The Use of Artificial Intelligence Techniques for Protein Structure Prediction.

by

Roslan Bin Ismail

A thesis submitted to the
Faculty of Science
University of Glasgow
for the degree of
Doctor of Philosophy

Department of Computing Science,
University of Glasgow
February,1989.

CONTENT

Abbreviations

KSAR	- Knowledge source activation record
KS	- Knowledge source
OGSS	- ordered groups of secondary structure elements

ACKNOWLEDGEMENT

I would like to thank many people especially my supervisors who have always provided me with their encouragement, guidance and help throughout the course of my study. Firstly, I would like to thank Dr.Ron Poet for his help, encouragement and supervision ever since I arrived here. My thanks go to Dr. James Milner-White for supervising me in many aspects of the work especially on Biochemistry. His patience and his interest on the work inspired me to work harder. I am grateful to Professor C.J. van Rijsbergen for suggesting the problem to work on, and many useful discussions on uncertainty techniques. I would also like to thank Dr. Stephen Todd for his encouragement on this work. My thanks to colleagues Sembok, Saeed, Riaz, Francis, Khaled, Djamal, Dr. Ruben Leon and others for sharing the life as students in the department. I pray to God for their future success.

Most of all, I would like to thank my wife, Hejar and my little daughter, Munirah for their patience and support over the past years.

Financial support and study leave was provided by Public Service Department of Malaysia from October 1985 to September 1987 and Forest Research Institute of Malaysia from October 1987 to March 1989, which is very gratefully acknowledged.

Declaration

This thesis is entirely on my original work and no part is done in collaboration. Where the work of others is used, explicit reference is made in the text. No part of this thesis has been, or is being submitted for a degree at any other university.

Dedicated to my parents, my wife and my daughter

ABSTRACT

The conventional technique for computerized protein structure prediction uses several programming languages such as Fortran, C, Pascal etc. With recent advances in programming languages and the development of rule-based systems, the computerized part of the problem is undergoing major change. This thesis sets out the idea of extending the properties of an intelligent rule-based system and recognising incomplete nature of knowledge for this problem. It reviews the existing architectures and characteristics that embody an intelligent system. As the outcome of the idea, a new system called PREDMOLL, written in Prolog, is developed. PREDMOLL is based on the blackboard architecture with several other extra features. This thesis also reviews some current uncertainty techniques and developes a formula based on a modifications of the Bayes theorem, to deal with multiple hypotheses. The problem of conditional independence assumption is reduced to the minimum. The formula is used as a decision-making criterion to determine secondary structure boundaries. For tertiary structure prediction, this thesis suggests a similarity value for primary sequence homology to overcome the problem of arbitrary uncertainty values in rules. PREDMOLL and the uncertainty techniques incorporated with it are used to test the hypothesis that the performance of protein structure prediction is improved by combining several methods. The test is carried out by a series of experimental predictions with user-defined rules and pre-defined constraints. The behaviour of PREDMOLL during the problem-solving process of the experiments is shown. The results obtained yield improvements in precision for secondary structure prediction and further improvements are expected. For tertiary structure prediction, some preliminary progress is shown and, due to lack of genuine rules,

ad-hoc rules are generated from the protein data base. The status of PREDMOLL and its advantages over other systems is discussed. Several suggestions are made to improve current facilities in PREDMOLL and problems in a wider domain. Suggestions are also made for further improvements in tertiary structure prediction

CHAPTER ONE
Introduction

1.1 AI in Biochemistry

Artificial Intelligence(AI) and its applications in the development of expert systems are important areas for future research in computer science [Jackson85]. The area is not confined to computer scientists but has attracted workers from other disciplines. As a result, many expert systems have been developed, based on cooperation between computer science and other disciplines.

Applications of AI in chemistry have been given special attention. Perhaps this is because of its complexity. For example, there is DENDRAL [Lindsay80, Barr81I, Barr81II, CohenPR82], a system to identify the organic compounds by analysis of mass spectrograms. DENDRAL is said to be among the earliest expert systems, and it is certainly one of the best known [Bramer85]. There is also CRYSALIS, which uses a blackboard architecture to infer the structure of a protein from a map of electron density derived from x-ray crystallographic data [Jackson85, Barr81II]. One should also mention MOLGEN, a system developed to provide intelligent advice to a molecular geneticist on the planning of experiments involving the manipulation of DNA [Stefik81I,Stefik81II] .

Interest in the development of expert systems for molecular biology and particularly for protein structure prediction is very recent. Interest in expert systems for protein structure prediction is due to recent progress in the technique of determining the amino acid sequence of proteins. This is a result of the revolution in the field of Genetic Engineering, where the exact molecular structure of many genes have been determined. Amino acid sequences, because they are derived directly from the DNA sequences encoding them, can be easily determined. This development initiated the idea of searching for a correlation between the amino acid sequence and the structure in 3-

dimensional space. Determining the structure of a protein in 3-dimensional space, however, is much more difficult than determining its primary structure. Hence, predicting the structure of a protein from its amino acid sequence is one of the most active research areas in molecular biology. Suggestions have been made that the application of AI in the problem, perhaps, provides considerable advantages, as discussed by Michie [Michie85]. Although attempts are still at a very early stage, a few systems for protein structure prediction have been developed.

1.2 Overview of Protein Structure Prediction

1.2.1 Motivations

Proteins are defined by Rawlings and others [Rawlings86] as linear polymers (polypeptide) of amino-acids that execute the structural, procedural and control information held in the genetic material of the cell(DNA). Since the recognition that the amino acid sequence determines protein structure, the search for algorithms to predict protein tertiary structure from amino acid sequence has been initiated. How the sequence determines a specific structure, however, has been a constant source of fascination and speculation since the problem was identified. No one, as yet, has described the transformation 'formula' to convert sequence to structure [Taylor87]. This problem which is generally referred to as the 'folding problem', has become a major stumbling block to further progress of research in molecular biology.

Determining the structure of a protein is considered as the first step towards understanding how it works, and provides a rational basis for the design of new proteins for medical and industrial use. Pardon[Neesham88] believes that the one development which would have a profound effect on the whole area of drug design, is a really good system for predicting protein structure. He emphasizes that 'Protein structure is the key area where the breakthrough has to be

made'. Sternberg [Sternberg83] listed the motivations for determining the protein structure in 3-dimensional space as: understanding at the atomic level of the function of the molecule and calculation of conformational energy, study of solvent accessibility, protein mobility, kinetics of folding and evolutionary relationships by the comparisons of tertiary structures.

1.2.2 Tertiary Structure Prediction

Three basic theoretical approaches have been developed to predict protein structure from amino acid sequence. Those approaches are:

a) *similarity of sequence*: this is probably the simplest approach. The technique is to find a protein with a similar sequence (or one that has a very high degree of similarity) and for which the conformation is already known. The unknown protein is predicted to have similar folding. This, the homology approach, uses an algorithm based on the inexact comparison of amino acid sequence. This approach suffers because of the small number of known conformations of homologous proteins (about two hundred). Furthermore, it involves an extremely large number of comparisons, and takes a large amount of computer capacity.

b) *energy minimization*: this approach uses energy calculations of atoms and molecules. The technique has limited capabilities because of difficulties in producing adequate energy functions. At present the approach is employed by the use of high-resolution nuclear magnetic resonance(NMR).

c) *semiempirical hierarchical condensation models*: this

> approach assumes that the folding problem can be divided into a series of smaller problems. The conventional divisions have been the prediction of secondary structure from amino acid sequence, the prediction of approximate tertiary structure from secondary structure, and the refinement of approximate tertiary structure. This approach is probably the most successful in practice[King87, Taylor87].

The divisions in the third approach are based on the way biochemists classify hierarchical levels of proteins as primary, secondary and tertiary. Briefly, a protein's primary structure is the linear sequence of amino acids. A protein's secondary structure is the sequence of the architectural 'units' superimposed on its primary structures. The 'units' are normally referred to as alpha helices, beta strands, turns, and etc. A protein tertiary structure is the folding of the primary and the secondary structures in three-dimensional space.

1.2.3 Secondary Structure Prediction

The difficulty in predicting 3-dimensional structure from a sequence of amino acids diverted attention to predicting secondary structure. This is because secondary structure prediction is less difficult than tertiary structure prediction. Methods dealing with predicting secondary structures are classified as empirical, pattern recognition, and others. All methods have claimed some success for predicting the secondary structures of α-helix, β-strand and β-turn.

1.2.3.1 Empirical Methods

The empirical methods to be discussed are based on Chou and Fasman [Chou74], and Robson[Robson79]. The first empirical method is by Chou and Fasman which claimed a 77% precision. The method receives attention because of its simplicity. It produces a probability

and propensity for each amino acids to be in α-helix, β-strand or turn conformation. The propensities are the frequencies of a given residue type to be observed in a particular secondary structure, normalized by the frequency expected by chance. The propensity values are rank ordered and grouped by decreasing value. The method then requires the application of several heuristic rules that attempt to determine the exact ends of secondary structures. These rules suffer from the defect of appearing somewhat arbitrary and even ill-defined [Taylor87].

The second method of empirical approach is employed by Robson and others [Robson79]. The principle is based on information theory and has a more sophisticated theoretical basis than the method of Chou and Fasman. In their method, the likelihood of a residue at position j in the sequence adopting an α-helical conformation is calculated from the location of residue types in a segment j-8 to j+8 by the formula:

$$L(a,j) = {}^{8}\Sigma_{-8} I(\alpha, R_{j+m})$$

The term I(a, Rj+m) represents the effect of the residue type at position j+m on the conformation at position j. With this approach, Robson and others quoted an accuracy of 50% for the prediction of α, β, and turn.

1.2.3.2 Pattern Recognition Methods

The method to be discussed is employed by Lim [Lim74]. It is based on a series of stereochemical rules for patterns of residue types in α-helices and β-strands. The rules consider both the size of a residue and its chemical properties, i.e. whether it is hydrophobic or hydrophilic etc. The large and small hydrophobic ones are denoted as H and h, hydrophilic ones as G and g while gly is considered independently. For example, a half-buried α-helix, in which the residue at position i points toward the core, would be expected to have hydrophobic residues at position i, i+3, i+4 or its reverse (i, i-1, i-4). β-

strands are located only in regions not predicted as a-helical. The rules were derived from all the known structures, but according to Taylor [Taylor87] the rules now seem arbitrary in the light of current knowledge of structure and structural analysis. The reported accuracy for the prediction of α-helices, β-strands and coil region using this method is 81%.

1.2.3.3 Other Methods

Taylor and Thornton [Taylor83,Taylor84] have used generated super-secondary and tertiary structures to refine the original secondary structure prediction. They employed an algorithm which contains a series of rules that determine the local conformation of the structure. Using this technique, Taylor and Thornton gained an 8.8% improvement in secondary structure prediction resulting in the correct assignment of 83% of secondary structural element in 14 proteins.

1.2.4 Precision of Secondary Structure

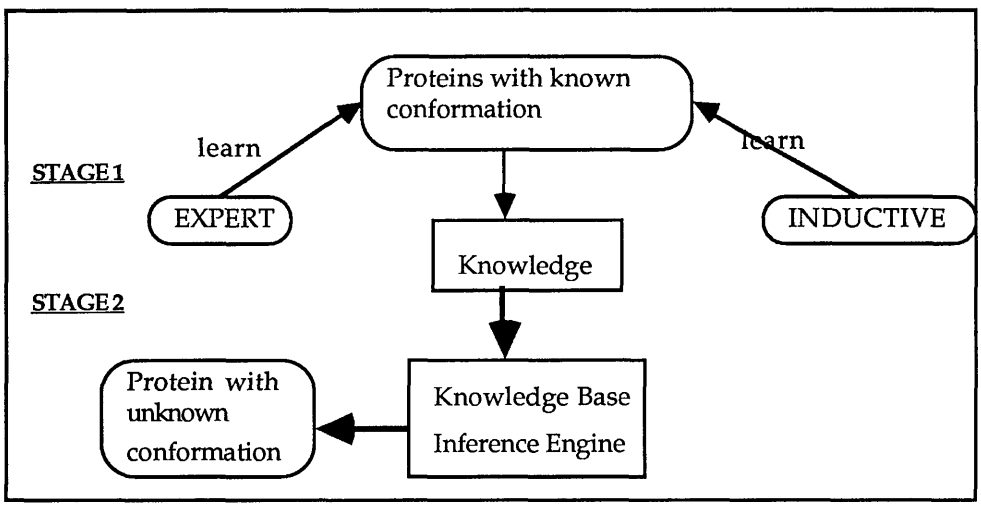
The accuracy of secondary structure is to be measured by the percentage of residues correctly assigned to all conformational states considered, including a coil or unstructured state whether they were specifically predicted or not. Nishikawa[Taylor87] has shown that the precision of a range of methods was around 50% only. This is obviously lower than claimed. And similar study by Kabsch and Sander were also found poor predictive accuracy for all those methods.

1.3 Application of Artificial Intelligence

1.3.1 Background

The development of technique in Rule Based expert systems (RBSs) has influenced the approach of computerised protein structure problems. Thus, the application of AI in predicting the protein

structure has marked a new era. Although, there are several expert systems being developed for molecular biology as discussed by Rawlings [Rawling88, Rawling87], only a few are specifically directed to the problem of protein folding. PLANS [CohenFE86], PROTEAN [HayesRothB86], ARIADNE [Lathrop87], and an inductive method by King [King87] are all intended to improve the prediction from the conventional programming technique as discussed in previous section, by means of representing knowledge in the form of rules.



Diag.1.3.1 : Stages of development for an expert system

Diagram 1.3.1 shows two stages of developing an expert system for protein structure prediction. In the first stage, it involves the extraction of knowledge from the proteins with known conformation. Rules that govern this conformation can be developed. There are two methods whereby the knowledge can be extracted. The first method is by the experts themselves going through various techniques to extract the knowledge manually. The techniques used, as discussed before are those such as empirical, and pattern recognitions in Section (1.2.3). The second method uses the computer itself instead of experts to extract the knowledge that govern the conformation of the protein. This method, generally known as 'machine learning', learns from the examples given by the users, and produces rules that can be induced from these

examples. This type of learning is called 'inductive learning'.

The second stage of an expert system development is to represent the knowledge as rules in the system and provide an inference engine. This provides a reasoning mechanism to draw inferences, make deductions, and prove or disprove rules or knowledge about the problem. Given enough knowledge about the problem, and given a good enough reasoning mechanism, an intelligent system to predict protein structure can be built. In the next sections, the way various systems have performed in predicting protein structure is discussed.

Characteristics	PLANS	ARIADNE	PROTEAN	KING
A. Description				
1.Structure To Predict	Secondary	Tertiary	Tertairy	Secondary
2. Method to provide rules	Expert	Expert	Expert	Inductive
3. Condition for input	Primary	Primary& Secondary	Atomic Positions	Primary
4. Pattern Matching	yes	yes	no	yes
5. Uncertainty	no	yes	-	no
B. System Architecture				
6. Independent Knowledge	no	no	yes	-
7. Divisions of rules into levels	no	yes	yes	-
8. Separate Control and Domain Knowledge	no	no	yes	-
9. Scheduling Mechanism	no	yes	yes	-

Fig.1.3: Comparison of various systems

1.3.2 *PLANS*

PLANS (Pattern Language for Amino Acid and Nucleic Acid Sequence), written in LISP, is used as a rule-based system to predict turns. It was developed by Cohen and others [CohenFE86] as an improvement to their system which was previously written in the programming language C. Because of the use of LISP, with its inherent advantages, PLANS is said to be a pioneer on the use of flexible recursive hierarchical pattern-matching language for protein prediction[Lathrop87]. It shows the advantage and the power of pattern-directed inference systems in predicting protein structure.

PLANS was created using diverse knowledge about turns. The antecedents are formed from complex primary sequence patterns which represent secondary structure elements. In order to predict turns, it firstly avoids locations that are predicted to favour alpha helices and beta strands. This process is called 'masking'. As the result of the richness of the knowledge about turns, the prediction is claimed to have more than 95% accuracy about turns for a test set of proteins with known structures.

The pattern language in PLANS allows residues to be grouped in various ways as well as in direct residue-for-residue matches. The pattern of the sequence is defined by several specific characters. Each character has an associated *offset*, an integer that is added to the actual sequence location of a match. The format for this pattern is as follows:

<name of pattern>	*<offset>*	*<"pattern">*
alpha_strong_phobic	0	"[ACFILM]"

In the example, a sequence is marked as the pattern 'alpha_strong_phobic' at offset 0 if a single residue matches with any of the single residues in the group "ACFILM". The characters '[' and ']' mean logical OR in that pattern.

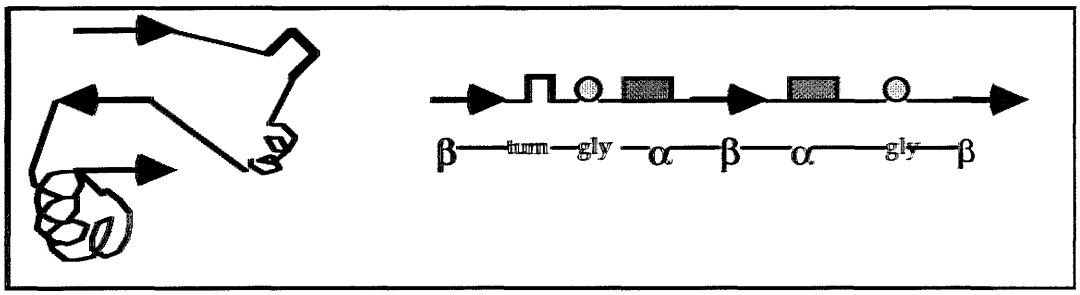
The obvious advantage of PLANS compared to conventional techniques is due to the use of the programming language LISP. The accuracy of the prediction is determined by the rules being applied. Although in Fig. 1.3 indicates that PLANS predicts secondary structure, the rules are for predicting turns only. There are no rules at present to predict other secondary structures, let alone the tertiary structure.

The rules in PLANS are similar to those of Lim[Lim74]. PLANS tries to avoid using a threshold value for each of its rules. Thus, there is no element of uncertainty in the rules, although it is accepted that the patterns in the rules are not always certain. Pattern descriptors in PLANS are hierarchically related such that patterns are grouped together to form other patterns. However, the various patterns do not really represent independent rules because they are interrelated and without one rule the others are affected. Again, there is no layer approach in PLANS to provide better efficiency in the scheduling mechanism. As shown in Fig. 1.3, there is no separation between control rules and domain rules in PLANS, and there is no focus or policy for implemeting the scheduling mechanism. Although rules used in PLANS are acknowledged to provide a higher accuracy to the prediction of turns, PLANS itself does not provide a mechanism for proper control strategy needed in expert system implementation.

1.3.3 *ARIADNE*

In contrast to PLANS, ARIADNE facilitates direct expression and manipulation of higher order structures, allowing direct use of secondary structure prediction. This approach improves the limitation of techniques such as PLANS, which relies on a substantial primary sequence similarity in order to make inferences about protein structure. It is generally accepted that similar primary sequences indicates a similar folded conformation, but the converse does not usually hold [Lesk80]. This is because secondary and higher order

structures are important in forming required spatial configuration, but by no means necessarily exhibit recognizable primary sequence patterns.



