the situations that come after their enactment. For example, the Industrial Disputes Act of 1947 was amended in 1953 to ameliorate the situation that came up due to the accumulating stock in the textile industry. Amendments are also made to meet the requirements of the changing social, political and other similar setups. Such amendments not only add new rules or conditions but also drop old rules once they become obsolete or irrelevant. For example, the 1953 amendment to the Industrial Disputes Act did not have any provision to pay the compensation for the affected workmen when the industrial establishment were either transferred or closed. At a latter stage this point was noticed by the Supreme court while dealing with an actual case. As a result old section 25-FF was replaced by new Sections 25-FF and 25-FFF. This amounts to rewriting of some rules in order to bring certain items or actions into the ambit of already existing body of rules and to put some outside its ambit. Amendments or changes in rules may also become necessary because of the abuse of certain existing provisions in some Acts. For example, as it was mandatory to pay the lay-off compensation for the laid-off employees whenever the lay-off period exceeded one continuous week after the expiry of first forty-five days, the employers used to lay off the workers only for one or two days in a week thus avoiding the payment of compensation.

It is obvious that any knowledge representation technique that is employed to represent the dynamic legal knowledge in the development of knowledge-based systems should primarily permit the addition, deletion and refinement of the legal rules without affecting the reasoning process i.e., the control structure, and hence facilitate the incremental and modular development of the systems. Here modularity not only pertains to the strength of coupling between the functional units of a program but also is concerned with the non existence of any sort of direct interaction between the
knowledge chunks.

Also, for in order to facilitate the development of a knowledge-based system modularly the knowledge representation technique employed should be able to hold the knowledge with a high degree of autonomy [59]. In other words, the selected representation technique should be capable of holding the domain as well as the profession's heuristic knowledge as independent nuggets or chunks.

4.2.2 Representing the legal knowledge

Of the many knowledge representation schemes that exist "production rules" are the one which has been most widely used. Apart from allowing the domain knowledge to be represented as independent pieces or chunks in the form of rules, the production rules technique of knowledge representation offer a representation of knowledge that is relatively easily accessed and modified, making it quite useful for systems designed for incremental approaches to competence ([60], [61]). As pointed out by Newell, in [62], production systems are not just considered as a convenient rule of thumb for approaching psychological models but are also viewed by many as a powerful technique, the power of which arises out of its close resemblance to fundamental mechanisms of human cognition. Though these production rules are encoded employing some constrained syntax, their uniform structure provide the advantage of allowing the rules to be understood by another part of the system easily in order to examine or modify the rules automatically [63].

The most fundamental and significant characteristic of the rule based systems or the production systems is that the rules interact only through a single channel (i.e., database). This uniqueness of the channel implies that the change in the fact situations (i.e., the modifications on the data
base) are accessible to every one of the rules. The fact that, in a production system, the next rule to be actuated is determined solely by the contents of the data base and no rule is ever called directly allows production system programs to be written modularly. Most acknowledged systems like MYCIN [3], R1 [6] etc., are some of the examples of rule based systems. As already discussed in chapter two, the heart of law and hence 'the legal system' is legal rules. Because of the many comparable features of the legal rules with the production rules as well as the fact that production rules extend the facility of developing the knowledge-based systems modularly, the legal rules have been viewed and dealt in a way similar to the production rules. As a result majority of the knowledge-based systems (i.e., expert systems; if one can call them so) in the legal domain that have been developed till date in order to assist legal practitioners in analyzing cases have employed rule-based knowledge representation scheme [64].

4.3 AN OVERVIEW OF RULE-BASED SYSTEMS

Production systems were first proposed in 1943 by Post [65] as a general computational mechanism. Since then many different applications have been carried out using production systems but with various modifications on the original idea. Each one had certain aspects of production systems enhanced to suit a particular kind of application.