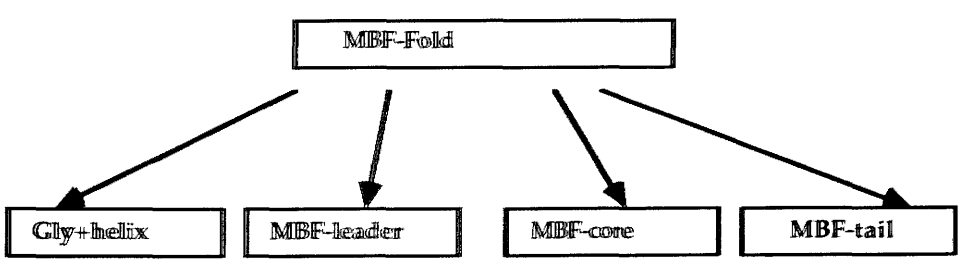
Diag. 1.3.3a: Unfolded 3-D

The pattern descriptor for antecedents in ARIADNE is the unfolded three dimensional structure that forms as a sequence of primary and secondary structure elements. The similarity of this pattern is used to predict the tertiary structure. As shown by the hypothetical super secondary structure in Diag 1.3.3a, the pattern descriptor is as follows:

β—turn—gly—α—β—α—gly—β

The "-" means any numbers of residues between secondary structure elements. The pattern descriptor is then compared with the amino acid sequence of proteins of unknown conformation. The highest degree of similarity and greater than the pre-determined threshold value gives the correct pattern. Thus, the sequence of unknown conformation being assumed to fold similarly to the known conformation.

****



****

Diag.1.3.3b: Rules in ARIADNE

In the implementation of ARIADNE, a pattern of an ordered group of secondary structures(OGSS) is used as a rule. In its example, five patterns are defined: Gly+helix, MBF-Leader, MBF-core, MBF-Tail, and MBF. As shown in Diag. 1.3.3b, a pattern MBF is formed from the other four patterns.

The pattern is defined according to the OGSS or by its combination with adjacent residues. As no perfect matches are expected, uncertainty values are attached to the pattern. However, the basis for the values seem to be arbitrary. An example for a pattern is as follows:

```
*****************************************
(defpattern MBF-Tail
  (pattern
      '(( D: score-if-missing -0.5),
        (Spacer: min 2 :max 2    ),
        ( G : score-if-missing -0.333)).
*****************************************
```

Fig.1.3.3(c): The pattern of MBF-tail

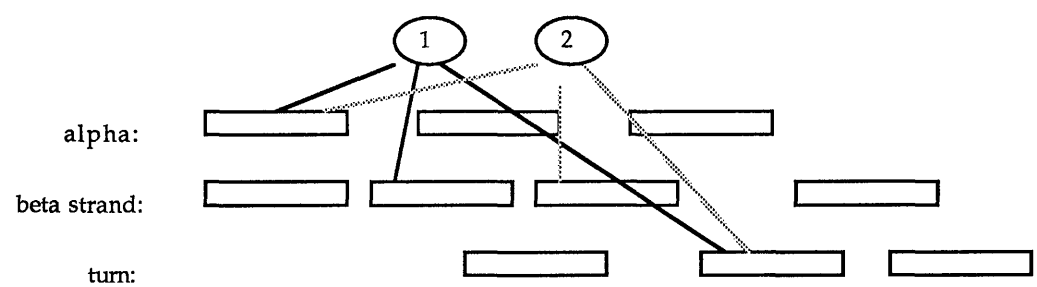
The definition for the pattern 'MBF-Tail' contains several flexible patterns. The absence of patterns is complemented by penalties of negative scores. Beside the doubt about the basis for uncertainty in the definition, there is no uncertainty value attached to the reliability of the pattern 'MBF-tail'. The absence of the value implies the certainty of the consequent pattern, which is doubtful.

The input for ARIADNE consists of the primary sequence and predicted secondary structure in addition to pattern descriptors as

antecedents in the rules. ARIADNE does not predict secondary structure, so, the predicted secondary structures as its input are generated from an established predicting system. As ARIADNE assumes that a perfect predictive method for secondary structures have not yet been developed, the unsolved predicted secondary structures are retained to be used as the input, along with the primary structure. Diag.1.3.3d illustrates the decision taken in ARIADNE where the predicted secondary structures are not solved for a particular location. Consequences of the decision are:

a) the number of possible locations for pattern are unnecessarily higher. Furthermore, the secondary structures are treated as certainty.

b) Since there are no methods for managing uncertainties for the pattern, the reliability of the successful pattern is affected.



Diag 1.3.3d: Combination of selection in ARIADNE

As shown in Fig.1.3, the system architecture in ARIADNE consists of all the features in the system architecture given except the separation of control and domain knowledge. In fact, there is no control knowledge source in ARIADNE and control problems are solved implicitly. The selection of tasks to perform is based on the degree of similarity throughout the problem solving. There is no

facility to solve the control problems and no mechanism to implement it. Thus, the system does not solve the problem according as in a dynamic problem-solving system.

1.3.4 *PROTEAN*

PROTEAN explores a constraint-based approach to infer the protein three dimensional structure directly. The method used in PROTEAN is different from the previous techniques, in that it derives from the use of high-resolution nuclear magnetic resonance (NMR), to determine the size and shape of the protein, identify the atoms, and the distance between the atoms. The technique is made possible with the knowledge about the architectural characteristics of secondary structures, atomic structures of all amino acids, and the radius of each atom. These data from NMR, taken together with the knowledge that have been developed, substantially constrain the space of plausible tertiary structure.

PROTEAN uses a variety of empirically derived constraints to identify valid positions for each of the proteins constituent atoms in three dimensional space. PROTEAN's problem belongs to a sub-class of constraint-satisfaction problems in which physical objects must be positioned in n-dimensional space so as to satisfy a set of constraints. So far it has been tried only on helices and coil.

As shown in Fig.1.3, PROTEAN consists of all the features listed under *System Architecture*. All the rules or knowledge sources are independent of one another. The knowledge sources are divided into levels, and domain knowledge sources and control knowledge sources are separated. The domain knowledge sources are divided into levels, *atom, superatom, solid, and molecules*. The control knowledge sources are grouped into several levels to reason about the problem solving strategy. The adaptive scheduler uses the current control plans to determine which knowledge source should determine its action at

each problem solving cycle.

1.3.5 Inductive Learning

King [King87] has led a new way of predicting secondary structure by using inductive learning. The method uses known structures like α-helices and β-strands in the learning process. The residues are first classified based on chemical knowledge, and the sets of residues formed into a structural description of the sets (a generalization lattice). The lattice is then used for the induction. The rules, based on sequences of sets, are induced which recognize secondary structure. The rules generated through inductive learning by King are claimed to be more accurate compared to the similar rules created by Cohen [CohenFE83].

There are thirteen rules produced by this method of which eight rules are for alpha helices while four rules are for beta strands. Since the technique is based on inductive learning, the rules are generated on positive samples. It does not give any guidance when overlapping occurs for different predicted secondary structures. Fig. 1.3 indicates that this is the only method in the list that does not use the expert to provide the rules. However, to learn the rules, the system should also need to be taught what to learn and from what examples. So, an expert who knows about the subject is also needed. Apart from this, the method provides a potential technique to understand and learn the rules that govern the mystery of tertiary structure as well as secondary structure of protein.

1.4 Discussions

In the previous section, the status of protein structure prediction has been described and discussed. In this section, several suggestions in

view of the current status of prediction, will be discussed. These suggestions are taken into account for its implementation to be carried out in this work.

1.4.1 Prediction Method In This Work

The implementation of protein structure prediction in this work is done in two phases and it is based on amino acid sequence as its only input. In the first phase, the sequence is used to predict secondary structure while in the second phase, the predicted secondary structure along with the primary structure are used to predict the tertiary structure.

1.4.2 Combination of Methods in Predicting Secondary Structure

In section 1.2.3, the reliability of a range of methods was described to be lower than expected. The methods which are technically independent of one another, perhaps, should be combined to produce a better result. The rationale of the idea is that the additional methods provide additional locations for secondary structure. This allow more locations to be predicted. The overlapping locations with similar secondary structure which are predicted by different methods, perhaps, can be used to increase the reliability.

In this work, three independent methods are chosen. The methods are based on the statistical approach, the stereochemistry approach based on Lim's rules [Lim74], and the inductive learning approach based on King's rules [King87]. The last two methods were discussed in section 1.2.2. For the first method, several simple heuristic rules are developed. The propensity value, which was also adopted in Chou and Fasman [Chou74] is used. However, in contrast to the method adopted by Chou and Fasman, a theory for the formulation of uncertainty is developed.

1.4.3 Tertiary Structure Prediction

The accuracy of tertiary structure is not usually available. This is understood due to the difficulty of predicting tertiary structure. In secondary structure prediction, several independent methods are used, with varying degrees of success. In tertiary structure prediction, however, homology is the only widely used method, and provides direct transformation of amino acid sequence to tertiary structure.

The knowledge that the secondary structure sequence can be used to predict tertiary structure, appears to reduce the problem. ARIADNE, is one of the systems that adopts this approach. The problem, though, is not easy. Generating rules, i.e. the tertiary patterns and their pattern descriptors as antecedent, is one of the fundamental problem in Biochemistry.

The difficulty of generating rules for tertiary structure is shown in ARIADNE where only one rule is used. In the definition of patterns for the rule, the basis of the 'score values' is not known. Furthermore, the reliability of the definition to predict the pattern is not provided. As mentioned, the problem that initially appears to be an uncertainty problem, becomes a certainty problem.

In this work, the difficulty in providing genuine rules is acknowledged and maintained. The temporary solution for this problem is by generating the rules directly from the protein data base. Several patterns which regarded as the regular patterns in protein data base are recorded. Rules to predict the patterns are generated. As the consequence, for a pattern 'X', rules dedicated to a particular location in proteins are generated.

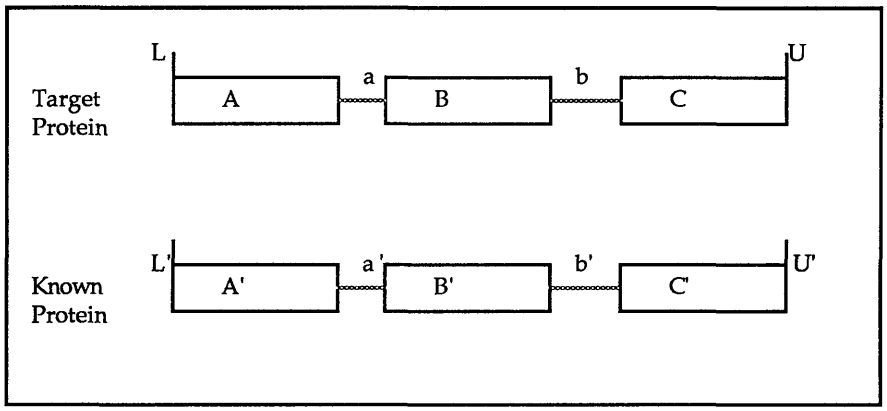
The antecedent in rules to be generated in this work contains exclusively OGSS. Neither score values nor primary structure information are used in the antecedent as in ARIADNE, because there

is no information about them. However, this defect is covered by the inexact primary structure homology between the target protein and proteins with known confirmation. This approach, perhaps, is complementary to that in ARIADNE which does not use primary structure homology as a basis of uncertainty.

Similarity of primary structure is useful to determine similarity of alignment and folding. In previous work in section 1.1.2, homology is performed for the entire amino acid sequence. The comparisons to be made for the approach are extremely large. In contrast, this work is concentrated at locations where exact similarity of OGSS is found. The steps of the approach adopted in this work are as follows:

a) Search the possible locations for patterns of tertiary structure in the target protein. The search is defined to be successful when the exact OGSS is found. The location in the target protein where the similarity is found is recorded. In Diag.1.4.3a, shows that the similarity of secondary structure sequence *A-B-C* in the target protein is recorded at a location from *L* and *U*.

b) The similarity is based on the inexact homology between the location in the target protein (from *L* to *U*) and the location in the known protein (from *L'* to *U'*).



Diag.1.4.3(a): Location for primary sequence homology

c) The similarity between the 'coil' (defined as the sequence of amino acids between two secondary structure). In Diag.1.4.3a, the locations of 'coils' to be compared are between *a* and *a'* and between *b* and *b'*.

1.4.4 Uncertainty

The uncertainty technique is relevant in protein structure prediction. Although the problem in dealing with uncertainty is widely accepted in AI, it is not in protein structure prediction. The absence of any credible technique to deal with the uncertainty situation, resulted in it being ignored. Several methods that have been discussed, avoided the problem [Lim74, King87, Lathrop87, CohenFE86]. Thus, in the current predictive methods, the reliability of prediction for a secondary structure at a particular location is not available. The reliability is only measured by the overall result for a protein as compared with a protein of known structure. As a consequence, there are no facilities to reason why a particular secondary structure should be adopted instead of others for a particular location. The method by Chou and Fasman[Chou74] uses the average of propensity values for that location. However it has the difficulty of determining the border and the decision about overlapping. The method by Lim on the other hand

avoids overlapping by predicting beta strands only outside the predicted alpha helices. So, these methods do not state the degree of certainty for adopting a particular structure or the degree of certainty for rejecting others.

In this work, the uncertainty problem in protein structure prediction is acknowledged. A technique to manage the uncertainty is developed. In secondary structure prediction, there are three aspects of uncertainty involved. They are as follows:

a) The degree of certainty for a secondary structure to be adopted by an amino acid sequence. The uncertainty value forms a basis for accepting or rejecting the possibility of the location adopting a particular secondary structure.

b) When conflicting secondary structures compete for a common location, the uncertainty method provides a basis for a better decision which improves the reliability of the prediction.

c) When several methods predict overlapping and similar secondary structures at a location, the uncertainty method increases the likelihood of that location adopting the secondary structure.

Several uncertainty methods are used in expert systems. However, two methods of uncertainty are considered for protein structure prediction The first one is based on Bayes theorem while the second is based on the Dempster-Shafer theory of evidence. The Bayes theorem is suitable because the probability value can be generated from the protein data base. Dempster-Shafer belief function is suggested because several predictive methods are used. The propagation technique is required to compute the revised 'belief' of a location to

adopt a particular secondary structure where overlapping occurs.

1.4.5 System Architecture

In previous sections, an approach in predicting protein structure has been proposed. It is illustrated as below:

> It has to predict secondary structure before tertiary structure is predicted. In predicting secondary structure, several predictive methods are used. The methods are performed one after another. When they conflict, decision has to be made based on .....

The above illustrates part of the approach in problem solving for protein structure prediction. In the implementation, the control plan for the way the problem has to be solved is implemented. Since there are many rules to follow, these rules are classified under control problems. In order to provide an efficient and intelligent problem solving behaviour, a system architecture these capabilities has to be designed and developed.

Fig 1.3 shows several characteristics that such and architecture should have:

a) The rules or knowledge sources for the system should be independent. The independence of the knowledge sources allows modification, addition and deletion of knowledge sources to be done easily without reorganizing and affecting the whole system.

b) The division of knowledge sources into levels provides efficiency and reduces complexities in scheduling. The efficiency is achieved by providing the control plan for each stage of problem solving. This allows the the system to consult only the relevant knowledge sources

at each stage instead of all of them.

c) The separation of control and domain problems provides facilities to concentrate a particular problem and a means to implement it. In solving control problems, the system decides what knowledge source to perform and at what point of the problem solving. This includes the strategy to implement the plan on how to solve problems, and decisions to focus on the most interesting problems. The domain knowledge sources concentrate on solving the domain problems and its action is executed when the condition is satisfied.

d) The selecting and scheduling of the knowledge sources are done by the adaptive scheduler which responds to the current focus and policies of the problem-solving in response to the dynamic problem-solving situation.

1.4.6 Programming Languange

The choice of programming language is important in this work. The application of AI has facilitated the knowledge about proteins to be easily represented. The previous difficulty in implementing Lim's rules using conventional programming technique has prevailed [Taylor87]. Cohen [CohenFE86] when implementing PLANS, quoted that conventional programming languages such as C, Fortran, Pascal and others are not suitable for pattern matching problems. In implementing PLANS, the programming language LISP was used and proved to be successful.

Besides LISP, Prolog is another programming language that is intended for AI problems. It is based on logic and provides the facility for inference mechanisms. Rawlings [Rawling86] used the language to analyse and represent motif topologies. Morfew and Todd [Morffew86]

used Prolog for the purpose of a protein query language to represent the patterns, somewhat similar to those in PLANS. Apart from representing the knowledge, Prolog has also been used for an expert system shell [Jones86] and this was found to be easy to develop. As a result, Prolog has been chosen as the programming language in the implementation of the present work. It facilitates development of control strategy, knowledge representation, and other tasks.

1.5 Summary and guide to the next chapters

Several suggestions to be implemented in this work have been described. Powerful tools to predict protein structure are proposed: the uncertainty technique is used to provide the reasoning behind the structure to be adopted; several predictive methods are used to provide a better precision for prediction; primary structure homology and secondary structure patterns are used for tertiary structure prediction; a system architecture provides an intelligent control strategy for the problem-solving . Chapters to discuss various aspects of these suggestions and its implementation are as follows:

Chapter Two concerns the design of an intelligent system to be called PREDMOLL for predicting protein structures. The main emphasis is given to control problems and decisions to be taken during the solution of the problem-solving. Discussions include knowledge representation, division of control problems into several levels, generic knowledge sources to be used and scheduling mechanisms. Comparisons of this architecture with others are discussed.

Chapter Three is a discussion on uncertainty techniques used in the prediction. Firstly it discusses the uncertainty problem in protein structure prediction and approaches to problem-solving. Secondly there is a discussion on the theoretical problems of uncertainty techniques. A probability approach based on the Bayes theorem and

the Dempster-Shafer evidence theory are selected for application. Bayes theorem is given attention in this Chapter. Criticisms on the validity of this method are discussed. A new procedure for multiple hypotheses has been formulated. The advantage of the new procedure is that the value of the overall probability of the hypotheses, in the light of new evidence, is maintained at one throughout the propagation. Furthermore, the problem of conditional independent assumption is reduced. Finally it discusses the application of the newly formulated technique in solving the uncertainty problem in protein structure prediction.

Chapter Four discusses about the analysis of protein data bases as a means of generating probability values for secondary structure and generating rules for tertiary structure. The protein data-bank used by us holds the data for fifty four high resolution proteins selected from the Brookhaven Data-bank. Hydrogen bonds and dihedral angles have been used as criteria to identify location of secondary structures. The probability values are based on the occurrence of individual amino acids within the observed secondary structure. These values are used to guess the locations of secondary structures by means of an empirical approach. Generating rules for tertiary structure is based on regular motifs formed by an OGSS. These rules provide PREDMOLL an implementation for tertiary structure prediction. Finally, this Chapter discusses PATTERN, a program used to locate a pattern of complex structures(beyond primary structure) in proteins. The program improves the facility of pure PROLOG in locating patterns as well as in providing significant contributions in other areas in this work.

Chapter Five discusses the performance of three predictive methods, namely the method 'stereochemistry'[Lim74], the method 'inductive'[King's87] and finally a method to be called 'statistic', which is developed in this work. The performance is based on comparisons between the prediction result and the actual secondary structure of the protein database. Investigations of the application of an uncertainty

technique on the predicted result are explored. Improvements based on the information about overlapping locations of several predictive methods are investigated. The values generated by these investigations are discussed.

The implementation of PREDMOLL is discussed in Chapter Six. The implementation involves a series of protein structure predictions, from a single method of secondary structure prediction to a tertiary structure prediction. The way PREDMOLL solves its control problems is discussed. Implementation of PREDMOLL for control problems of other expert systems is also discussed.

Chapter Seven discusses the overall contribution of the present work. It includes the status of PREDMOLL as a means of providing an intelligent system for protein structure prediction and for general problems. An example of implementing PREDMOLL with other prediction systems is also discussed. The status of the uncertainty techniques used are also reviewed. Discussions involve its role in providing a robust technique to determine secondary structure boundaries. Weaknesses of the present technique are discussed. Discussions on prediction results provide a better perspective of the advantages and limitations of this work.

Conclusions and future research are discussed in Chapter Eight. For PREDMOLL, this includes ideas about improvements in the technique and better interactive facilities. For improvements in protein structure prediction, the need for provision of more rules for secondary and tertiary structure is anticipated.

CHAPTER TWO

Control Strategy and Rule-Based System

2.1 Rationale of RBSs

Problem-solving in protein structure prediction involves a diverse source of knowledge which needs to be managed. This knowledge can be classified into two broad categories. The first category is the knowledge about the problem that has to be solved(to be called the 'domain problem'). The second category is the knowledge about how the problem should be solved by using knowledge in the domain problem(to be called the 'control problem'). Knowledge for domain and control problems are represented in rule form. This is because Rule-based systems (RBSs) are perhaps the best currently available means for codifying the problem solving know-how of human experts(or expert systems). They address a need for capturing, representing, storing, distributing, reasoning about and applying human knowledge on a computer system.

The RBS was originally proposed by Post(1943) and has since been used in various applications [Frost86]. The first application was used in the specification of grammars and the construction of parsers for programming languages. The system provided a representation of decision logic for transaction processing and report generation. In the late sixties to the middle seventies, several notable systems were developed, DENDRAL[Buchanan69], MYCIN[Shortliffe76] and the HEARSAY speech recognition system [Lesser75]. Since then, the approach has been used extensively in the construction of expert systems in a wide range of applications and is acknowledged as the best means of building expert systems that incorporate large amount of judgemental, heuristic and experimental know-how.

Although many different techniques emerged for organizing collections of rules into automated expert systems, most RBSs share

certain key properties :

a) RBSs uses conditional IF-THEN rules to represent the problem-solving skills of experts;
b) Existing knowledge can be refined and new knowledge added for incremental increases in system performance;
c) The system is intended to solve a wide range of complex problems by selecting relevant rules and then combining the result in appropriate ways;
d) The scheduler in the system schedules the best sequence of rules to be executed;
e) The system can explain conclusions by retracing the reasoning for decisions that have been taken.

2.2 Control Problems in Expert Systems

As a new technology, RBSs have gone through many stages of improvement. Despite increasing sophistication in problem-solving knowledge and heuristics, most RBSs employ relatively simple control programs [HayesRothB85]. However, the pre-determined control program is in contrast with the way experts guide their problem-solving. Instead, they always planned how the problem should be solved by planning strategies for problem solving. Using knowledge about a current problem, they will use their experience on: solving a similar problem; reasoning for taking a particular decision; planning the sequence of next actions; and interrupting the current problem-solving when some unexpected behaviour occurs. Thus, a prescriptive control plan is composed out of modular control heuristics. Then the control knowledge sources respond to, generate, and modify the solution elements on a control blackboard, under the control of a scheduling mechanism. The scheduler chooses the most feasible action to be executed.

The central issue in the control problems is which of the rules the

system has to execute at particular point. This arises when several potential actions may be possible to improve the solution at that point. To evaluate the actions, the scheduler consults the system about the current interest of problem solving and chooses one of actions that maximally contribute to the solution of the problem. In this case, the concept of RBSs differs radically from von Newmann architectures. In RBSs the control involves in an iterative cycle of: identifying the heuristic rules that bear on the current problem at interest; and applying a rule to solve or simplify it.

2.3 Evolutionary Process of Control Architecture

The radical differences in concept in RBSs and von Newmann architecture are a result of a series of evolutionary process. However, all control architectures throughout the evolutionary process have retained the three basic steps as follow:

a) identify the set of permissible next computations;
b) select the next computation from the among the permissible computations;
c) execute the next selected computation.

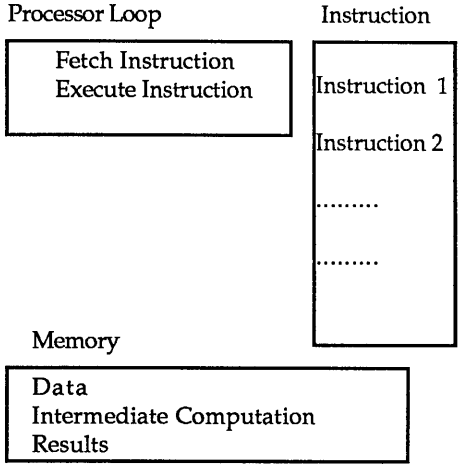
The evolutionary process is divided into three stages. The first stage is based on von Newmann architecture, the second stage is based on the production system, and finally, in the third stage, it is based on a layers approach.

2.3.1 von Newmann

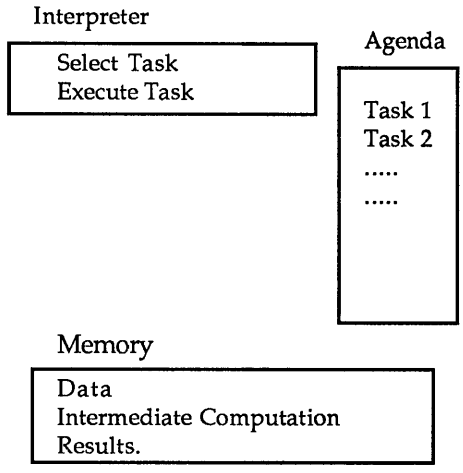
This architecture is considered as the first architecture for control strategy. It has been used in digital computers for constructing compilers for programming languages. The three steps use are as follows:

a) load the current contents of the instruction counter (an address);
b) fetch the next instruction by fetching the content of the loaded address;
c) execute the fetched instruction (a single instruction).

The execution of the instruction causes the changes in the memory. The architecture is shown in Diag. 2.3.1 as below.



Diag:3.2.1: von Newmann architecture



Diag3.2.2: Production System

2.3.2 Production System Architecture

The control architecture of a production system as shown in Diag. 2.3.2 is divided into three stages. However all the three have similar steps as follows:

a) Form the conflict set;
b) Resolve the conflict set;
c) Execute the selected rules.

In the first step, the production system recognizes a set of n permissible next computations as the set of productions whose conditional parts are true under the current problem solving using simple pattern matching. In the second step, it resolves the conflict using the method as will be shown in several sections below. In third step, it executes the selected computation which is the action part of the rule. This architecture defines a pattern-directed programming environment. The conflict resolution strategy for the second step has the following evolution as follows:

2.3.2a Black Box Approach

The architecture as shown in Diag 2.3 indicates some similarities with von Newmann architecture. However, the tasks are usually more complicated than machine instructions and the selection criteria are richer. In this architecture, it is taking the first task found to satisfy a condition [Frost86,Feigenbaum81,Stefik81]. Thus, tasks are executed in order, the order in which they are created. This approach lacks scheduling information in the interpreter.

2.3.2b Priority Approach

This architecture is basically the improvement from section(2.3.2a) which includes the facility for scheduling by assigning a numerical value to each rule [Frost86, Stefik81]. The highest priority is the task that has the highest value in the agenda. It is then be scheduled and executed. There are several weaknesses associated with this architecture. Firstly, the difficulty of assigning a value to each rule. This is because the rules have different priorities at different points in the problem solving process. A fixed value limits its usefulness at various points in the problem solving. The second weakness is the difficulty for the system to know all of the tasks in order to assign the value. So, eventhough the architecture provides a mean to deal with the scheduling process, it fails to provide a control mechanism to deal with the current status of the problem solving.

2.3.2c Task Centred Scheduling

This architecture is an improvement to the numeric priorities in section(2.3.2.b). In this architecture, a function is assigned to each rule. The function calculates the priorities of the rules at various points in the problem solving. Although it is more complicated than numeric priorities, the efficiency is not much improved [Stefik81]. In the case of

multiple goals, each functions should be able to take account of all of them. In the worst case, the functions for each task need to take account of all the other possible tasks. These weaknesses lead to a combinatorial explosion for possible interactions between tasks. This is because the number of interactions is equivalent to the square of the tasks, and, for multiple goals, the number of interactions grow exponentially.

2.3.3 Layers Approach

In the production system, the scheduling can be unmanageable due to the complexities of interactions among tasks. This is because the approach does not provide a hierarchical framework for complex control strategy. It provides no concept or global perspectives to bear on scheduling. The layers approach is proposed to overcome this complexity by organizing the problem into various levels of abstraction. This approach is used by Meta-Planning and Blackboard architecture. In MOLGEN where the Meta-Planning is being implemented, the basic idea of agenda ia extended by having another agenda to decide on the level of the problem. After the level has been decided, the system decides on the tasks that belong to the level of the problem.

A blackboard architecture is another architecture that recognizes the meta-level of a problem. The problem is divided into various levels of abstraction and rules which are called *knowledge sources* (KSs), are also assigned to these levels. If the idea of agenda is extended in MOLGEN by having another agenda to deal with meta-level problems, the agenda in pure blackboard architecture uses only one agenda. The three step process for a blackboard architecture with sophisticated scheduler is as follows:

a) Update the agenda;
b) Schedule a pending rule;
c) Execute the scheduled rule.

The three steps shown above do not radically differ from the three steps as in the previous architectures. However, in the second step where the scheduler selects a rule to be scheduled, it uses knowledge about the current interest in the problem solving, the problem states, the rule characteristics, scheduling heuristics and others. In the third step, the action part of the scheduled rule is executed which normally has the computational power of a program of production rules. This architecture permits multiple, possibly conflicting, scheduling criteria and an intelligent flexibility in program performance.

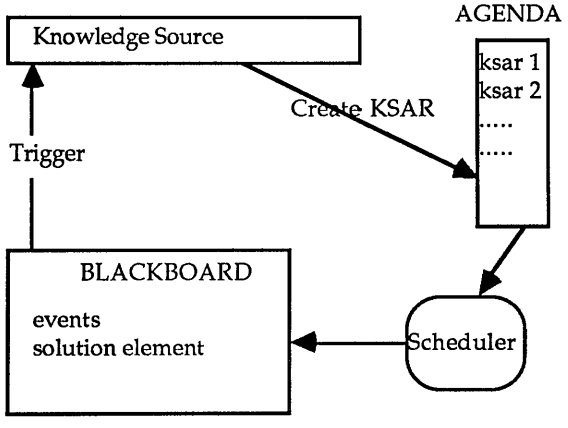
2.4 Blackboard Architecture

2.4.1 The Principle

Blackboard architecture was originally intended for HEARSAYII, a system for natural language problems, but later on was used successfully beyond it. It is intended for domains in which large amounts of diverse, errorful and incomplete knowledge are used to solve problems. A means of organizing the application of this knowledge and the cooperation between them is provided. The architecture is often described as an hypothesis-and-revise paradigm. The problem-solving is approached as an incremental development of solutions, subject to possible revision at all stages. The emergence of solutions are in the form of islands , independent fragments that expand and merge to form the overall solution. Events and consequences are the partial or intermediate results that the system produces while attempting to solve the problem.

The blackboard is a global data structure used to organize the events and consequences. It provides a means of handling communication between independent knowledge sources. The blackboard is divided into a number of planes corresponding to

different aspects of solution process such as the control problem and domain problem. Each plane is divided into a number of levels corresponding to representations at some levels of abstraction of that aspect of the solution. The system works in a cycle. With the addition of events and consequences to the blackboard, the system will select which KS that can possibly utilize the new information. To do this, it will evaluate the condition of a KS. If true, a new task is added to the agenda. The entry to the agenda is called knowledge source activation record (KSAR). It contains documentations of the the condition satisfaction of KS, its intended action and user defined measures of reliability, usefulness, etc. In agenda, the KSAR is treated as a pending task and is so rated by the scheduler. The highest rating of KSAR is selected. The action part of the KS is executed and proposes changes to the blackboard. A new state of the blackboard is produced and the process begins again with the construction of a new agenda.



Diag. 2.4 : Blackboard Architecture

2.4.2 Variations in Blackboard Architecture

Pure blackboard architecture as mentioned above has also gone through many stages of development. However, these developments are strongly associated with the type of applications that the system was originally intended for. Early types of blackboard architecture used in HEARSAYII and CRYSALLIS do not separate between KS for domain

and control. Later, HEARSAY-III was developed, based on a uniform blackboard environment with separate domain and control blackboards and default a low level scheduler that simply schedules any pending KSAR. The system was intended to support general control problem-solving. However, according to Hayes-Roth[HayesRothB85], HEARSAY-III still does not provide any theory for an effective control for problem-solving or for implemention of control.

Control blackboard architecture [HayesRothB85] was developed as the complement of HEARSAY-III. It provides a theory of effective control behaviour i.e. the behaviour that motivates it, and the mechanism to implement it. It uses the same three steps as pure blackboard architecture but introduces several innovations. In step one, it generates an expanded To-Do-Set of permissible computations and divides it into two categories of task. The first category is tasks that are triggerred by the current state of the solution and events generated by the previous tasks. The second category is tasks from the first category of which their actions are feasible to be performed. In step two, feasible tasks are rated according to the dynamic scheduling criteria to select one of pending KSARs. The highest rated KSAR is scheduled to be executed. In step three, the scheduled KSAR is executed, of which it may be either a domain application action or a control action whose consequences reflexively influence step two. These innovations are introduced in control blackboard architecture to provide a flexible environment for explicit and dynamic control solving: interpretable representation of actual control plan; and a uniform mechanism for integrating domain and control problem-solving.

2.5 Control Architecture in PREDMOLL

The architecture of PREDMOLL is derived from earlier architectures in that it separates the knowledge sources on domain problems and from those on control problems. Basic controls are similar to any blackboard architecture and has three steps of operation

as follows:

a)Update Agenda:

a1) Due to changes in the blackboard, the trigger mechanism is activated. In PREDMOLL, it consults the control blackboard to get advice on its triggering policy.

a2) When trigger conditions of a knowledge source are satisfied, a unique number of a KSAR is assigned along with other documentations associated with it. The KSAR is stored in a trigger list.

a3) For each of KSAR in the trigger list, its feasible conditions are validated. If they are satisfied, the KSAR is moved from the trigger list to a feasible list.

b) Schedule a set of KSARs:

b1) For each of KSAR in the feasible list, its priority to be scheduled is rated by an adaptive scheduler. The scheduler consults all current decisions in the control blackboard. It computes a rating value for the KSAR. The scheduler then schedules a set of KSARs of which have been decided by current control decisions.

c) Execute each KSAR in the set:

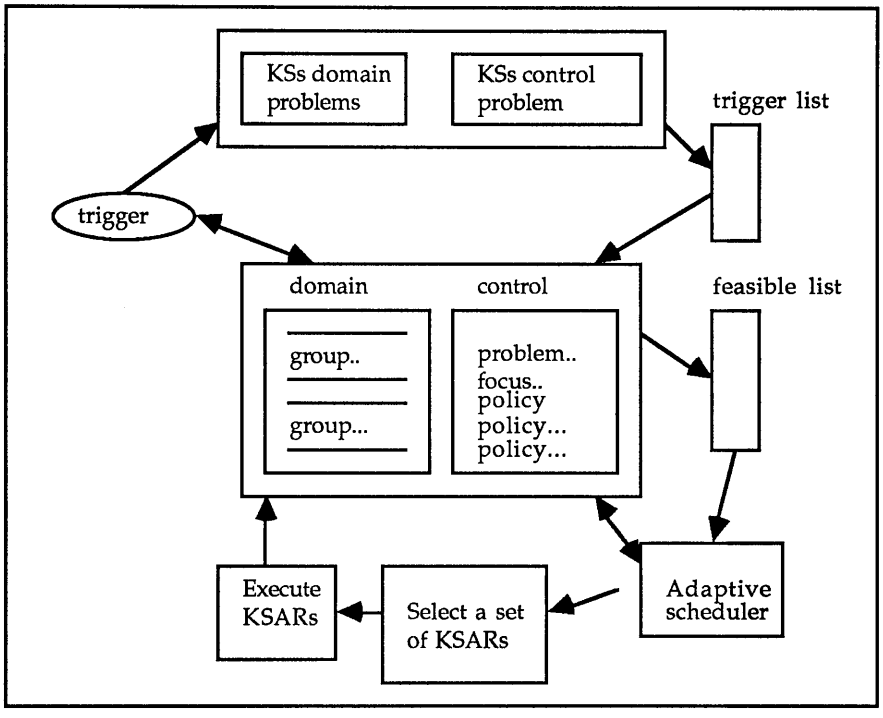
c1) Each KSAR in the set is executed. Their actions change the state of solution in blackboards. Thus, the process is continued in the next cycle.

Although basic steps of the operation are similar, it differs slightly in the actual implementation. These differences are shown during discussions on components of PREDMOLL. Its components are:

- knowledge sources;
- domain problem;
- control problem;

- basic control;
- scheduler.

The descriptions of each component are discussed in the following sections.



Diag 2.5: Control Blackboard of PREDMOLL

2.6 Knowledge Source

A knowledge in PREDMOLL is represented by the knowledge source in a uniform format that is divided into five components. Those components are: the description of knowledge source; triggering conditions; feasible conditions for checking the validity of KSAR to be executed; scheduling variables as criteria to be used in scheduling; and the action part of the knowledge source by which it will change the state of solution.

2.6.1 Description of a knowledge source

This component consists of: a name of the knowledge source and its function; a problem that the knowledge source belongs to; area of expertise, that is a location in the blackboard which is either a domain or a control blackboard; and a level in the blackboard. If the location or the problem is quoted as 'any', then the knowledge can be used at any problem and at any location in the blackboard. An example of the component is as below:

```
**********************************************
ks_name(gf_b, 'group between former and breaker').
ks_problem(gf_b, predict_secondary).
ks_blackboard(gf_b, [domain, island]).
**********************************************
```

Fig 2.6.1: Description of a knowledge sources

The example indicates that the knowledge source 'gf_b' is used to solve the problem 'predict_secondary'. The area where the knowledge is going to be used is at the level 'island' for the domain blackboard. Knowledge source 'gf_b' is to be used to solve a structure of an amino acid sequence which is made of from: an amino acid sequence that can form a particular secondary structure('former'), a sequence of non-committed amino acid, and, an amino acid sequence that cannot form the structure('breaker').

2.6.2 Trigger Conditions

Trigger conditions describe 'at what event and state of solution the knowledge should be applied'. The condition satisfaction causes the knowledge source to participate in the problem solving. In PREDMOLL the triggering conditions consist of two parts. The first part involves conditions for its triggerring, which consists of events and/or states of solution. Once these conditions are satisfied, it performs the second part. In the second part, it assigns a unique number to the KSAR and its associated documentations. In PREDMOLL, an event can be considered as the summary of action by a task performed previously,

or, information about the knowledge source in the current problem-solving. An example for the triggering condition is as follows:

```
************************************************
ks_trigger(gf_b) :-
        /* event (information about KS) */
      not(ksar_ks(KSAR, gf_b)),
        /* state of solution */
      group_prop(S1,E1,Struct,K2,'former',active),
      group_prop(S2,E2,Struct,K2,'breaker',active),
      Space is S2-E1, Space > 1, E is S2-1,
      not (in_between(S1,E1,S2,E2,Struct)),
      /* bind  variables */
      ksar_bind([[ksar_ks, gf_b],
            [ksar_group, [1,S1,E1] ],
            [ksar_struct, [1, Struct] ],
            [ksar_group, [2,S2,E2] ],
            [ksar_struct, [2, Struct] ],
      fail.
************************************************
```

Fig.2.6.2: An example of trigger conditions

In the example, the condition for the event is that the knowledge source 'gf_b' is not currently triggered in the trigger list. The condition for the state of solution is satisfied when there is a sequence of 'former','space' and 'breaker'. The predicate 'in_between' is to determine that no groups with similar structure exist between the groups. When these conditions are satisfied, a new KSAR is created. A unique number is assigned to the KSAR and it records the current cycle i.e. the cycle when the KSAR is created. Attributes for a KSAR vary from one knowledge source to another, depending upon the usefulness of those variables in the subsequent process on the knowledge source. The predicate 'fail' is a built-in predicate in PROLOG; it instructs the system to look for other locations that satisfy the conditions. So, all locations in the sequence where the knowledge source is desired to be performed are recorded.