4.3.1 A typical rule based system

There are three basic components in a production system: a set of rules in the rule base, a data base and an interpreter. The interpreter is sometimes referred to as an "executive" or an "inference engine".
A rule is, in its simplest form, a pair of symbol strings. In Lisp, these symbol strings are list structures. These symbol strings are ordered, with right hand side (RHS) and left hand side (LHS). The data base is a collection of symbols. The interpreter scans the LHS of each rule in the rule base. When it finds one rule which has the LHS that matches with some of the symbols in the data base, it executes the RHS of that rule.

4.3.1.1 Rules

A rule has LHS and RHS. One side of a rule is evaluated. If the evaluation succeeds, the other side of the rule is executed. Evaluation usually means matching the pattern in the rule to the data base. If a match is found, the evaluation succeeds. A rule can be viewed as a simple conditional statement. The LHS specifies the condition and the RHS is the action which will be carried out if the condition is satisfied. The general form of a rule will be as shown in figure 4.2.

There are many different ways of writing a rule apart
from the form shown in figure 4.2. Some rules have the form where the conditions are on the RHS; such as in Prolog and RITA [66]. These different arrangements are due to the different ways in which the interpreter has been designed. Hence the procedures are slightly different with different rule forms, the overall patterns of operation in both forms are similar; i.e., the scanning of rules, matching the conditions with the data base and performing the action.

\[
\begin{align*}
\text{C1 and C2 and \ldots and CN} & \quad > \quad a_1, a_2, \ldots, a_M \\
\text{C1 or C2 or \ldots or Cn} & \quad > \quad a_1, a_2, \ldots, a_M
\end{align*}
\]

where c(x) = a condition \( x = 1 \) to \( N \)

and a(y) = an action / conclusion \( y = 1 \) to \( M \)

Figure 4.2. General form of a production rule.

4.3.1.2 Data Base

The data base is a collection of symbols. These symbols represent the essential features of the task domain. They are (1) the static features of the task domain and (2) the dynamic states of the inference process such as the change of values of some variables. The semantics of these symbols and that of the rules form a closed world. This is an important feature in rule based representation.

Like rules, the organization of the data base can be
arranged to suit a particular application. There are ways to organize a data base to improved efficiency as well as clarity. Whatever the organization, the data base is always the sole storage medium for all state variables of the system. This contrasts with the procedural programs where there is no provision for separate storage of control state information. In a rule based system, the data base is where all the information is recorded and it is universally accessible to every rule in the system.

It turns out that the data base is the only communication channel between rules. This is because in a rule based system, one way of having rule independence is to prevent the rules from invoking one another directly. One rule can invoke only another indirectly by making changes in the data base.

4.3.1.3 Interpreter

The interpreter is the heart of the system's processing mechanism. The structure of the interpreter reflects the character of the system. There are different types of structures and they can be generalized into a basic framework of 'select-execute' cycles. In such a cycle, a rule whose conditions are satisfied by the current state of the data base is executed. Its action modifies the data base and the cycle repeats with a new rule selection phase. This sequence of operations is sometimes referred to as a 'recognize-act' cycle.

For most purposes, the interpreter can be treated as a black box which is capable of detecting changes in the data base, invoking rules and carrying out actions. It is the components inside this black box that define the characteristics and the power of the system. Hayes-Roth [67] summarizes the components and their individual roles as
The executive (interpreter), or pattern directed inference mechanism, has ..... components that communicate with the various memories (data base) to detect conditions of interest and initiate appropriate actions. It employs a "change monitor" to detect perturbations in the memory that may require attention and a "pattern matcher" to compare the observed situations with those defined as warranting responses by the inference rules ....... The "processor" calculates and implements the actions thus selected. The "knowledge modifier" effects the changes..... .

In Hayes-Roth's description, the change monitor, the pattern matcher and the processor with the knowledge modifier are the most fundamental components of the inference engine. The change monitor constantly compares the current data base with the past state and invokes the pattern matcher to find a rule which is applicable. The processor executes the action and if it entails modifying the data base, the knowledge modifier will be initiated.