2.6.3 Feasible Conditions

In PREDMOLL, feasible conditions involve validating conditions of pending KSARs in the trigger list. Those KSARs were created to provide solutions and alternatives to the state of solution at the cycle when they were created. When a KSAR is executed, it changes the current state of the problem solving. As the consequence, it creates three situations for the remaining KSARs as follows:

a) The action part of the pending KSAR is no longer useful because its function has been performed by the previous KSAR, or its action is no longer desirable under the new state of solution. In this situation, the KSAR is deleted from the trigger list.

b) Conditions for its creation are still valid but it is not feasible to perform yet. Its presence in the trigger list is still needed. In this case it remains in the trigger list.

c) It satisfies feasible conditions. So the KSAR is moved to the feasible list. All KSARs in the feasible list can be executed under the current problem-solving.

An example of feasible conditions for a knowledge source is shown as below:

```
*****************************************
ks_feasible(gf_b) :-
        feasible_ksar(KSAR),
        valid_group(KSAR,1,S1,E1,Struct,'active'),!.

ks_feasible(gf_b) :-
        feasible_ksar(KSAR),
        reject_ksar(KSAR).
*********************************************
```

Fig.2.6.3a: An example of feasible conditions

In the knowledge source 'gf_b', the validity of the first group is checked. It purposes is to see whether the location has been used by another KSAR, or if it is still available to be used. When these conditions are satisfied, the KSAR is sent to the basic control. The basic control transfers it to the feasibled list. However, when conditions fail, the KSAR is removed from the trigger list. This situation is best described in an example to locate a boundary for a secondary structure as below:

23	24	25	26	27	28	29	30	31	32	33	34	35	36
G	G	C	W	A	F	S	A	I	A	T	V	E	G
b	b	-	f	f	f	-	f	f	f	-	-	-	b

Cycle 1:

KS	Group	KSAR
gb_f	[(23,24),(26,28)]	1
gf_f	[(26,28),(30,32)]	2
gf_b	[(30,32),(36,36)]	3

Trigger	feasibe	Reject
1	1	-
2	2	-
3	3	-

Chosen Action: KSAR No-2.

Cycle2:

KS	Group	KSAR
gb_f	[(23,24),(26,32)]	4
gf_b	[(26,32),(36,36)]	5

Trigger	feasibe	Reject
4	4	1
5	5	3
1		
3		

Chosen Action: KSAR No-5.

In the above example, there are fourteen amino acids from sequence number 23 to 36. An amino acid is represented by its single letter code written in a capital letter in the second line. The third line indicates the preference for each amino acid to form a particular secondary structure. The letter 'f' indicates 'former' while letter 'b' indicates 'breaker'. The task is to determine whether the hypothesis of the location is true and thence the boundary of the secondary structure.

In Cycle 1, three knowledge sources are triggered and each of them is assigned by their unique KSARs number. The number of KSARs are registered in the trigger list. Then, each KSAR in the trigger list is checked as to whether its feasible conditions are still valid. In Cycle 1, feasible conditions of all KSARs are still valid and they are transferred to the feasible list where they are evaluated by scheduler. Based on the evaluation, KSAR 2 is chosen to be executed. When KSAR 2 is executed, it changes the solution in the blackboard as follows:

- a group 'former' of sequence number 26 to 32 is created and its status is 'active';
- status of a group of sequence number 26 to 28 is changed from 'active' to 'inactive';
- status of a group of sequence number 30 to 32 from 'active' to 'inactive'.

In Cycle 2, as the result of changes from actions in Cycle 1, two knowledge sources are triggered. Their KSARs are created and recorded in trigger list along with two pending KSARs from Cycle1. As in Cycle 1, conditions of all KSARs in the trigger list are checked. Since there were changes in the status of groups (26,28) and (30,32), the conditions for KSARs 1 and 3 are no longer valid. Thus, KSAR 1 and KSAR 3 are rejected from the trigger list. KSAR 4 and KSAR 5 are then transferred for their evaluation in the feasible list, where one of KSARs is chosen to be executed.

2.6. 4 Scheduling Variables

In PREDMOLL, the scheduler evaluates the priority of each pending KSARs according to its knowledge source scheduling variables and other variables generated during the triggering process. These variables are compared with the policies adopted by the scheduler at that cycle. In PREDMOLL, the variables in scheduling variables are those given by users. An example is as follows:

ks_schedule(gf_b, credibility, 0.8).
ks_schedule(gf_b, method, statistic).

In the example, the credibility of the knowledge source is 0.8 and its method is classified under 'statistic'. These information and the value are provided by users or domain experts based on their experiences or experiments. In the scheduling process, if current interest of the problem-solving adopts policies for method 'statistics' and 'credibility' greater than 0.75, so, both policies are positive to this knowledge source.

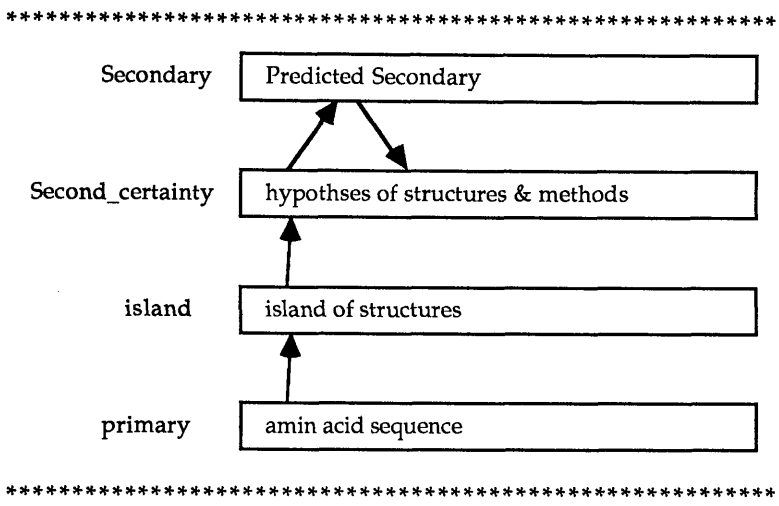
2.6.5 Action

This is the component of the knowledge source for changing the solution of the problem solving. It is performed when a KSAR belonging to a knowledge source is chosen for execution. There are two possible tasks that it will perform. The first is to improve the state of the problem-solving by changing the current state of solution. The second is to change the current control behaviour of the problem solving; this is done by summarizing the changes to the solution or control behaviour by creating an event or laying out new policies. Some actions perform either one of the tasks while others do both.

2.7 Domain Problems

Knowledge sources for domain problems are grouped into one of the various levels of abstraction which represent a broader definition of their tasks. Levels are interrelated and work together to solve a problem. However, some of knowledge sources do not belong to any particular problem and can be used for general problems. Levels for domain problems depend on the problem being solved. Levels for domain problems in HEARSAY-II are *parameter, seqment, syllable* and *word,* while, in PREDMOLL, levels for secondary structure predictions are *primary, island, second_certainty* and *secondary.*

Domain problems in PREDMOLL are relevant to domain problems in HEARSAY-II. In both, a detailed low-level description encodes, through a hierarchical organization, a recognizable higher-order pattern. Both use a combination of *data-driven* (or bottom-up processing) and *goal-driven* (or top-bottom) techniques. In the first stage, a process starts from the lowest level and provides all possible hypotheses at an associated level using *bottom-up* processing. In the second stage, it select the most credible pattern and uses the *top-bottom* processing to extend it. In PREDMOLL, in order to predict secondary structure from primary structure, all possible hypotheses from various methods are generated through the hierarchical levels *primary, island* and *second_uncertainty.* From the level *secondary_certainty* it selects the first location of the most credible hypotheses and uses this location to extend the boundary of the structure. To implement this, PREDMOLL again consults rules from lower levels to hypothesize the proposed extension of the boundary.



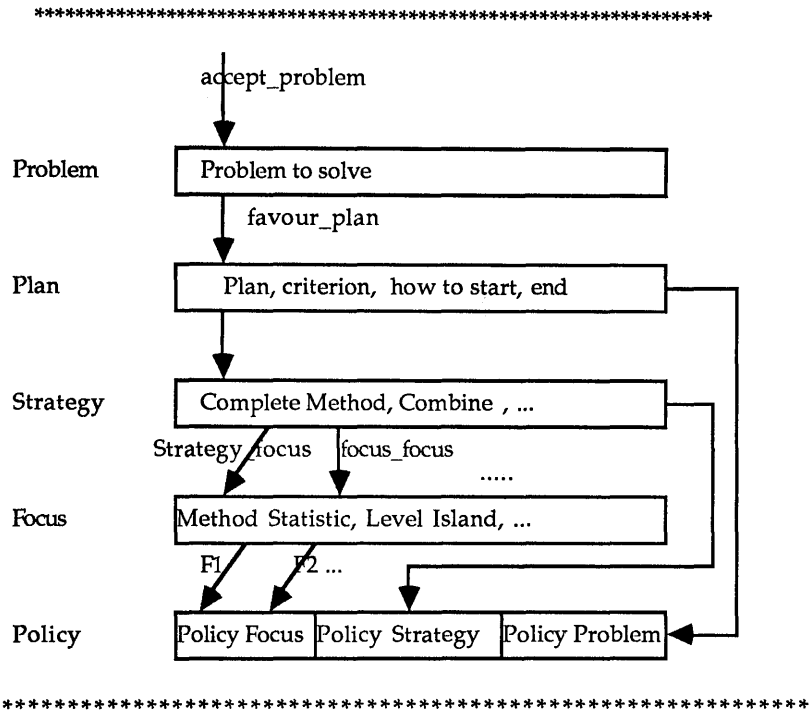
Diag. 2.7: Levels in Domain Predict Secondary

In diagram Diag. 2.7, hierarchical levels of the domain problem for predicting secondary structures are shown. Knowledge sources in level *primary* uses the information about a sequence of amino acids to hypothesize locations of secondary structure in the form of islands. Knowledge sources in level *island* use those hypothesized islands to predict the boundary of secondary structures. These predictions about the possible locations of secondary structures are sent to level *second_certainty*. As a consequence, there are various overlapping secondary structures predicted by various predictive methods in that level. This information is then used by knowledge-sources in level *second_certainty* to calculate a reliability of each location and post it to level *secondary*. Knowledge sources at level *secondary* choose the most credible location and then extend the boundary of the structure by searching relevant solution at lower level. Details of the problem-solving for domain problems are discussed in the Implementation Chapter.

2.8 Control Problems In PREDMOLL

Control problems are solved in a similar way as domain problems. Knowledge sources are grouped into several levels of

abtsraction to represent different categories of control decisions in PREDMOLL. Decisions at levels *problem, plan, strategy, focus* and *policy* determine the system's control heuristics operation at a particular point of a problem-solving. These control heuristics are used to evaluate pending KSARs in order to choose a set of KSARs for execution.



Diag. 2.8: Levels In Control Problem.

Diag.2.8 shows levels in control problems to be used in PREDMOLL. The problem to be solved is identified at level *problem* and its solution is planned at level *plan*. At this level, it plans strategies to solve the problem and set criteria for the problem completion. At level *strategy* it guides the temporary problems to be solved at level *focus*. At level *Focus*, it guides the knowledge sources at domain problems about the main interest of problem-solving and invite them to solve it.

In contrast to other systems such as OPM[Hayes_RothB85], decisions at level *policy* in PREDMOLL appear at every level in

control problems. The knowledge sources at level *policy* are created independently to interpret control decisions at any level and installs relevent policies to be adopted by the scheduling mechanism. Each policy is terminated by a generic knowledge source according to its criteria. However, by default it remains operational until the control decision which create it is terminated. So, criteria for a policy created in level *strategy,* by default, is the termination of the strategy.

In PREDMOLL, most of the communication between knowledge sources in control problems are performed by using events. So, an event is used as one of their triggering conditions as well as the result for their action. Thus, in PREDMOLL, for each of KSAR being created for knowledge sources in control problems, there are three essential attributes to bind as follows:

ksar_ks	:	name of knowledge source
ksar_event	:	event that trigger the KS
ksar_ksar	:	ksar which create the event.

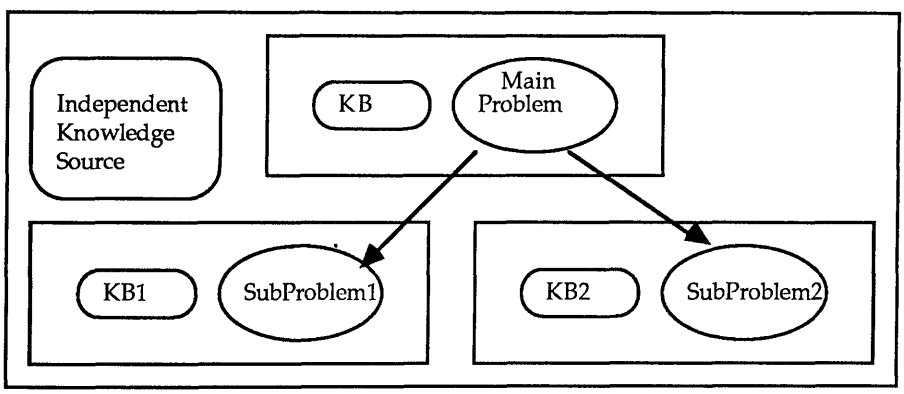
2.8.1 Level Problem

In PREDMOLL, a decision in level *Problem* is to accept the problem that the system is going to solve and record it in the blackboard. The system knows about the intention to solve a particular problem 'P' when a recent event has shown that the problem 'P' exists. The scheduler implements the intention by executing a knowledge source which accepts the problem as 'the problem' the system decides to solve. The decision is sent to lower levels in the control blackboard for further implementation. This decision is shown by a generic knowledge source 'accept problem' as follows:

```
*******************************************
trigger:          recent event is problem P
bind_var:         problem P,
                  ksar that create the event,
                  intention: to solve problem P,
                  reason: problem P has been recorded.
action:           event: accept problem P.
*******************************************
```

Fig.2.8.1a: Part of knowledge sources 'accept problem'

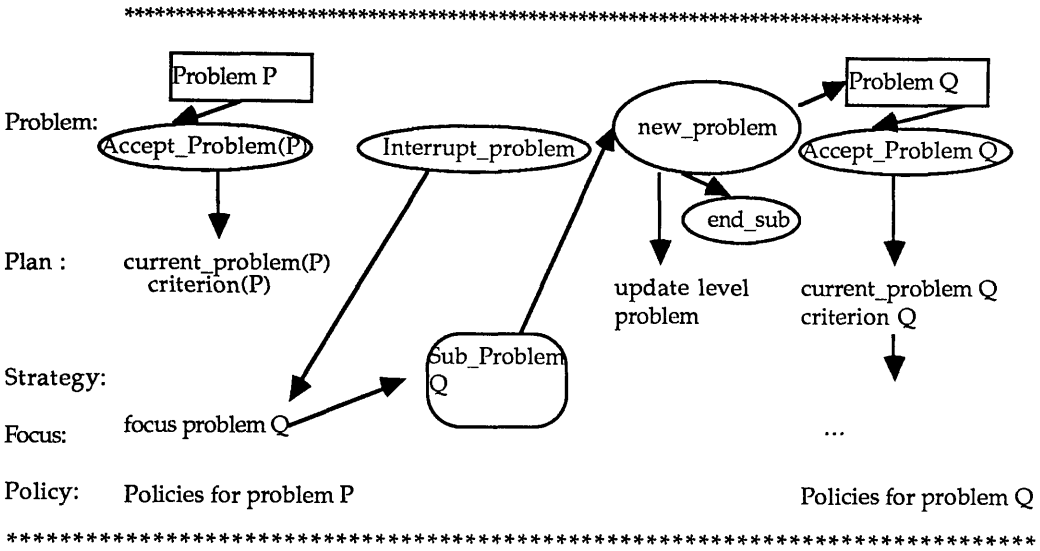
PREDMOLL in its implementation recognizes the need to coordinate and solve several sub-problems in order to solve the main problem. In Diag. 2.8.1, it shows that the 'MainProblem', 'SubProblem1' and 'SubProblem2' are independent problems and each one of them has its own blackboard and knowledge sources. Each is also entitled to use 'Independent Knowledge Sources' which are availabled to all problems. However, 'MainProblem' needs the solution of problem 'SubProblem1' and 'SubProblem2' before its problem is solved.



Diag. 2.8.1: Problem and Sub-Problem

PREDMOLL tackles the problem of coordinating sub-problems by using three generic knowledge sources. They are 'interrupt_problem', 'new_problem' and 'end-subproblem'. The process is interrupted by 'interrupt_problem' at level *focus* which indicates that there is a sub-problem to be solved. The knowledge source 'new_problem' records the new problem in the blackboard and at the same time changes all

the current foci and policies of the problem-solving. The knowledge source 'end_subproblem' is triggered to terminate the sub-problem from the problem-solving.



Diag. 2.8.1.2: Identify a new problem

In the Diag.2.8.1.2, the control at level *focus* indicates that the system has to solve problem 'Q' in order to solve problem 'P'. The system adopts the plan when the knowledge source 'interrupt_problem' interupts the current problem-solving and change the policies to solve problem 'Q' and record it in the blackboard that there is a new sub-problem 'Q'. The knowledge source 'new-problem' is triggered when there is a new problem being recorded in the blackboard and the current policy is now is favoured the action at level *problem*. The action component in knowledge source 'new-problem' changes the level of the current problem and postpones all the control strategies for problem 'P'. It records the problem 'Q' as the current problem in the problem-solving and the knowledge source 'accept_problem' accepts it. At the same time generic knowledge sources 'end_subproblem' and 'end_problem' are triggered to end 'Q' as the problem as well as the sub_problem when their respective criteria are satisfied. The system continues the process as before.

2.8.2 Level Plan

Decisions at this level are used to guide the entire problem-solving behaviours. When the system has accepted a problem as *the problem* to be solved, knowledge sources at this level install it as the current problem in the control blackboard and set its criteria. Facts such as the number of levels for domain problems, a sequence of strategies to be used, are planned in this level. In several systems, a single strategy is sufficient to solve a problem, eventhough ideally several of them are needed. PREDMOLL allows a sequence of strategies to be planned. The sequence is recorded by a predicate 'strategy_planned' which contains the sequence of strategies and criteria for their termination. A generic knowledge source 'plan_strategy' is used to implement those strategies. It is triggered when there is a sequence of strategies in the control blackboard and implements the first strategy of the sequence. The sequence of the strategies is planned ealier by a users-defined knowledge source in this level.

2.8.3 Level Strategy

Decisions at this level provide a strategy to solve a part/all of the problem determines by its knowledge source. The knowledge source for a strategy is usually provided by users. It is usually triggered by another control decisions that requires it to be implemented, or by policies that favour some of its characteristics. The first strategy decision is normally created after the problem has been planned, but it also can occur at any point in the process. It installs its strategy as the current strategy and set its criteria for termination. A generic knowledge source 'end strategy' is triggered when a new strategy is installed and terminates the strategy operation when its criteria are satisfied. A generic knowledge source 'change strategy' is triggered when the current strategy is terminated and another strategy is available in the sequence to be implemented.

A strategy in PREDMOLL provides parameters for generic control knowledge sources at level *focus* to be implemented. Knowledge sources at level *focus* generate more specific decisions in order to implement the decision at level *strategy*. In order to implement the strategy, PREDMOLL provides two alternatives. The first alternative is to specify some criteria to decide on various focus decisions, i.e. searches control knowledge sources at level *focus* which satisfy those criteria. In the second alternative, PREDMOLL provides a generic sequential plan for focus decisions. The plan is represented by a predicate called 'focus_planned' which specifies the sequential plan of the focus decisions. The format for 'focus_planned' is as follows:

focus_planned< Focus Plan, Focus Series >

The 'Focus Plan' is the creator of the focus decision 'Focus Series'. An example is shown in a user-defined control knowledge source 'strategy_second1' as follows:

```
**********************************************************
action: Let S is strategy_second1,
        install the current  startegy a strategy S,
        the criterion for the strategy S is
              method completed,
        focus plan for strategy S is :
              focus_start: method(statistic),
              focus_end: method(stereochimistry),
              focus_next: one_up,
              focus_criterion: criterion(strategy(S), method(M)),
        focus order for strategy S is :
              1 for method statistic,
              2 for method inductive,
              3 for method stereochemistry,
        criterion for strategy S with method M is:
              .....
        focus planned for method statistic is :
              focus_start: domain level primary,
              focus_end: domain level second_certainty,
              focus_next: one_up,
              focus_criterion: criterion(method(M),domain(Level)),
```

```
focus order for method statistic:
        1 for domain level primary,
        2 for domain level island,
        3 for domain level second_certainty,
    .....
    .....
**********************************************************
```

Fig.2.8.3: Part of the knowledge source 'strategy_second1'

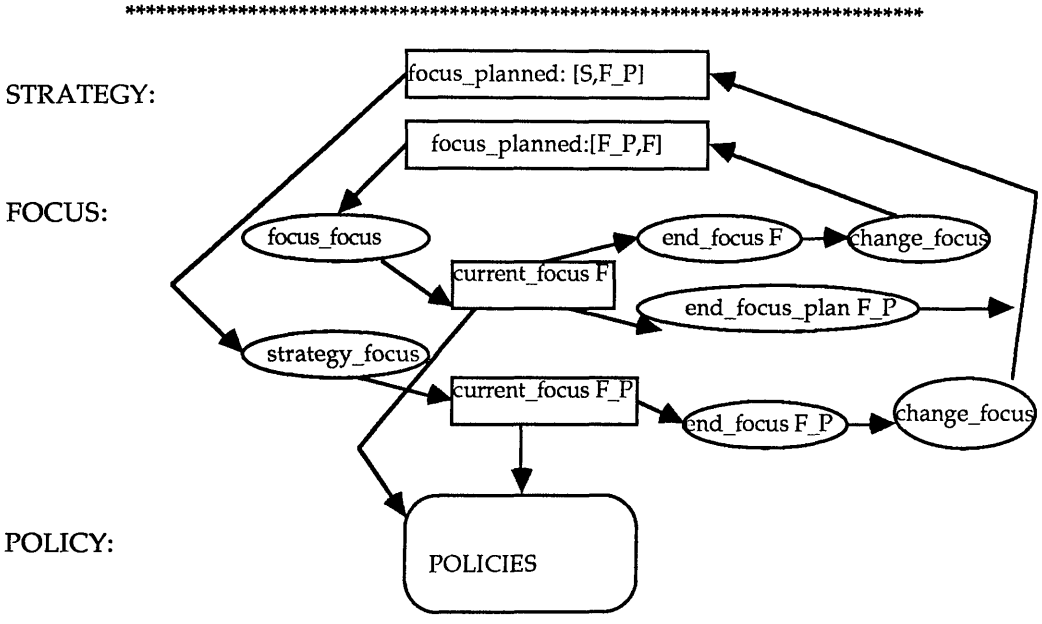
As shown in the control knowledge source 'strategy_second1', there are four predicates specific to 'Focus Series' to implement a strategy as a series of more specific decisions at level *focus*. Predicate 'focus_first' describes as the first focus decision while 'focus_end' is the last focus decision that the system decide to perform. Predicate 'focus_next' suggests a new focus when the current focus is completed. Finally, a predicate 'focus_criterion' is specify criteria for the current focus decision to be completed. The order of the focus decision sequence is laid out in a predicate 'focus_order'. So, in the example in Fig.2.8.3, the general sequential plans for strategy 'strategy_second1' is to perform decisions for the method 'statistic', followed by the method 'inductive' and finally the method 'stereochemistry'.

2.8.4 Level Focus

Decisions at level *focus* usually implement decisions at level *strategy*. However, PREDMOLL allows other *focus* decisions which operate independently from decisions at level *strategy*. Decisions at this level are explicitly temporary and operate during restricted local problem-solving. In PREDMOLL, several complementary and competing focus decisions may operate simultaneously. They establish a local problem-solving and execute KSARs with particular attributes and values. As a consequence, they are used to rate KSARs and influence scheduling and triggering decisions.

Several generic control knowledge sources operate in this level. There are 'strategy_focus', 'focus_focus', 'change_focus', 'end_focus'

and 'end_focus_plan'. The control knowledge source 'strategy_focus' performs all decisions laid by the strategy in 'focus_planned'. In the example in Fig.2.8.3, it plans the order of methods as accordance to the order of the predicate 'focus_order'. In level *focus*, PREDMOLL installs the current focus and its criteria. In the example, it shows that for each of the method, there is another 'focus_planned' involved. PREDMOLL tackles this problem by using a generic control knowledge source 'focus_focus' to provide another current focus in the problem solving. As the consequence, PREDMOLL provides facilities for the hierarchical focus in the problem solving.



Diag.2.8.4 Implementing a startegy at level *focus*

When a focus decision is installed due to a hierarchical focus , a generic control knowledge source 'end_focus_plan' is triggered. Its criteria is for all focus decisions specified by the 'Focus Plan' have been completed. For each of a current focus decision being installed, a generic control knowledge source 'end_focus' is created and its criteria is for the focus decision to be completed. When a current focus is completed, a new current focus must be installed. In order to change the current focus , a generic control knowledge source, 'change_focus'

is executed. It changes the current focus, as planned by 'focus_planned' and 'focus_order', to a new current focus decision.

2.8.5 Level Policy

Control decisions at this level contrast sharply with decisions at other levels. The decisions are implemented at any control level. It involves in any control decisions and provides the policies to be adopted by the problem-solving. Hence, it directly determines its control behaviour. When a problem is accepted to be 'the problem' to be solved, a policy decision installs a policy to favour this problem. Similar situation is adopted when a strategy decision or a focus decision is installed. These are done by several generic knowledge sources such as 'policy_problem', 'policy_plan', 'policy_strategy' and 'policy_focus'. Policy decisions remain operational until its criteria are satisfied. In that case, by default, policies created at level *problem* remain operational until the problem is completed. For each policy decision being installed, a generic knowledge source *end_policy* is triggered and its action removes the all policies laid by the decision.

Decisions at this level influence the scheduling and the triggering mechanism. For the scheduling mechanism, decisions at this level influence the rating of pending KSARs as long as pending KSARs contain attributes and values described in policy decisions. For triggering mechanism, it consults knowledge sources at the domain blackboard when policy decisions contain their attributes. In this case, if decisions at this level favour the method 'statistic' and at the domain level 'island', the triggering mechanism triggers knowledge sources which have those attributes only. Other domain knowledge sources are not consulted.

```
**********************************************
trigger:
            event: focus method(statistic) is installed.
action:
            schedule_policy: method(statistic),
            criteria:

focus method(statistic) is completed.
**********************************************
```

Diag.2.8.5: Interpreting Control Decisions

Diag.2.8.5 shows one of responses at level *policy* as a result of control decisions at level *focus*. When a control decision at level *focus* installed 'method(statistic)' as the system local problem-solving, one of knowledge sources at level *Policy* installs a scheduling policy to favour it. Others may install a trigger policy to trigger knowledge sources with that attributes only. When the focus decision is terminated, the generic knowledge source 'end_policy' knows it by its policy criteria.

.

2.9 Basic Controls

Three knowledge sources are used to provide a basic control in PREDMOLL: 'update_agenda', 'select_ksar' and 'execute_ksar'. These knowledge sources are not involved in the scheduling process but when they are triggered by the system, they are simply being executed.

2.9.1 Update_Agenda

The purpose of knowledge source *update_agenda* is to provide the trigger list with all knowledge sources that satisfy a new state of problem solving. It is triggered when an event 'change_status' occurs, indicates that the chosen KSAR has recently been executed. The action part of the 'update_agenda' increases the value of the current cycle with one, updates the trigger list, creates the feasible list and calculate a rating for each KSAR in feasible list. The action part of knowledge source 'update agenda' is as follows:

```
**************************************************
action :-
  increase the cycle by one,
  remove previously executed KSAR
          from feasible list,
   move all  KSARs in feasible list to
          trigger list,
 create a new  trigger list for
         all KSs that satisfy the new event and
        new solution states,
  for all KSARs in trigger_list:
        check their feasible conditions and
              if satisfy put the KSAR in feasible list,
               if the feasible condition is fail then the KSAR
         remain in the trigger list,
     if  the feasible conditions are no longer
         valid remove the KSAR from trigger list,
  for each KSARs in feasible list:
    compute their priority based on the  current
         policies.
**************************************************
```

Fig.2.9.1: Part of knowledge sources 'update agenda'

It is important to distinguish between the trigger list and the feasible list. The trigger list contains all KSARs which satisfy the trigger conditions of their knowledge source. The feasible list is a list for all KSARs previously in trigger list that satisfy their feasible conditions of their knowledge source. Each KSAR in the feasible list is rated for its scheduling priority, based on the current scheduling policies of the problem-solving.

The system architecture for PREDMOLL allows only useful knowledge sources being searched to be triggered. This is determined by the trigger policy in the control blackboard. This reduces the effort of searching for useful knowledge sources under the current interest in the problem solving. Under the present scheme, at any stage of problem solving, it consults all knowledge sources on control blackboard and knowledge sources at level 'any' in domain blackboard for triggering. For knowledge sources at domain blackboard, they are

consulted when there is a trigger_policy in the control blackboard favour them.

2.9.2 Select KSAR

In this knowledge source, it is triggered by an event 'new_agenda'. The action part of this knowledge source is to select the highest rating KSAR. If several KSARs have similar values, it chooses the first one. PREDMOLL allows another stage of selecting KSARs by second stage scheduling. In PREDMOLL, this is an option, of which a generic knowledge source 'select_ksar' performs two tasks:

a) Select the highest rating which does not contradict with rules in second level of scheduler;
b) Search KSARs that are executable in this cycle without affecting the outcome of the problem solving.

As a consequence, PREDMOLL allows a set of KSARs to be executed in one cycle. However, the second level is an option in PREDMOLL and will be discussed under section(2.10).

2.9.3 Execute KSAR

The task of this knowledge source is to execute a set of KSARs which has been scheduled by the knowledge source 'select_ksar'. When a KSAR of the set has been executed, the 'select_ksar' records this event as 'change status'. This event triggers knowledge source 'update_agenda' and the process is started again.

2.10 Scheduling The KSARs

This is an important component in the control strategy. In order to identify the next action to be executed, all executable KSARs have to

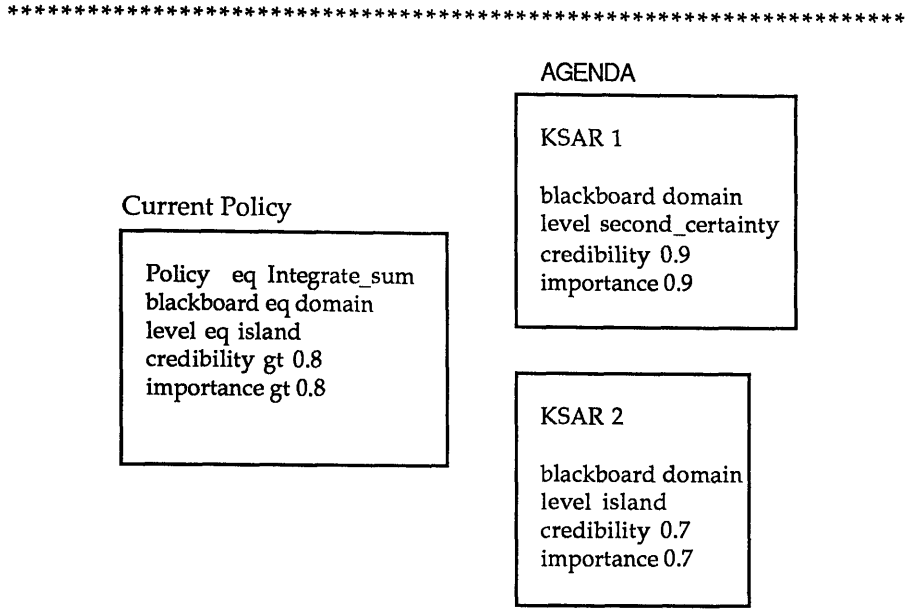
be rated. In PREDMOLL, competitions among KSARs to be scheduled are essential because several KSARs are competing for the same locations in the sequence. Once a KSAR is executed, feasible conditions for several KSARs are no longer valid, thus, they are removed from the trigger list. Each of KSARs in the feasible list is rated against the current scheduling policies.

Current scheduling policies are stored in predicate 'scheduling_policy'. Its variables are scheduling_criteria, operator and a value of the scheduling_ criteria. The highest rank of scheduling_criteria is called 'combine_policy'. The 'combine_policy' determines the integration of all scheduling criteria. At the moment, two types of 'combine_policy's are provided as alternative as follow:

integrate_sum : the sum of the ratings for all criteria;
integrate_wight_sum : the sum of rating with their weight
for all criteria.

2.10.1 Scheduling based on numerical rating

There are two stages of scheduling a process in PREDMOLL. The first stage is to choose the next action based on the highest rating of pending KSARs. The calculation is shown in the following example:



Diag 2.10.1: Scheduling Priority

In PREDMOLL, if a characteristic of a KSAR satisfies a scheduling_criteria in the current policy, then the value 100 is added to its rating. In the example rating values for KSAR1 and KSAR2 are calculated as shown in Fig2.10.1(). Fig.2.10.1(b) shows that if the combine_policy is integrate_sum, then KSAR 1 is chosen. However, if the policy is integrate_weight_sum and a weight for an action level is 3 while the rest are 1, then KSAR 2 is chosen. In this case, different types of policy produce a different priority for the competing KSARs. This element is important to provide a flexible and dynamic control strategy for an intelligent system.

policy: integrate_sum

	blackboard	level	credibility	importance		TOTAL
KSAR1:	100	+ 0	+100+(0.9-0.8)*10	+100+(0.9-0.8)*10	=	302
KSAR2:	100	+ 100	+0	+0	=	200

Fig:2.10.1(b1): An example to calculate a rating value

policy: integrate weight sum

	blackboard level	credibility	importance	TOTAL
KSAR1	100 +0	+101	+101	= 302
KSAR2:	100 +100*3	+ 0	+0	= 400

Fig:2.10.1(b2): An example to calculate a rating value

2.10.2 Second stage of scheduling

The second stage is optional under the current implementation of PREDMOLL. The problem-solving is informed by a control knowledge source if this type of scheduling policy is to be adopted. In the second stage, the policy does not involve in calculating the rating value for competing KSARs but is based on some features determined by the policy. Under the current implementation, two policies are installed. They are 'unconditional' and 'independent'. Their function are as follows:

unconditional - the scheduler obeys the policy unconditionally. If there is a KSAR which has highest numerical rating value but one of its attribute is unconditionally lower than one of inferior rating KSARs, then it cannot be chosen to be executed.

independent - a knowledge source which is categorized as 'independent' can be executed once its feasible conditions are satisfied. It is executed along with the chosen action at that cycle.

2.10.2.1 Policy 'unconditional'

The policy 'unconditional' is introduced to avoid any difficulty in providing numerical values for planning policies. An example is

shown as follows:

```
**********************************************
policy2 unconditional structure:
    1.  alpha       gt    beta strand
    2.  beta turn   lt    beta strand
    3.  beta turn   gt    gamma
**********************************************
```

Fig. 2.10.2.1: an example of policy unconditional

In the above example, the system would like to implement a policy where a KSAR of structure alpha to be executed ahead of KSAR of structure beta strand, irrespective of their numerical rating values. An easier way to do it under the numerical system (stage one) is by assigning numerical values for each of the structures and allow a very high weight for policy structure. During the scheduling process, it is hoping that the intended task will produce the highest rating. However, the numerical technique is implicit in implementing the policy. PREDMOLL solves the problem directly using two steps: solves the order of the structure and they become alpha > beta strand > beta turn > gamma; the scheduling mechanism obeys the order of structure if one of the attribute of competing KSARs contains it. An example of competing KSARs are as follow:

KSAR: 1	Rating: 300	Structure: beta turn
KSAR: 2	Rating: 250	Structure: alpha

In the example, KSAR 1 has the highest rating among competing KSARs. However, since the scheduling policy contains the unconditional policy for structure, it ordered those structures and found that KSARs of beta turn should not be scheduled ahead of KSARs of alpha. So, KSAR 1 is not chosen as imposed by the policy. The scheduler is then checked the next highest rating of KSAR. If the next KSAR does not have structure as one of its attribute, then it will

be scheduled. Otherwise, a similar process is repeated until one of KSARs is selected. If there is more than one 'unconditional' policies and during the competition there are conflicting decisions, under the current implementation of PREDMOLL it is resolved by the value of the numerical rating.

2.10.2.2 Policy 'independent'

This is a new concept in blackboard architecture to be introduced in PREDMOLL. Under the current concept of blackboard architecture, it allows only one task to be scheduled and executed at each cycle. However, in PREDMOLL, more than one tasks are allowed to be scheduled and executed at any cycle. The rationale for the policy is as follows:

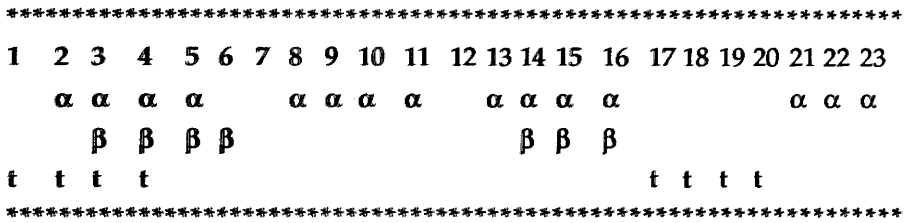


Fig: 2.10.2.2 : Overlapping and 'isolate'

In Fig.2.10.2.2, there are several overlapping locations where their secondary structures conflict. Beside that, there are locations which are not overlapping and they are to be called 'isolate' [at locations 8-11,17-20, 21-23]. Locations and their structure are as follows:

```
*****************************************************
KSAR: 1          Location: 2-5      Structure: alpha
KSAR: 2          Location: 3-6      Structure: beta strand
KSAR: 3          Location: 1-4      Structure: beta turn
KSAR: 4          Location: 8-11     Structure: alpha
--------
--------
*****************************************************
```

Fig.2.10.2.2(a): An example for the policy 'independent

In the example, there is competition between KSAR 1,2 and 3 to be selected. Once one of them is executed, the remaining tasks are no longer useful and are rejected. However, locations under 'isolate' are not involved in any competition and eventually should be executed. Their actions do not affect other locations. So, under other blackboard architectures, three cycles are required to execute the 'isolate' locations in Fig.2.10.2.2. This is time consuming as for each cycle it requires the basic control to consult each rules for the satisfaction of their trigger conditions.