4.3.1.4 Control Structure

The control structure of a rule based system is the way the interpreter carries out its flow of inference. The two most basic control structures are forward chaining and backward chaining. The recognize act cycle described earlier belongs to the forward chaining type. Systems built with this type of control structure are known as Antecedent-Driven System (ADS). For example, forward chaining has been employed in R1 [6] where as backward chaining has been used in MYCIN [3] and PLANNER [68]. In the case of backward chaining the scanning of the rules become goal directed. When a rule is executed, the LHS of the rule is evaluated. If any condition
in the LHS of the rule cannot be satisfied by simply looking up the data base, a sub goal is setup to deduce the result. This sub goal is carried out by a rule which can provide a solution to that particular condition. Systems built with this type of control structure are known as Consequent-Driven Systems (CDS).

4.4 AN UNIQUE DEMAND OF THE LEGAL REALM

The legal profession, unlike medical or engineering professions, while delivering judgments or presenting arguments before a court of law uniquely and rather compulsorily expects to intensate the effectiveness of the 'legal reasoning' process by quoting the source or origin from where the knowledge being used has been gleaned from. For example, in the medical profession whenever a doctor prescribes certain medicine to a patient, neither the patient is interested in knowing where it is mentioned that for the disease he or she has that particular medicine or treatment should be administered or prescribed nor the doctor is bothered about in letting the patient known about such source information. Similarly, while teaching "the maximum power transfer theorem" to the electrical or electronics engineering students neither the teacher mentions nor the students are interested in knowing when and where the theorem was proposed. Only they are interested in its knowledge content: For the maximum power to be transferred the output impedance of the source circuit must be equal to the input impedance of the target circuit. However, in the legal profession whenever someone is punished or acquitted of some charge it is necessary to mention as per which law, rule or the precedent the judgment is given or the argument is put forth. Nourishing source references is not only an established practice in the legal profession but also is a way of extending the natural justice and enlightening the persons involved of their rights and duties. Any lack or absence of
such source references or citations in support of one's arguments results in:

(1) Unnecessary and prolonged delay in the resolution of the problem and
(2) Subjects the 'case on hand' to the danger of discrete and unjust outcome.

The representation technique, therefore, selected to represent the legal knowledge should be capable of holding not only the chunks of knowledge as independent nuggets but also, equally importantly, should be capable of holding the source references from where the particular chunk of knowledge being used has been gleaned from.

4.4.1 Modified Production Rules

A production rule in its primitive form will have the general structure depicted in figure 4.2. Figure 4.3 presents a typical legal rule written in the form of a conventional production rule that has the knowledge regarding the eligibility of an employee for the retrenchment compensation engrossed in it.

As discussed in one of the foregoing sections legal rules of the form given in figure 4.3 and not having the source or authority references associated with them will be of little use. In order to incorporate the source of origin of the knowledge chunk being represented by a production rule the format of the conventional production rules has been modified by subjoining a third part called the ASPER part to the already existing IF and THEN parts. Figure 4.4 gives the modified version of the rule that is given in figure 4.3.
IF the employee is retrenched and
the employee works above the ground level and
the establishment is not involved in building construction and
the employee has worked for more than one year continuously
THEN the employee is eligible for the retrenchment compensation.

Figure 4.3. A sample production rule.

IF the employee is retrenched and
the employee works above the ground level and
the establishment is not involved in building construction and
the employee has worked for more than one year continuously
THEN the employee is eligible for the retrenchment compensation.

ASPERS SECTION 25FFF of the ID act.

Figure 4.4. A modified production rule.

A modified production rule, thus, consists of three parts: IF, THEN and ASPER. Figure 4.5 depicts the general format of a modified production rule.
(RULE-NAME IF ( <c1> and <c2> and ......<cN> )
THEN ( <a1> and <a2> and .....<aM> )
[ ASPER (s1, s2, ......,sL) ] )

where c(x) = a condition \( x = 1 \) to \( N \)
a(y) = an action / conclusion \( y = 1 \) to \( M \)
and s(z) = a source reference \( z = 1 \) to \( L \)

Figure 4.5. General form of a modified production rule.