Under the current implementation of PREDMOLL, all knowledge sources classified under 'independent'(applied for the above 'isolate' case) are scheduled once their feasible conditions are satisfied. They are executed along with the chosen KSAR at that cycle. The technique reduces the processing time because their actions do not affect the problem-solving process. In PREDMOLL, most knowledge sources at level *policy* are classified under 'independent'. This is because their actions are required for the implementation of scheduling or triggering policies.

2.11 Summary of Features of PREDMOLL

2.11.1 Coordination of independent problems

PREDMOLL defines a problem as consisting of several independent sub-problems. In order to solve the main problem, PREDMOLL has to solve all related sub-problems. Each sub-problem is a problem by itself and has its own blackboards. PREDMOLL determines which sub-problems, and at what point of the problem-solving, that a sub-problem has to be performed in order to solve the main problem.

2.11.2 Problems are solved by a uniform control behaviours

PREDMOLL solves control problems for various independent sub-problems by uniform control behaviours. It divides control problems into decisions at level *problem, plan, strategy, focus* and *policy*. Decisions at level *plan* are implemented at level *strategy* while decisions at level *strategy* are implemented at level *focus*. Decisions at level *policy* interpret control decisions at higher levels for their triggering and scheduling policies. These decisions are uniformly used in problem and sub-problems during the problem-solving process.

2.11.3 Domain and control problems are explicitly represented

There is a clear distinction between control problems and domain problems. Control problems are defined as a real-time planning problems whose solution determines what actions the problem-solving should perform at each point in the problem-solving. Domain problems are solved by creating and modifying decisions on the domain blackboard. Thus, PREDMOLL explicitly specifies domain and control knowledge sources to solve their respective problems on domain and control blackboards.

2.11.4 Knowledge sources are independent

In PREDMOLL, knowledge sources are independent and each is considered as an expert on its problem. They communicate through the events and changes in the blackboard. This modularity provides an advantage to the system as it can delete or add knowledge sources without any reorganization.

2.11.5 Single basic control loop

PREDMOLL maintains the three basic steps in its control. They

are represented by knowledge sources 'update_agenda', 'select_ksar' and 'execute_ksar'. They manage triggering, scheduling and executing of all problem-solving actions for both domain and control actions in a uniform and single control loop.

2.11.6 Selection and execution of a set of KSARs

In contrast to many other blackboard architecture systems, knowledge sources 'update_agenda', 'select_ksar' and 'execute_ksar' are intended to schedule and execute a set of KSARs at any cycle of problem-solving.

2.11.7 Modification To The Representation of Knowledge and Behaviour

As all knowledge sources and all solution elements generated for control and domain problems are recorded in the blackboard, they are represented as data structure. This information is available for modification in the continuing problem-solving process.

2.11.8 Adaptation to dynamic problem-solving

The three basic control knowledge sources have no specific control knowledge of their own. They are executed whenever their conditions are satisfied. The behaviour of the overall problem-solving is determined entirely by dynamic interactions between the domain and control knowledge sources. Knowledge sources from control problems guides the current interest of problem solving while knowledge sources from domain problems solve the temporary problem.

2.11.9 Policies are interpreted by knowledge sources at level *Policy*

Control decisions at any level are interpreted by independent

knowledge sources at level *Policy* to be used for scheduling and triggering mechanism. For each policy decision, criteria for its termination are determined. Thus, control decisions at any level influence the scheduling and triggering policy during the problem-solving process.

2.11.10 Control decisions influence triggering mechanisms

Most of the policies are involved in influencing the scheduling mechanism only. In PREDMOLL, however, control decisions are also used to influence the triggering mechanism.

2.11.11 The actions are selected by a uniform mechanism

Knowledge sources are represented as condition-actions which generate and modify the solution elements under the management of the scheduling mechanism. The scheduling mechanism consults all the policies in the control blackboard in evaluating the competing KSARs.

2.11.12 Two stages of scheduling mechanism

The scheduler is allowed to schedule more than one KSAR at any cycle. The scheduling mechanism has two stages. The first stage is to compute the numerical rating for each KSAR based on the policies adopted by current problem solving. KSARs are ranked according to their value of the numerical rating. In the second stage, the policies "unconditional" and "independent" are applied. A chosen KSAR has to be satisfied with the policy "unconditional" before it can be scheduled. On the other hand, remaining KSARs which are classified under "independent" are also scheduled to be executed along with the chosen KSAR.

2.12 Comparison with other AI systems

In this section, the features in PREDMOLL are compared with HEARSAY-II and OPM. HEARSAY-II uses a sophisticated scheduler while OPM uses a control blackboard architecture.

Feature	HEARSAY-II	OPM	PREDMOLL
1. Coordinate various independent problems	no	no	yes
2. Problems are solved by a uniform control behaviours	no	no	yes
3. Domain and control problems are explicitly represented	no	yes	yes
4. Knowledge sources are independent	yes	yes	yes
5. Single basic control loop	yes	yes	yes
6. Selects and execute a set of KSARs	no	no	yes
7. Modifies representation of their own knowledge and behaviour.	yes	yes	yes
8. Adapts to dynamic problem-solving	no	yes	yes
9. Policies are interpreted by KSs at level Policy	no	no	yes
10. Control decisions influence triggering mechanism	no	no	yes
11. Action is selected by a uniform mechanism	yes	yes	yes
12. Two stages of scheduling mechanism.	no	no	yes

Fig. 2.12: Comparison of various blackboard architecture systems

2.12.1 HEARSAY-II

The HEARSAY speech understanding system was developed as part of speech understanding research project and has been one of the most influential of all AI innovations over the years. The main

contribution of this project is the idea that independent knowledge sources cooperatively solve a problem by posting hypotheses on a global blackboard data structure.

The HEARSAY research project went through two stages of development. The first stage, called HEARSAY-I, used three independent knowledge sources to represent knowledge about *acoustics* and *phonetics*, *syntax* and *semantics* of the domain. The second stage is called HEARSAY-II and includes the development of many specialized knowledge sources. The blackboard is divided into levels of hierarchical units such as *segments*, *syllables*, *word*, *word-sequence* and *phrase*. Hypotheses about these sentence units are posted at the appropriate level, along with the time frame that indicates when the unit is hypothesized to occur in the utterance.

HEARSAY-II incorperates two stages of processing. In stage 1, it processes bottom-up by exhaustively scheduling all knowledge sources that produce hypotheses hierarchically. In stage 2, top-down processing involves predicting, testing, and concatenating multiple-word sequences, one or more of which will eventually account for all of the word spoken. During stage 2, control strategy operates oppurtunistically by scheduling knowledge sources that have the highest ratings on criteria as follow: efficiency, reliability, credibility of triggering, and importance.

In Fig.2.12, it is shown that the architecture in HEARSAY-II does have features for independency of knowledge sources, single basic loop, ability to modify representation of their own knowledge and behaviour, and uniform mechanism by using the scheduler to select a KSAR. However, the architecture is not capable of handling independent sub-problems, so it does not have the first two features as in Fig.2.12. However, it does not explicitly represent knowledge sources to solve control problems because its strategy is implicit. This is because the scheduler, the only means of control, has no

knowledge of it and does not explicitly implement it. In stage 2, it has only the knowledge of the oppurtunistic scheduling criteria to choose KSARs, but regardless of any point and strategic phase in the problem-solving, it will choose KSAR that has the highest rating for the criteria. In feature 6, it selects only one KSAR at any cycle. In feature 8, it cannot adapt to dynamic problem-solving because the criteria are fixed throughout the whole process. It does not have knowledge sources to implement its scheduling and triggering policies. Finally, as in any blackboard architecture systems, there is one stage of scheduler to compute the rating for competing KSARs.

2.12.2 OPM

OPM was developed using control blackboard architecture and is claimed to achieve a very high degree of intelligence[HayesRothB85] . It was developed to tackle the problem of 'multi-tasking planning'. The architecture is as an extension of HEARSAY-II in two aspects. In the first aspect,the domain and control problems are explicitly represented. The control problem is divided into hierarchical levels and the system explicitly solves the control problem by record it in control blackboard. In the second aspect, the architecture adapts to dynamic problem-solving situation. This is made possible when KSARs are scheduled against the focus and policies of the problem solving at that particular situation. So, in Fig. 2.12. this architecture has the features 3 and 8 as addition to features 4,5,7,11.

The architecture of the system however, still cannot deal with independent sub-problems in the system. It only solves a problem within one blackboard and cannot coordinate sub-problems independently. So, the architecture does not have features 1 as well as features 2. In feature 6, it can only schedule one KSAR at any cycle and there is one stage scheduling mechanism only. Each policy used by the scheduler is decided as part of knowledge sources at higher control levels and the policy is not from an independent knowledge source as

shown by feature 9. None of the policy is intended other than for scheduling mechanism, thus, it does not have feature 10, that is the influence over triggering mechanism of knowledge sources.

2.12.3 PREDMOLL

The architecture in PREDMOLL is an extension of previous architectures. It has the capabilities of coordinating independent problems and treats them as sub-problems. When solving a problem and sub-problems, PREDMOLL uses a uniform control behaviour to solve the control problems.

There are several other differences between architecture in PREDMOLL and in OPM as shown in Fig.2.12. The two stage scheduler in PREDMOLL requires the facility for PREDMOLL to schedule and execute a set of KSARs at one cycle. These facilities along with other several features in the second stage of scheduling mechanism, provide several advantages that are not availabled in OPM. In fact, the example that has been given earlier for the second stage scheduling cannot be explicitly implemented in OPM. Furthermore, control decisions at any level in PREDMOLL are used to influence not only scheduling policies but triggering policies as well. This is in contrast to OPM, where not only it does not influence triggering policies, but it scheduling policies are only started at level *focus* only.

One of other major improvements in PREDMOLL compared to OPM is the way knowledge sources at level *policy* are used to implement their policies. Knowledge sources at level policy response to any decision for control problems by implementing it decisions into policies. They determine the criteria for the policies to be terminated. By default, the policies are terminated once the decision which created it is terminated. This is not availabled in OPM although it is claimed to adapt to a dynamic problem-solving. Policies in OPM are implemented as part of knowledge sources at higher levels. Although it is used to

influence scheduling, but it does not tell the reasons for its employment and does not have criteria for its policy termination. It is terminated by other higher knowledge source especially 'terminate_focus' which simply replace them with other policies.

In addition to the differences and improvements, PREDMOLL also differs from OPM in several other aspects. This differences are follows:

a) PREDMOLL does not allow any KSARs whose conditions are not satisfied to be considered for execution. In this case, the process of screening KSARs is done when those KSARs are transferred from the trigger list to the feasible list. In OPM, all KSARs are rated and the highest is chosen for execution. If the chosen KSAR is not feasible, the second highest KSAR is scheduled. So, in OPM there is no differentiation of the function of trigger list and feasible list, whereas in PREDMOLL it does.

b) PREDMOLL allows local problems to be focussed hierarchically by using the generic control knowledge source *focus-focus*. PREDMOLL knows how to plan this control behaviour and how to terminate it smoothly. OPM on the other hand does not have this facility.

2.13 Conclusions

PREDMOLL has inherited the advantages of previous blackboard architecture systems. It offers a flexible and modular environment for complex problem-solving. It allows users to introduce, modify, and remove control knowledge sources independently. It places all parameters governing system behaviour under a system control. It does not simply solve the problems, insteads, it knows how to solve it and plan control strategies. It plans when to implement, interrupt,

resume, or terminate strategies. It determines which policies apply in current problem-solving and which ones should be adopted, and it determines how to integrate various policies in order to schedule pending problem-solving actions.

PREDMOLL provides several improvements for blackboard architectures which can be seen from its comparison with other systems, notably OPM. These improvements provide significant contributions to the intelligence of the control strategy in problem-solving process. However, despites its advantages, PREDMOLL also inherits the disadvantages of blackboard architecture, i.e. high overheads. It records all rules and solutions in the blackboard and these processes need a lot of computer memory. In the Implementation Chapter, this problem will be discussed and some ad-hoc measures taken.

CHAPTER THREE
Uncertainty Method

3.1 Introduction

Prediction has always been associated with uncertainty. This is because the knowledge to be used in prediction is not always complete. For proteins, it is believed that there exists a system which governs the position of atoms so that the protein can function accordingly. Perhaps because of its complexity, a satisfactory theory to explain this has not yet developed.

In Chapter One, the need for an uncertainty technique to be used in protein structure prediction was highlighted. This is because several methods which are based on physical and chemical patterns of amino acid sequence have tried unsuccessfully to show that certainty theories for protein structure exist. The methods avoided the use of any uncertainty values and in particular a threshold value in making decisions about protein structure.

This Chapter explores the techniques of uncertainty to be used in protein structure prediction. In this investigation, discussions are directed to the following areas:

a) Approaches to uncertainty problems in protein structure prediction;
b) Reviews on the theory of the Bayesian method and criticisms on its application;
c) Application of uncertainty techniques for the analysis of amino acid sequences and propagation due to a combination of methods;
d) Advantages and limitations of techniques.

3.2 Approaches to the problem

The problem of protein structure prediction is to determine which locations in the sequence form a particular structure. For secondary structure prediction, as discussed in Chapter One, the problem can be approached in a series of steps:

3.2.1 Sequence of amino acids

3.2.1.1 Exact Similarity

The first idea to be used in prediction is the application of simple statistical techniques. Firstly, this is done by observing sequences of amino acids that form a particular secondary structure. Each sequence of amino acids that form secondary structure is stored. The prediction is performed and comparison is made in terms of exact similarity of all the amino acids in the sequence. Although the idea seems to be simple, the implementation is not practical. As the number of proteins grows, the sequences to be stored become very large. Furthermore, many workers have found that a similar sequence of amino acids can adopt different secondary structures. An illustration is as follows:

```
*********************************************************
S1: α    α    α    α    α    α    α    α    α    α
A:  Y1   Y2   Y3   X1   X2   X3   Z1   Z2   Z3   Z4

B:  M1   M2   M3   X1   X2   X3   L1   L2   L3   L4
S2:                β    β    β
*********************************************************
```

Fig.3.2.1.1: Different structures for similar sequence

As can be seen in the example, the sequence [X1,X2,X3] adopts an alpha helical secondary structure in A, while the similar sequence in B adopts a beta strand conformation. This is because of the effect of the neighbouring amino acid sequence and long distance interactions between structures. To avoid the problem, an observation is carried

out on the number of sequences [X1,X2,X3] that adopt beta strands and the number which do not. The probability is calculated as the ratio of the number of positive and negative accurance of the sequence to adopt the structure out of the total sequence observed. So the rules can be formulated as below:

$$\text{probability}([\beta,\beta,\beta\ ] \mid [X1,X2,X3]) = p_{positive}$$

The probability value can be further improved if other information such as the amino acids surrounding it are added.

3.2.1.2 Transformations

This is an extension from the method in section(3.2.2.1). In this method, some of the amino acids in the sequence are transformed into their physical and chemical properties. It obviously reduces the number of sequences to be stored. The method is employed in principle by King, Lim, Cohen as discussed earlier in Chapter One. However, none of the methods attach uncertainty values to their rules. In this work, the uncertainty values for these methods are investigated. This is done by comparing its prediction with the actual locations of structure from the protein database.

3.2.2 An amino acid as uncertainty problem

3.2.2.1 Single Model

This method is in contrast to the previous method, i.e. using an amino acid as the method for prediction. Instead of storing sequences of amino acids as in section(3.2.1), this method stores the uncertainty value for the formation of a secondary structure by a single amino acid. The uncertainty value is derived from (the calculation of) its probability value. The probability of amino acid A_i given a secondary structure S_j can be computed as follows:

$$P(A_i \mid S_j) = \frac{\text{Total occurrences of amino acid } A_i \text{ in structure } S_j}{\text{Total occurrences of amino acid in structure } S_j}$$

Since there are 20 types of amino acid in proteins and four types of secondary structure (alpha helix, beta strand, beta turn and coil) to be used in prediction, the following equations are satisfied:

$$\sum_{i=1}^{20} P(A_i \mid S_j) = 1 \qquad \text{and} \qquad \sum_{j=1}^{4} P(S_j \mid A_i) = 1$$

The equations are true because no amino acid adopts more than one secondary structure and the secondary structures concerned are mutually exclusive:

$$p(S_j, S_k \mid A_i) = 0$$

If the prediction is based on this approach is adopted and the decision is taken based on its highest probability value, the hypothetical outcome is shown below:

Amino:	gly	met	asn	asp	ala	leu	tyr.....
Structure:	t	α	β	α	α	β	t.....

Fig 3.2.2.1: Single Model

The hypothetical outcome using this method indicates that the straight forward method produces undesirable results and the boundary of the structure cannot be determined. An improvement to this method is discussed in the following section.

3.2.2.2 Creating Islands

This method borrows an idea applied in techniques for analysing natural languages. To search along the amino acid sequence in order to determine the location of a secondary structure is obviously a

tremendous task. In order to reduce the complexity of the problem, a procedure is developed to create 'islands' of potential locations for the structures and their boundary. The potential locations(*former*) and *breaker* are defined below:

$$\text{propensity}(S_j \mid A_i) = P(A_i \mid S_j) / P(A_i \mid \text{total protein})$$

$$former_j: \quad \text{propensity}(S_j \mid A_i) > 1$$

$$breaker_j: \quad \text{propensity}(S_j \mid A_i) < p_{breaker}$$

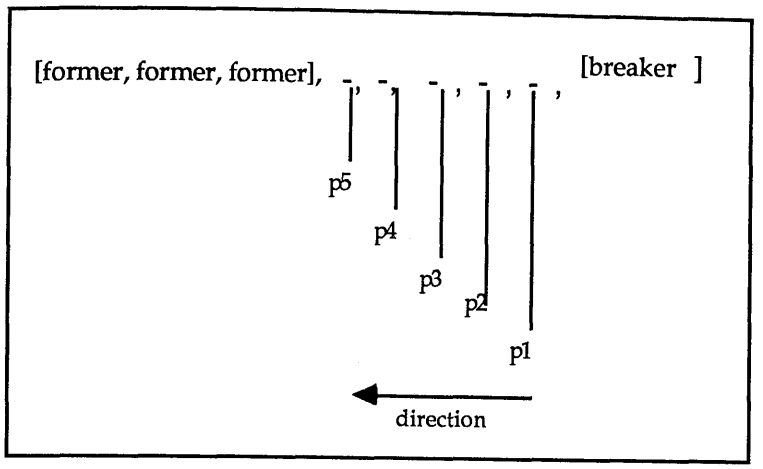
The parameter 'propensity(S | A)' is defined as "the inclination of the location to adopt secondary structure S if the amino acid at that location is A". The calculation of the propensity is based on the ratio of probability($A_i | S_j$) to the probability of A_i for the protein. So, if the propensity value is greater than one, then there is a strong inclination (above average) that it will adopt secondary structure S_j. Conversely, a poor propensity value give rise to a breaker, a location that possibly not to be part of the secondary structure S_j.