It is not just a coincidence that most acknowledged computer systems in the legal realm, like EUROLEX [69], WESTLAW [70], etc., that are being used currently are legal information retrieval systems but it is due to the absolute need of the legal profession. This is because a legal practitioner always wants to have a look at the relevant precedents as well as the statutory provisions in order to analyze the case on hand and hence to substantiate his or her argument by providing proper citations. In practice, because of the very large number of decided cases as well as the very wide range of statutory provisions that exist, digging out or retrieving relevant information is not only intellectually tedious but also is time consuming. The information retrieval ability of every system, whether human or computer, crucially depends upon the indexing strategy that is made use of to organize the information in the storage i.e., memory. The ASPER part of a modified production rule, apart from satisfying the established practice of providing the required citations, can be used as indices to access and retrieve the required information automatically. Of course, such an automatic 'legal information' retrieval needs both the case
law as well as the statute to be stored in the computer by employing a suitable knowledge structure.

The subjoining of the ASPER part to the normal IF and THEN parts of a production rule has not affected any of the advantageous traits like the single channel interaction of the rules and the facility for building a knowledge-based system incrementally as well as modularly. But it has provided an additional advantage of being capable of using it as an index for the online retrieval of the related information.

As the motive behind modifying the conventional production rules is primarily to provide the all important references or citations that is demanded by the legal profession, it need not and will take part in the 'recognize-act' phase of the problem solving process. Also, whereas the normal IF and THEN parts have something to do with the human cognitive structures, the ASPER part has no such relevance. Hence, its presence in a rule during the recognize-act phase is discretionary. However, the information present in this part will be used while answering the how? and why? questions during any interactive consultation process as well as during generating the legal arguments or generating summary at the end of an interactive legal consultation session.

4.4.2 Types of Rules

Depending upon the purpose, rules have been classified into two groups called triggering rules and inference rules. The difference between these two types of rules lies in the knowledge content of their consequent parts. The consequent part of the triggering rules hold certain (control) knowledge that steers the system's attention towards some required 'concept' or 'object'. However, in the case of inference rules, the consequent parts hold knowledge regarding certain conclusions. Thus whereas the firing of an inference rule
generates a 'new fact' for the case on hand, the firing of a triggering rule provides a direction for the continuance of the analysis. The presence of just one member on the consequent part of a rule and the occurrence of the word combinations like "use section", "use part", etc., in it enables the problem solver to discriminate a triggering rule from an inference rule. For example, in figure 4.6, RL009 is a triggering rule and RL065 is an inference rule.

After having selected a 'concept' for exploration or entered into a 'situation', it is natural to draw all the possible conclusions before actually shifting his or her attention. The inference rules, therefore, have been assigned an higher priority over the triggering rules.

4.4.3 Rule forms

As suggested by Frederick Hayes-Roth [61] rules may exist in several alternative forms or translations that suit different purposes. For instance, one machine-oriented translation might serve the need for high performance at run time, another human-oriented form might use English to support publication and explanation, and so on. Following this suggestion two forms of rules called full-form rules (which will be in natural language) and terse-form rules have been employed.

As already mentioned in section 4.4.1, the ASPER part does not take part in the actual recognize-act phase of the problem solving process and hence its presence during the actual run time is not required. The terse-form rules are obtained by (i) dropping the entire ASPER part and (ii) by dropping 'noise words' like TO, IS, THE, OF, etc., from the full-form rules. The two forms of rules are stored on different files with a one-to-one relationship existing between them.
**Full form rules:**

RULE 009 IF the dispute is regarding retrenchment compensation THEN use Section 25FFF ASPER Chapter V-A of the ID act.

RULE 065 IF the number of days worked is greater than 239 and the workman works above the ground level THEN the period of continuous service is greater than one year ASPER Section 25 Part-I

**Tense form rules:**

RL 009 IF dispute retrenchment compensation THEN use Section 25FFF

RL 065 IF days worked greater than 239 and works above ground level THEN service greater than one year

*Figure 4.6 Rule types.*
Being shorter in length and hence convenient for the pattern-matching, the terse-form rules take part in the actual recognize-act phase. On the other hand, being in natural language, full-form rules are used both while answering why? and how? questions as well as during the generation of consultation summaries. Rules in their full-form aid in refining and updating the knowledge base easily. Also full-form rules extend the possibility of using the system as the "domain knowledge" tutor [71]. Figure 4.6 gives some examples of both terse-form and full-form rules.

4.5 THE NEED FOR AN ADDITIONAL CONSTRUCT

In pure rule-based systems the recognize phase of the basic 'recognize-act' control cycle itself consists of two sub-phases: Select and Conflict Resolution. During the select phase all the rules on the knowledge base are exhaustively excited and one or more potentially applicable rules are chosen (i.e., triggered) and passed on to the conflict resolution algorithm which chooses (i.e., fires) one of them. As the knowledge base grows, the exhaustive excitations of all the rules in it as well as the system's involvement in conflict resolution strategies not only reduces the efficiency of the system but also downgrades the responsiveness of the system. In the legal domain unless a particular aspect like 'closure of an establishment', 'laying off an employee', etc., is disputed neither the lawyer nor the judge considers or focuses his or her attention towards the rules attached with those aspects. This practice of the legal practitioners demands the domain knowledge to be properly organized so that only the appropriate aspects are viewed at. The problem of organization of knowledge is of paramount importance in large knowledge base domains. The problem is not only one of efficiency, but is one of focus and control. Things simply don't work if the knowledge base is not properly organized [72]. A solution that has been offered to achieve this has
been to utilize the knowledge about the knowledge. In the realm of artificial intelligence the knowledge about knowledge is referred to as meta-knowledge and the rules encapsulating such meta-knowledge are known as meta-rules. Meta-rules organize the production rules according to some meta-knowledge criterion [73]. These meta-rules are also encoded using the same format as the inference rules. In rule-based systems, rules are also used to set default values for some parameters whose values can not be determined by some other means under certain contexts. The uniformity of rule structure often forces different types of knowledge to be represented by using the same syntax, and results in what is known as a flat rule base [74]. A flat rule base hides the function of different knowledge chunks in the system and hence causes confusion on the part of the user (generally experts) who are expected to work with the knowledge base. For example, it will be difficult to determine whether a given rule was written to infer new information, to summarize information or to ensure that other rules would be invoked. Also, while production rules are natural mechanisms to model some aspect of human cognition, meta-rules seem to have no such counterpart. Thus arises the need for another construct that employs Concepts and not the Rules in order to organize the large number of production rules that hold the judgmental knowledge.

4.6 KNOWLEDGE ORGANIZATION

In the legal domain one of the important and reliable sources of the knowledge is the statute. Whether a particular law is based on customs or is due to the industrialization of a country, it will have official authenticity with it only when it is legislated in the form of an ACT. Such ACTs will have the knowledge embedded in them in the form of chapters. Chapters are made up of sections and sections are made up of one or more parts. For example, the Industrial Dispute Act of India comprises of twelve chapters, chapter V-A comprises of
Figure 4.7. Knowledge Organization.
thirteen sections and section 25-B comprises of two parts. Generally both chapters as well as sections will have a heading or title that clearly indicates the purpose of the corresponding chapters as well as the sections. For example, chapter V-A's heading is 'lay-off and retrenchment' whereas the heading of section 25-B is 'period of continuous service'.

The organization of the legal knowledge, shown in figure 4.7, forms an hierarchical tree structure. Figure 4.7 does not show the interrelationship between sections or chapters that may prevail. For instance, while dealing with "the eligibility for retrenchment compensation" one has to know "the period of continuous service". Section 25FFF deals with the former whereas section 25B deals with the latter. As a result, while administering a dispute regarding "the eligibility for retrenchment compensation", if the value for "the period of continuous service" is not known it has to be reckoned using section 25B. Such requirements have also corroborated the need of certain mechanism to shift the problem solver's attention under some circumstances.