In this work, 'island' is defined as at least made up from three contiguous *formers*, while a *breaker* is sufficient to terminate the boundary.

minimum 'island': [$former_j$, $former_j$, $former_j$]

breaker: [$breaker_j$]

A 'real breaker' which determines the boundary is hypothesized in the location separating a *former* and a *breaker*. In order to determine the 'real breaker', each location in the interval is tested. In this case, a criterion based on uncertainty technique is developed. If the criterion is greater than a pre-determined (threshold) value, then the sequence is predicted to adopt a certain secondary structure. Otherwise, the hypothesis is rejected. The situation is illustrated in the diagram below:



Diag.3.2.2.2: Searching for boundary

In the diagram, the system assesses the furthest amino acid as its first strategy and then continues backward as shown by the 'direction'. At each location, if the probability is higher than the threshold value, the process stops and the most probable boundary for the structure is determined. So, the problem to be solved in this method is that of calculating the probability for a sequence of amino acids and the testing of hypothesis. This problem is represented as follows:

$$\text{probability}(S_j \mid [A_1, A_2, A_3, \ldots, A_n]) = p \quad ?$$

$$p >= P_{threshold}$$

There is no simple technique to calculate the probability for uncertainty with arbitrary length. The simple statistical technique available is similar to the technique in section (3.2.1.1) where all of the amino acid sequences have to be stored.

Fortunately, with the progress of uncertainty techniques in expert systems, various alternatives are possible. Modifications of the Bayesian method are relevant to the technique of calculating uncertainty. The Bayesian method and its application in calculating the uncertainty for a sequence of amino acids are discussed in the next

sections.

3.2.2.3 Combination of methods

This is another way to improve the performance of prediction which is discussed in Chapter One. The technique combines several methods of prediction and takes advantage of the outcomes of the combination as follows: reliability values generated by the overlaps between the different methods; and additional locations predicted by additional methods. This is shown by the following example:

Location:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
Method1:		α	α	α	α	α	α	α	α	α					:p_1
Method2:							α	α	α	α	α	α	α	α	:p_2

Fig: 3.2.2.3: Combination of methods

In Fig.3.2.2.3, method 'Method1' predicts location [2-10] as alpha helices with uncertainty value p_1 while 'Method2' predicts location [7-14] as alpha helices with uncertainty value p_2. There is overlapping at location [7-10]. The overlapping implies that there is stronger support for location [7-10] to adopt the structure alpha helices, thus, its uncertainty value should be higher than p_1 and p_2. The technique of propagation in this situation is discussed in techniques of uncertainty in expert system.

3.3 Overview of uncertainty techniques in expert systems

3.3.1 Background

In section (3.2.2), it has been suggested that the technique of calculating uncertainty in protein structure prediction can be explored by means of the techniques applied in expert systems. Indeed, the biggest advantage of expert systems is its ability to deal with uncertain data or information [CohenPR85]. There are expert

systems such as PROSPECTOR, MYCIN and others which deal with uncertain evidence. As the result of the rapid development of expert systems, various uncertainty techniques studied in depth by scientists from various disciplines, are revised, modified and developed to suit to the relevant problems.

Uncertainty plays important roles in expert systems because in real situations, a rule such as 'something will certaintly happen if certain conditions are satisfied' may not always apply. Furthermore, the truth of the conditions of the rules may not be known with certainty. Thus, in expert systems, besides the influence of the credibility of rules, the quality of evidence used also determines the strength of confidence in the solution.

3.3.2 Classification of evidence

The strength of evidence, in most cases, is represented by numbers. The numerical degree of uncertainty is used for: to distinguish between weak, medium or strong degrees of belief; to decide whether one hypothesis is more likely than another even if the interpretation of the hypotheses are not known; and a meaningful quantities such as probability[CohenFE85].

Uncertainty arises when the total uncertainty values for a proposition and its negation are less than one. This is formulated by assuming that the uncertainty for a proposition is x while its negation is y and x and y are non-negative numbers. The classification of evidence for a variety of uncertainty values x and y is shown below:

a) x=1 and y=0 classical logic
   x=0 and y=1

b) x+y =1 x and y are probability values
   x < 1, y < 1

c) x+y < 1 x and y are not probability values;
   1-x-y represent the disjunction of proposition

	and negation without commitment to either;
x+y > 1	represent inconsistency

3.3.3 Methods of Reasoning

Knowledge bases for expert systems are made up of facts and rules representing the expert's view of the application problem. They are separated from inference mechanisms in which the facilities for explaining how decisions are reached, are provided. Three inference methods used in expert systems are as follows:

a) *Deduction*

Deduction is one of the methods used for reasoning. The basic principle is to infer a new fact if the condition is true. Deduction is defined as a systematic method of deriving a conclusion that cannot be false when the premises is true. The example is as follows:

if P,R,S then Q

So, if P,R,S are true, then Q is deduced to be true. Although the technique is successfully applied for theorem proving and validation of methods, in a real application as in rule(i) it is not really useful. This is because, a real application always deals with inexact knowledge, and as a consequence, rules and facts in general cannot be guaranteed to be true.

if pattern($[a_1,a_2,..,a_n]$) then secondary_structure($[\alpha_1,\alpha_2,..,\alpha_n]$) ..(i)

pattern($[a_1,a_2,..,a_n]$) : true

secondary_structure($[\alpha_1,\alpha_2,..,\alpha_n]$) : ?

b) *Induction*

In order to arrive at the above rule, data exploration has to be made. It observes the occurrence of a pattern and the structure and since it holds for most cases, then it is decided to be generally true. The method of this reasoning is called induction. It is defined as a process

by which a general conclusion is drawn from a set of premises, based mainly on experience or experimental evidence. The conclusion is beyond the information contained in the premises, and does not follow necessarily from them. Thus, an inductive argument may be highly probable, yet lead from true premises to a false conclusion. The universal generalization is then replaced by a probabilistic statement which indicate that it is very likely to be true but that there is a risk it may not be.

c) *Abduction*

In many expert systems, however, the use of rules are the other way round. This method of reasoning is called abduction. Abduction is defined as a process by which the conclusion is investigated in order to find out how much the antecedent can be contributed for it to be true. The method of abduction for the previous rules is as below:

if pattern($[a_1,a_2,..,a_n]$) then secondary_structure($[\alpha_1,\alpha_2,..,\alpha_n]$) ..(ii)

secondary_structure($[\alpha_1,\alpha_2,..,\alpha_n]$) : true

pattern($[a_1,a_2,..,a_n]$) : ?

In this method, it observes the frequency of the pattern of amino acids occurring on all ocassions when the secondary structure is true. In this case, abduction, like deduction, requires that there are pertinent facts and apply this fact to infer a new fact. Unlike deduction, however, more than one answer is possible in abductive reasoning. In rule (ii), if the secondary structure alpha helices is true, then there are many other possible patterns involved. So the strength of the rule shows its possibility to be true.

3.3.4 Methods Dealing With Uncertainty

A variety of different methods have been proposed and five are recognized as methods dealing with uncertainty in expert systems

[Liu87]. Those are: the Bayesian approach based on probability theory, certainty factor, the Dempster-Shafer method based on the theory of evidence, approximate reasoning based on possibility theory, and non-numeric approach based on the theory of endorsement. The methods other than Bayesian were developed because of the suggestion that the probability theory is not a general theory to deal with all problems in expert systems. In this work, the approach using probability theory and theory of evidence are described.

3.4 Approach Using Probability Theory

3.4.1 Background

The probability theory is attractive in various application areas because the concept is well established. In expert systems, the Bayesian method is widely used when dealing with uncertainty using the probability approach. The most popular example of expert systems using the approach is PROSPECTOR. There are several criticisms on this method and those are normally focussed on the assumption of the conditional independent [Bundy84, White84, Quinlan83], the consistency of the result [Quinlan83, White84], and the updating scheme [Pednault81, Glymour85, Johnson86].

Frost [Frost86] gave more comprehensive deficiencies for the approach in which PROSPECTOR has been used as a model. In one of the deficiencies, it is criticised that as only a single value can be used for uncertainty on probability approach, the single value does not indicate the precision and in fact is a summary of 'for' and 'against' the hypothesis. The criticism is not true. In sampling techniques, in order to get overall average for sampling, there are always variations in the averages in this sample. The standard deviation for the sampling averages can be calculated. In our work of predicting the structure of protein, probabilities of amino_acid 'ala' associated with alpha helix in protein A, B,....N, are computed as p1,p2,...pN. The average probability

for 'ala' in protein and its standard deviation can be calculated. Thus, precision for the probability can be calculated based on the standard deviation as mentioned. The range of probability is shown as below:

$p_{lower} = p - s_p$; $p_{upper} = p + s_p$ (for a unit of standard error)

$p_{true} : [p_{lower}, p_{upper}]$

$p_{false} : [1-p_{upper}, 1-p_{lower}]$

The criticism of using a single value is presumably based on the approach in PROSPECTOR for which a single uncertainty value is being used. However, the example in PROSPECTOR does not reflect the capability of the technique. As it has been shown, the technique allows the precision to be attached with its probability to represent uncertainty.

The basic laws of probability are as follow:

****

L1 : Total probability of all events E_1 ,..., E_n is equal to 1.

$Sp(E_i) = 1$

L2 : If E is an event and probability of E is equal to p then probability of ¬E is equal to (1-p).

pr(E) = p, pr(¬E) = 1-p.

L3: If E_1 and E_2 are two events then:

$pr(E_1,E_2) = pr(E_1)\ pr(E_2 \mid E_1)$

If E_1 and E_2 are independent then:

$pr(E_2 \mid E_1) = pr(E_2)$

****

Fig.3.4.1 : Basic laws of probability

3.4.2 Bayesian Method in PROSPECTOR

Thomas Bayes has provided a theorem for evaluating the posterior probability of a hypothesis given a sequence of evidence as follows:

$$pr(H_i \mid E_1,\ldots,E_n) = \frac{pr(E_1,\ldots,E_n \mid H_i)\, pr(H_i)}{pr(E_1,\ldots,E_n)} \quad (1)$$

In developing PROSPECTOR, Duda, Nillson and Hart [Duda79] used assumptions and modifications on the method as follows:

$$pr(E_1,\ldots,E_n \mid H_i) = \prod_j pr(E_j \mid H_i) \quad (2)$$

$$pr(E_1,\ldots,E_n \mid \neg H_i) = \prod_j pr(E_j \mid \neg H_i) \quad (3)$$

$$\frac{pr(H_i \mid E_1,\ldots,E_n)}{pr(\neg H_i \mid E_1,\ldots,E_n)} = \frac{\prod_j pr(E_j \mid H_i)\, pr(H_i)}{\prod_j pr(E_j \mid \neg H_i)\, pr(\neg H_i)} \quad (4)$$

PROSPECTOR defined the term odd ϕ and likelihood ratio LS as follows:

$$\phi(E) = pr(E)/pr(\neg E) \qquad LS = pr(E \mid H) / pr(E \mid \neg H)$$

By using this definition, (4) will become:

$$\phi(H_i \mid E_1,\ldots,E_n) = \prod_j LS_j \phi(H_i) \quad (5)$$

3.4.3 Updating Method for Multiple Hypotheses

3.4.3.1 Overview

Many workers attempt to make use of multiple hypotheses using the probability approach developed in PROSPECTOR. Pednault and others [Pednault81], while analyzing the updating scheme, concluded that, if there are more than two hypotheses, no updating could take place and that the assumption (2) and (3)are too strong to be satisfied unless the hypotheses and the evidence are independent.

Glymour [Glymour85] refutes Pednault's claim with a counter example of rather special form but leaves open the question whether their result would be true with an added assumption to rule out that special case. Johnson [Johnson86] showed that Glymor's result does not hold even with the added assumption. Johnson used the conditions assumed by Pednault and concluded that, at most, one of the items of the evidence can alter the probability of any given hypothesis, but multiple updating for any of the hypothesis is precluded.

We believe that [Pednault81], [Glymour85] and [Johnson86] made their conclusion as a result of using equation (4) and assumptions (2) and (3). While assumption (2) looks reasonable for many applications, assumption (3) is unreasonably strong. This is shown as below:

Suppose there is hypotheses $H_1, H_2, H_3, ..., H_n$ and independent evidence $E_1....,E_n$:

$$p(E_1....,E_n \mid \neg H_2) = p(E_1....,E_n \mid H_1 \vee H_3 \vee ... \vee H_N)$$

Using assumption (2):

$$= \{p(E_1 \mid H_1) \vee p(E_1 \mid H_3) \vee ... \vee p(E_1 \mid H_N)\}..$$
$$.....\{p(E_n \mid H_1) \vee p(E_n \mid H_3) \vee ... \vee p(E_n \mid H_N)\} \quad \text{(iii)}$$

In PROSPECTOR, the assumption is reasonable because there is only a pair of hypotheses involved. Thus, the negation of the hypothesis is its alternative which is another single hypothesis. However, when there are more than two hypotheses, the method as applied in PROSPECTOR needs more information before it can be applied. Equation (iii) indicates that there are a lot of interactions between the posterior probabilities and information is needed about the interactions between the hypotheses. To ignore the existence of these interactions will inevitably be too strong.

In this case, it is suggested that the method used by PROSPECTOR is to be used for a pair of hypotheses H and ¬H. In order to use equation (5) for multiple hypotheses (more than two hypotheses), the formula is reformulated from equation(2) and

equation (3) and an additional independent assumption (6) is provided as follows:

$$pr(E_1,\ldots,E_n) = \prod_j pr(E_j) \qquad (6)$$

The new formula for multiple hypotheses is derived from (1) and, together with assumptions (2) and (6), will become (7) as follows:

$$pr(H_i \mid E_1,\ldots,E_n) = \prod_j \{ pr(E_j \mid H_i)/pr(E_j) \}\, pr(H_i) \qquad (7)$$
$$= \prod_j PF_j\, pr(H_i)$$

where $PF_j = pr(E_j \mid H_i) / pr(E_j)$

PF in this case is called the 'Propagation Factor' for the method when the evidence is supporting the hypotheses.

3.4.3.2 Conditions for propagation factor PF

The propagation factor PF_j produces three situations depending on the value of p(E | H). The value taken is one of the following:

$$P(E \mid H) = \begin{cases} 0 \\ \neq 0 \\ \text{don't know} \end{cases}$$

Charniak and Mc Dermott [Charniak85] resolve the problem of 'don't know' by transforming the equation (7) using the algorithm as follow:

$$\log(pr(H_i \mid E_1,\ldots,E_n)) = \log(p(H_i)) + \Sigma \log(PF_j) \qquad (8)$$

If the evidence E says nothing about the hypothesis, that is nothing known about the evidence on some of the hypotheses, then:

$$pr(E \mid H) = P(E)$$
$$\log PF_i = \log (P(E_j)/P(E_j)) = \log(1) = 0.$$

In this transformation, if nothing is known about the relationship between the evidence and the hypothesis, then it is assumed that they are independent. As the result, the value for the propagation factor is equal to 1 and its logarithm is 0. The purpose of the transformation is that the value of propagation is not change if nothing is known about relationship between evidence and hypothesis.

The transformation cannot hide the real issue. If the transformation is using $\log_e$ the outcome is not 0 anymore. Furthermore, if there is a real evidence that the posterior probability is 0, then the transformation is not working because the value its transformation is infinity.

An improvement for the above problem is proposed in this work:

a) if the value of $p(E \mid H) \neq 0$, then the value is used as usual;
b) if there are known that $p(E \mid H) = 0$, it is treated as zero. As the result, the probability for hypothesis will become zero which is consistent with the relationship.
c) if nothing is known about the relationship between evidence E and hypothesis H, propagation is not performed. This is the same effect as assuming E and H as independent and the value for PF is equal to one

Although the value of the propagation does not change in (c), the relative value of the hypotheses is affected. This is discussed in the next section.

Example:

Suppose that there are hypotheses H_1, H_2, H_3 which are mutually exclusive and exhaustive, while there are independence evidence $E_1,..,E_n$ and others E_e. Probability values of $p(E_i \mid H_j)$ are as

follows:

Hi	E1	E2	E3	E4	E5	E6	E ε	p(Hi)
H1	4/20	-	5/20	6/20	-	3/20	2/20	1/3
H2	6/30	8/30	-	1/3	5/30	-	1/30	1/2
H3	-	4/10	3/10	-	2/10	1/10	-	1/6
p(Ei)	1/6	12/60	8/60	16/60	7/60	4/60	3/60	

Fig.3.4.3.2(a) :probability of evidence given H is true

The value of probabilities for each hypothesis is calculated based on the equation (7). After a sequence of evidence, the value of probabilities are shown in Table 2 as below:

Hi	H1	H2	H3
p(H)	1/3	1/2	1/6
E1	4/10	6/10	1/6
E3	3/4	6/10	3/8
E6	27/16	6/10	9/16
..	..	..	..
..	..	..	..

Fig.3.4.3.2(b): probability of hypotheses given a sequence of evidence

In contrast to the methods of Pednault[Pednault81], Glymour[Glymour85] and Johnson[Johnson86], the result in Fig.3.4.3.2(b) shows that the probabilities of each hypotheses is updated for each evidence added. In

this case, for each evidence, it is treated as supporting the hypothesis even when the value of its probability is small. This is because the probability value cannot take negative value which can be used to show against the hypothesis.

Fig 3.4.3.2 applies the principle in this work as discussed in section (3.4.3.2), that is the value of hypothesis does not change when the evidence is not known whether it supports or against the hypotheses. In stage one, the value $p(E_1 \mid H_3)$ is not known, so the value of $p(H_3 \mid E_1)$ remains to be 1/6 as before that is for $p(H_3)$. In stage two, when evidence E_3 is added, the value $p(E_3 \mid H_2)$ is not known, so $p(H_2 \mid E_3)$ in stage two does not change. The situation is similar in stage three for H_2 when its value remains as before even with additional evidence E_6. The values for other hypotheses where the relationship between their hypotheses and the evidence are known, show improvement.

Although the updating scheme works, the probability values increase without any means of controlling it. This is shown in Fig 3.4.3.2, where the total values of the probability for the hypotheses are greater than one. The value is expected because of the independent assumption being used in the method. In method used in PROSPECTOR and another suggested by Charniak and Mc Dermott[Charniak85], the decision is based on the highest value of the probability. In this work, however, an improvement of the method is suggested to overcome the problem of the total probability and is discussed in next section.

3.4.3.3 Relative Probability

As the total probability exceeds one, it contradicts the law of probability (L1). A procedure is required to maintain the total value of probability at one after each evidence is added. This is not only to maintain the method according to probability law, but it is also to

provide a technique for comparing the hypotheses for making decision. In this work, a new parameter is developed to be called 'relative probability'. The parameter replaces the probability in equation(7) and ensures that the total value at each stage of additional evidence is one. The parameter 'relative probability' (Φ) is derived by normalizing the value of propagation for each hypothesis by its total. This is shown as follows:

From equation (7), suppose that there is evidence E_1, then:

$$p(H_i \mid E_1) = PF_i p(H_i) \qquad (9)$$

the equation (9) is divided by the total probability of the posterior probability $p(H_i \mid E_1)$ as below:

$$\frac{p(H_i \mid E_1)}{\sum_i PF_{i1}\, p(H_i \mid E_1)} = \frac{PF_{i1}\, p(H_i)}{\sum_i PF_{i1}\, p(H_i \mid E_1)} \qquad (10)$$

$$\Phi(H_i \mid E_1) = \frac{PF_{i1}\, p(H_i)}{\sum_i PF_{i1}\, p(H_i \mid E_1)} \qquad (11)$$

In the next propagation, the relative probability is used instead of the value $p(H_i \mid E_1)$. As the result, the new general formula of relative probability Φ for the hypothesis for a sequence of independent evidence $E_1, \ldots, E_n$ is shown in equation (12) as follow:

$$\Phi(H_i \mid E_1,..,E_n) = \frac{PFi\ \Phi(Hi \mid E1,\ldots,En\text{-}1)}{\sum_i PF_i\ \Phi(H_i \mid E_1,\ldots,E_{n-1})} \qquad (12)$$

The equation (12) is applied to the previous example and the result is shown in Fig.3.4.3.2(c). The relative probability of the

hypothesis for given a sequence of evidence provides facility for comparison between the competing hypotheses. This is because at each new evidence added to its propagation, the total value for all the hypotheses is always equal to one. So the decision easily can be made depending on its relative probability value compared to the other hypotheses.