The classification between the different parts of a section and the purpose of such parts are not readily available. A system developer has to identify these classifications carefully before any actual implementation is attempted. The mapping of the legal knowledge as it appears in any ACT into a structure like the one that is actually used is by no means automatic. This needs a thorough analysis of the concerned ACT in order to come up with a right structure and the concepts that form the structure. Such an analysis is also required because of the unnecessary structural complexity and ambiguity that exists in the statutes due to the liberal use of words like UNLESS, IF-THEN, IF-THEN-ELSE, IFF (if and only if), etc. An analysis may employ either the data flow diagrams of structured systems analysis or And/Or graphs. Because of the fact that the data flow diagrams are both too procedural as well as obscure the distinction between the
conjunctions and the disjunctions, they are not practically useful. Instead the And/Or graphs, because of their graphical syntax and ability to focus attention on logical structure rather than data, have been found to be more suitable [75] and, therefore, have been employed in this work. The actual codification warrants a bit of trial and error effort and also needs some conditions (i.e., rules) to be added.

4.7 CONCEPTS AND FORMS

In order to obtain an appropriate organization and use of the domains judgmental knowledge as well as to facilitate the required focused reasoning the Chapters and the Sections of the ACT as well as their Parts have been treated as Objects or Concepts. Such objects are identified by the phrases like 'period of continuous service', 'conditions precedent to retrenchment of workmen', etc., which are nothing but their titles or headings. These objects have been represented by frame-like structures called FORMS, employing Minsky's frame mechanism [76].

4.7.1 Forms

A FORM is a frame-like structure that ties together knowledge about a particular context, situation or concept. It is a structure consisting of many multifaceted slots. Figure 4.8 gives some of the possible slots of a FORM. Typically the facets of a slot are for storing the values, value type, default value, questions, help information, etc.
4.7.1.1 Types of slots

All the slots of a FORM have been classified into three groups as:

(i) INERT slots,
(ii) SERVICE slots,
and (iii) POINTER slots.

<table>
<thead>
<tr>
<th>NAME</th>
<th>25B</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>VALUE SECTION</td>
</tr>
<tr>
<td>HEADING</td>
<td>VALUE Period of continuous service</td>
</tr>
<tr>
<td>STATUS</td>
<td>VALUE S</td>
</tr>
<tr>
<td>MORE-GENERAL</td>
<td>VALUE CONSULT</td>
</tr>
<tr>
<td>MORE-SPECIFIC</td>
<td>VALUE (25B1 25B2)</td>
</tr>
<tr>
<td>HAS-PARTS</td>
<td>VALUE 2</td>
</tr>
<tr>
<td>FIND-FACT</td>
<td>VALUE SERVICE TRIGG (UTILIZE SERVICE)</td>
</tr>
<tr>
<td>USERULES</td>
<td>VALUE (RL051 RL052) ADOPT The service is uninterrupted</td>
</tr>
</tbody>
</table>

Figure 4.8 A typical FORM.

The slots that do not take part in any sort of reasoning process are known as INERT slots. These slots provide only topographical information about the object being represented by it. HEADING, TYPE, HAS-PARTS, etc., are some examples of the inert slots.

The slots which actually get involved in the reasoning process are called the SERVICE slots. The names of such slots...
are nothing but certain procedure names which are treated as tasks and are executed by posting them on to a task agenda. USERULES, FIND-FACT, etc., are some examples of the service slots.

The slots that indicate the hierarchical relationship among the various FORMs (objects/concepts) are known as the POINTER slots. MORE-GENERAL and MORE-SPECIFIC are the only two pointer slots that have been used.

4.7.1.2 Types of FORMs and the hierarchical relationship

Three types of FORMs called SECTION form, PART form and INTERPRETATION form have been used. The type of a FORM is identified by the value of the TYPE slot of a FORM. A SECTION form holds the knowledge corresponding to a section of the ACT. A PART form holds the knowledge corresponding to a part of a section of the ACT. An INTERPRETATION form holds the knowledge drawn from certain important precedents, encoded in the form of production rules. It is at this level that the system tries to reason with the precedents. An INTERPRETATION form is mainly used to procedurally get a value for a legal variable (for the definition of a legal variable refer to section 5.2.1) whenever it is not available explicitly. All these FORMs themselves form a hierarchical structure, as shown in figure 4.9, similar to that of medical or botanical classification system. The most general "objects" are placed at the top of the hierarchy and the most particular at the bottom. The hierarchy relationships are indicated by the MORE-GENERAL and MORE-SPECIFIC slots. This hierarchical structure serves the purpose of differentiating the knowledge, of assimilating new knowledge by inserting it in appropriate place, or retrieving the right piece of knowledge as response to a query. For legal practitioners, these structures serve the function of organizing their case analysis knowledge. The concrete details of each section, part, etc., are encoded in
Figure 4.9. Hierarchical relationship between FORMS.
the production rules attached to the FORMs. For example, RL025 and RL026 that hold the knowledge of section 25FFF have been attached to the FORM 25FFF with the help of the USERULES slot.

4.7.2 Default values / Assumed facts

Apart from the frame mechanism itself one of the most insightful ideas of Minsky [76] was to provide default values. The legal domain also needs this facility of providing such default values or assumed facts, as legal practitioners always assume or take them for granted those facts which are not under dispute and are favorable to their clients. Also on some occasions certain crucial data (fact) may not be available on the data base which causes the system to a halt condition. Missing of required facts may be either due to the non availability of the relevant information on the input data itself or may be due to the non retrieval of the fact during the problem understanding phase, which depends on the competence of the problem understander. Assuming such facts avoids the analysis of the problem on hand from "tripping over". Though the use of many assumed facts (i.e., default values) degrades the system's performance a bit it has been considered as 'graceful degradation' and preferred to a 'tripping over' situation in practice [77]. The ADOPT facet, when exists, of the USERULES slot aid in making such assumptions. As a legal practitioner may work either from a petitioner's point of view or from a defendant's point of view such assumptions are critical and cannot be automatic. Hence user's consent is taken before the actual assumption is made. Depending upon the user's response either a proposed fact or its complement, drawn from an association-list, that maintains such facts and their complement, is assumed.
4.7.3 Procedural attachment

Apart from organizing the "how" knowledge FORMs extend the facility of attaching the procedures. Procedures are attached [78] to indicate how and when certain operations are to be performed. For example, it is necessary to verify whether a particular fact is already available or not before actually attempting to either compute it or get it by querying the user. Such procedures may get activated either automatically or conditionally depending upon their position of attachment and are called demons or servants respectively [79]. The slot name FIND-FACT is a demon procedure that verifies the availability of a value for a legal variable (e.g., service) associated with its value facet. If the required value is readily available the analysis continues with the aid of the next SERVICE slot. Otherwise, the UTILIZE function (which is equivalent to the LISP's EVAL function) attached to the TRIGGER facet of the FIND-FACT slot executes the "service" procedure. The "service" procedure thus behaves like a servant procedure.

4.8 SUMMARY

In this chapter a knowledge representation technique that is suitable for representing any type of legal knowledge has been presented. Among the many existing knowledge representation techniques production rules seem to be more appropriate to represent the dynamic legal knowledge and hence has been thoroughly examined. The format of the conventional production rules has been modified in order to fulfill one of the vital requirements of the legal realm. The discussion has revealed that the use of only the production rule, even in its modified form, has certain limitations on the knowledge organization and hence its use. It has been shown how such limitations could be overcome by employing frame like structures called FORMs. Use of FORMs also has brought in all
the advantageous traits of the frame representation like procedural attachment, assignment of default values, etc., and also assists in distributed problem solving. A system called TIDA employing the "FORMs and modified production rules" technique of knowledge representation has been developed. The required domain knowledge has been culled from the chapter V-A of The Industrial Disputes Act of India.