There is another advantage from this technique. In previous techniques [Duda79, Charniak85] the value of propagation for hypothesis remains as before when nothing is known about its relationship with the evidence. The similar result is also shown in Fig 3.4.3.2(b) as discussed earlier. The approach using relative probability, however, reduces the significance of the hypothesis as shown in Fig.3.4.3.2(c). When nothing is known about the relationship between H_2 and E_6, there is no propagation involved, instead the percentage significance of relative probability $\Phi(H_2 \mid E_1,E_3,E_6)$ in competing hypotheses is reduced from 35% to 21% (the value of $\Phi(H_2 \mid E_1,E_3,E_6)$). This technique obviously provides a self adjusting facility for the system to make a decision based on the contribution of each hypothesis in the light of evidence. This is in sharp contrast to the previous techniques where the insignificance of the inactive hypotheses are not readily apparent because they are not being compared with the overall hypotheses.

Hi	H1	H2	H3
p(H)	1/3	1/2	1/6
E1	0.343	0.514	0.143
E3	0.43	0.35	0.22
E6	0.59	0.21	0.20
..	..	..	..
..	..	..	..

Fig.3.4.3.2(c): probability after normalization

3.4.3.4 Comparison with PROSPECTOR

The idea in PROSPECTOR is to use the term 'odd', that is the ratio of degree of belief over degree of disbelief (its negation), as a basis for decision-making process. Higher value of the ratio indicates that the hypothesis is more likely to be accepted. Although the value can be used as a criterion for competing hypotheses, there is no means to constraint the value and maintain consistency. The values increase as long as there is a new evidence to support it. The problem of consistency is not addressed and there is no idea about the effect of conditional independent assumption.

A different approach is adopted in this work. Equation (12) is formulated to take into account the problem of inconsistency and conditional independent assumption. Instead of to normalizing the value with its negation as in the PROSPECTOR, the probability value of a hypothesis is normalized over the total value of all the hypotheses. This is shown from the derivation of equation (12):

$$\frac{p(Hi \mid E1,\ldots,En)}{\sum_i p(H_i \mid E_1,\ldots,E_n)} \approx \Phi(H_i \mid E_1,\ldots,E_n) = \frac{PFi\ \Phi(Hi \mid E1,\ldots,En-1)}{\sum_i PF_i\ \Phi(H_i \mid E_1,\ldots,E_{n-1})} \quad (13)$$

It can be seen that in contrast to PREDMOLL, the normalization is based on the total value of posterior probability of the hypothesis. As a result, the relative probability follows the probability law and the following conditions hold:

$$\sum_i \Phi(H_i \mid E_1,\ldots,E_n) = 1$$

and $$\sum_i p(H_i \mid E_1,\ldots,E_n) = 1$$

Therefore, the relative probability is used as the probability value to maintain the consistency of total of probability.

3.4.3.Managing the problem of conditional independence assumption

Many workers dealing with uncertainty have indicated that the conditional independence assumption that is used in expert systems such as in PROSPECTOR is usually false. According to White [White85], there are no solutions as yet to those problems. Shortliffe suggests that the dependence among the inputs can be embedded in a larger condition if the rules are created carefully [CohenPR85]. Theoretically, the technique using the probability approach acknowledges that there is no easy way to solve the independent problem. The usual strategy adopted is to explore more information about the independence between the conditionals. But in reality, this approach is not practical when there is a large amount of evidence to be considered. The information to be explored is too large and the approximation using the assumption is considered to have some credibility depending upon the application of the problem.

The above problem is relevant to protein structure prediction.

The equations (12), is formulated based on the conditional independence assumption as follows:

$$pr(E_1,\ldots,E_n \mid H) = pr(E_1 \mid H)pr(E_2 \mid E_1,H)pr(E_3 \mid E_1,E_2,H) \ldots pr(E_n \mid E_1,\ldots,E_{n-1},H) \quad (12a)$$

independent assumption:

$$pr(E_1,\ldots,E_n \mid H) = pr(E_1 \mid H)pr(E_2 \mid H)\ldots pr(E_n \mid H) \quad (12b)$$

In principle, the secondary structure is made up from a sequence of amino acids with the correct order. The calculation of its probability, given a secondary structure H, is shown in equation (12a). However, with the conditional independence assumption, the order of the amino acids is no longer important as shown in equation (12b). This is because the value of probability is the same as long as the evidence is the same.

This is not true for the sequence of amino acids. The order of amino acids places great importance on the determination of the structure. Thus, the conditional independence assumption in reality is false. However, to follow directly equation (12b) is not practical as discussed earlier. In this work, the problem of conditional independence assumption is tackled as the outcome of the normalization. This is shown as follows:

Without the assumption, equation (2) is:

$$pr(E_1,E_2 \mid H) = pr(E_1 \mid H)pr(E_2 \mid H) - \rho_{12}\{ pr(E_1 \mid H)pr(\neg E_1 \mid H) pr(E_2 \mid H)pr(\neg E_1 \mid H)\}^{1/2} \quad (14)$$

is simplified by:

$$p = q - \omega \quad (15)$$

From (14) we can get ρ as below:

$$\rho_{12} = \frac{pr(E_1,E_2 \mid H) - pr(E_1 \mid H)pr(E_2 \mid H)}{\{pr(E_1 \mid H)pr(\neg E_1 \mid H)pr(E_2 \mid H)pr(\neg E_1 \mid H)\}^{1/2}}$$

where ρ_{12} is called *correlation coefficient* between $E_1 \mid H$ and $E_2 \mid H$.

From the equation (15), if the value of ρ_{12} is close to zero, then the conditional independence assumption is acceptable. Therefore:

$$pr(E_1,E_2 \mid H) \approx pr(E_1 \mid H)pr(E_2 \mid H) \qquad (16)$$

or $\equiv p \approx q$ where $\omega \approx 0$.

So, the probability of hypothesis for a sequence of evidence when the conditional independence assumption is acceptable:

$$pr(H \mid E_1,E_2) \approx pr(E_1 \mid H)pr(E_2 \mid H)\ p(H)/p(E_1,E_2) \qquad (17)$$

However, when $\omega \neq 0$, then the equation (16) is no longer valid. As a result, the affect ω has to be taken into account. This problem is solved as follows:

From equation(15):

$$p + \omega = q$$

then equation (17) will become:

$$pr(H \mid E_1,E_2) = (p+\omega)\ \{pr(H)/pr(E_1,E_2)\}$$

and is simplified as follows:

$$pr(H_i \mid E_1,E_2) = (p+\omega)\ h = ph + \omega h$$
$$\equiv p_i + \omega_i$$

where:

p_i is the actual value of the probability for hypothesis i;

ω_i is the additional value due to the assumption.

When the normalization is performed, the effect of the assumption is changed as follows:

$$\Phi_i \equiv \Phi(H_i \mid E_1, E_2) = \frac{p(Hi \mid E_1, E_2)}{\sum_i p(H_i \mid E_1, E_2)} = \frac{p_i + \omega_i}{\sum_i (p_i + \omega_i)} \qquad (18)$$

$$\equiv p_i + \varepsilon_i$$

In this case the effect of the conditional independence assumption ω_i does not exist any more and it is replaced by a new parameter ε_i. Equation (18) becomes:

$$\Phi_i = p_i + \varepsilon_i$$

$$\Phi_i - p_i = \varepsilon_i$$

$$\sum_i (\Phi_i - p_i) = \sum_i \varepsilon_i = \sum_i \Phi_i - \sum_i p_i = 1 - 1 = 0$$

So, the total effect of the conditional independent to the hypotheses is zero. Perhaps, an experiment can be carried out to show that $\sum_i \{\varepsilon_i\}^2$ is the minimum as compared to any other parameter by other approaches. So, it is shown that Φ is a good estimator for p where the effect of the conditional independence assumption is transformed into a new parameter, whose average is equal to zero.

3.5 Approach using evidence theory

3.5.1 Background

The evidence theory was introduced by Dempster and Glen Shafer. The theory is known as the D-S theory. An advantage of this theory over previous approaches is its ability to model the narrowing of the hypothesis set with accumulation of evidence. This is because an expert uses evidence which may not only give rise to a single

hypothesis but also to the sets of hypotheses, which together comprise the concept of interest. The disadvantage of this theory is its impracticality for computer based implementation due to an evidence-combination scheme that assures computational complexity with exponential time requirement. Gordon and others adapted this theory to apply on hierarchical relationships among hypotheses [Gordon85]. Shafer and others improve further by not using the assumptions imposed earlier by Gordon [Shafer87].

Another consequence of the generality of the D-S theory is avoidance of the Bayesian restriction that commitment of belief to a hypothesis implies commitment of the remaining belief to its negation. In D-S theory, measure of belief assigned to each hypothesis in the original set may sum to a number less than one. Some of the remaining belief can be directed to the sets of hypotheses that comprise higher level concept of interest.

D-S theory uses a number in the range [0,1] inclusive which indicates the degree to which the evidence supports the hypothesis. Unlike some other theory, negation of the hypothesis due to evidence against it, does not use a negative number.

The degree of belief for each evidence, denoted by m, on the subject Θ is represented by basic probaility assignment (bpa). The m(E) measures the total belief committed exactly to A, where A is an element of 2^{Θ}, the enlarge domain of all subset of Θ. If m(A) is p and m assigns no belief to other subsets of Θ, then m(Θ) = 1-p. Thus, the remaining belief is assigned to Θ and not to the negation of the hypothesis, as would be assumed in Bayesian approach.

3.5.2 D-S for singleton hypotheses

In the secondary structure prediction, each location adopts one structure only. In that case, the application of D-S theory to singleton

hypothesis is relevant. There are two cases to be considered:

a) they may confirm both hypotheses;
b) each may bring evidence to bear on different competing hypotheses.

Propagation for each case is shown in the following examples:

		α	Φ
		p1	1-p1
α	p2	p1p2 α	p2(1-p1)α
Φ	1-p2	p1(1-p2)α	(1-p1)(1-p2) Φ

Case 1: Both Confirm

		α	Φ
		p1	1-p1
¬ α	p2	p1p2(0)	p2(1-p1)¬ α
Φ	1-p2	p1(1-p2)α	(1-p1)(1-p2) Φ

Case2: Competing hypotheses

Case (1) : Belief $(\alpha) = 1 - (1-p_1)(1-p_2)$ .... (13)

Case (2): Belief$(\alpha) = p_1(1-p_2)/\{1-p_1p_2\}$ ..... (14)

3.6 Application of Bayesian method in prediction

3.6.1 Principle of Calculation

The calculation of uncertainty using the modified Bayesian method as shown in equation(12) is used for secondary structure prediction. The method is based on the following principle:

a) Let A be the hypothesis, that 'n' contiguous amino acids in the protein are possible to adopt to a particular type of secondary structure;
b) Let sequence $[p_1, p_2, .., p_n]$ represents propagation factor of 'n' evidence, $E_1, E_2, .., E_n$ to support the hypothesis A;
c) The calculation of relative probability, given the evidence

$E_1, E_2, .., E_n$ is used as if the evidence comes from various sources of knowledge;

d) The amount of belief for that location to adopt to the hypothesis A is then calculated using the formula as discussed in equation(12).

3.6.2 Method of application in prediction

The calculation of relative probability values at any location using the above principle involves the calculation of secondary structure *alpha, beta strand, beta turn* and *not_abt* (not alpha or beta strand or beta turn). The advantage of the method is the confidence of the decision for a particular location being calculated. For each location, the probability for all the hypotheses are shown and the decision is based on it. The example is as follows:

The system determines whether the location of the amino acid sequence $E_1, E_2, E_3, .., E_n$ will form an alpha helix.

$\Phi(\text{alpha} \mid E_1, E_2, E_3, .., E_n) = p_{\text{alpha}}$

$\Phi(\text{beta strand} \mid E_1, E_2, E_3, .., E_n) = p_{\text{beta\_strand}}$

$\Phi(\text{beta turn} \mid E_1, E_2, E_3, .., E_n) = p_{\text{beta\_turn}}$

$\Phi(\text{not abt} \mid E_1, E_2, E_3, .., E_n) = p_{\text{not\_abt}}$

$\Sigma\Phi(H_i \mid E_1, E_2, E_3, .., E_n) = 1.$

When the calculation for each secondary structures has completed, there are two types of tests to be carried out before the location can be confirmed as adopting alpha helices. The tests are as follows:

test 1: p_{alpha} is the highest ?

test 2 : $p_{\text{alpha}} \geq P_{\text{threshold}}$?

In the first test, it is checked whether the relative value is the highest among the structures while the second test, checks against the

threshold value. Failure of one of these tests results in the hypothesis being rejected. Acceptance of the hypothesis also attaches the reason of its acceptance i.e. the other structures are not suitable. The threshold value is the value based on the experiment. The value is derived from the precision required to be achieved at a pre-determined reliability.

3.7 Application of D-S theory in prediction

In section(3.2.2.3), the problem for the application of this theory was discussed. The technique is used when there are overlapping locations predicted by independent methods. The improvement of reliability is contributed from the overlapping locations. This can be shown as follows:

Method	structure	location	reliability
Method1:	alpha helix	[2-10]	0.6
Method2:	alpha helix	[7-14]	0.7
Overlap:	alpha helix	[7-10]	?

The reliability of the overlap locations is calculated as equation (13) as follows:

$$\text{reliability} = 1 - (1-p_1)(1-p_2)$$
$$= 1 - (1-0.4)(1-0.3) = 0.88$$

The reliabilities for the locations predicted by the methods are recalculated due to the new reliability for the locations that overlap:

$$\text{Method 1: reliability[2-10]} = \{ 5*(0.6) + 4*(0.88) \}/(5+4)$$
$$= 0.724$$
$$\text{Method 2: reliability[7-14]} = \{ 4*(0.88) + 4*(0.7) \}/(4+4)$$
$$0.79$$

The reliability contributed by the overlap locations is shared equally for each location of the method. The reliability for 'Method 1' increases from 0.6 to 0.724, while in 'Method 2', its reliability increases from 0.7

to 0.79. This is one of the technique being used to take advantage of the overlap between several predictive methods. The technique is used in the implementation of PREDMOLL for secondary structure prediction which will be discussed in Chapter Six.

3.8 Summary of the chapter

In this chapter, two major uncertainty techniques have been discussed with regard to an application in the secondary structure prediction. The first technique uses a Bayesian method where several modifications have been made. As a result of the modifications, the technique uses 'relative probability' as a means of measuring the probability of the hypothesis given a sequence of evidence. The principles involved in applying the technique in secondary structure prediction are described. The technique provides a better way of measuring the status of the hypotheses in the light of additional evidence where the total value of 'relative probability' for all hypotheses remains one. As a consequence, the values are self adjusted to the new evidence added to it. In the second technique, the method of improving the uncertainty technique using the D-S theory of evidence is described. The method is used when there are overlaps among the locations predicted by two independent methods.

CHAPTER FOUR
Analysis of Proteins With Known Conformation

4.1 Introduction

In Chapter Three, a prediction method using the concept of 'island' is proposed as one of the methods to be used for predicting secondary structure. The method uses the value of the 'propagation factor'. Its calculation is equivalent to the 'propensity' value in other methods[Chou74]. In order to provide the values, a set of proteins is used as its sampling units. So, the generation of the 'propagation factor' values based on the amino acids found at regions of secondary structure on a set of proteins is discussed in this Chapter.

Beside the secondary structure prediction, a method to be used in predicting tertiary structure is suggested in Chapter One. In this Chapter we define a set of hypothetical structural motifs based on secondary structure as the alternative for the lack of genuine tertiary structures for prediction. The structural motifs are created from the selected proteins. To facilitate the implementation, rules representing the structural motifs for each proteins are generated. A brief description of the rules is provided.

In order to generate the values and rules for secondary and tertiary prediction, we define criteria for assigning secondary structure from its protein database. The definitions are for secondary structures alpha helix, beta strand and beta turn. These definitions are useful for other workers when they compare their result with ours or vice versa.

Finally, we discuss a facility for locating patterns in sequences of amino acids. This facility is used to capture patterns which are more complicated than residue patterns. The description of the facility and its advantageous under the present work are discussed.

4.2 Protein Data Bank

4.2.1 Selection of Proteins

This section concerns the selection of proteins from the protein data bank and the transformation of their atomic configuration data into a more suitable form. To select a set of proteins, data from the Brookhaven Protein Databank are used. The data are generated using X-ray crystallography and consists of information about atomic configurations. The accuracy of protein structure assignment is determined by the accuracy of the coordinates. This accuracy ranges from less than 1.5Å(where individual side chains can be seen clearly) to unrefined structures at a resolution just sufficient to trace the protein chain.. In this work, fifty four high resolution proteins from the Brookhaven Protein Databank are used.

Information is assigned for each atom in the data-bank and the following attributes are used in this work:

****

Atom	-	The atom name.
AtomNum	-	The atom number
Residue	-	The amino-acid name
ResNum	-	The residue number
X	-	coordinate X of the atom
Y	-	coordinate Y of the atom
Z	-	coordinate Z of the atom

****

Fig.4.2.1: Attributes of atom required

4.2.2 Representing Data For Analysis

The information about each atom and its coordinates needs to be processed for higher order parameters. This provides clearer patterns for assigning a structure at a particular location. To transform the raw data to more meaningful parameters requires much calculation. Hence, an appropriate computer language, in this case C, is used to calculate parameters hydrogen bond (h_bond) and dihedral angle. In this work, these parameters are provided for each protein. However, some work has to be done to transform this information the appropriate form.

Beyond the process of calculating those two parameters, the rest of the work involves pattern recognition. This is done by searching patterns of h_bonds or/and dihedral angles formulated for several secondary structures. To implement these tasks, PROLOG which is ideal for this sort of work, is used. As a result, all the information about proteins are transformed as PROLOG facts. There are shown in Fig.4.2.2. The transformation involves some adjustments for negative numbers and the names of constants (which cannot start with a number or consist capital letters).

****

predicate	description
1. amino_seq(N,Name)	- N is amino acid number in the sequence. Name is the name of amino acid.
2. amino_coord(N,X,Y,Z)	- the coordinate of atom C_α for amino acid number N. X,Y and Z represent the axis of the coordinates.
3. dihed_angle(N, ϕ, ψ)	- the information about the dihedral angles ϕ and ψ.
4. h_bond(N_CO,N_NH,Diff)	- predicate to show that there is a hydrogen bond from CO of amino acid number N_CO to NH of amino acid number N_NH.

****

Fig.4.2.2: PROLOG facts for each protein

4.3 Assigning Secondary Structure

4.3.1 What Type of Algorithm ?

The successful analysis of the relation between amino acid sequence and protein structure requires an unambiguous and physically meaningful definition of structures. In the Brookhaven Protein Data Bank, several secondary structures have been assigned by crystallographers. However, according to Kabsch and Sander [Kabsch84], there are several shortcoming in the compilation. This is because the secondary structure assignments in the Brookhaven Protein Data Bank by crystallographers are often subjective and incomplete.

The problem arises because no two crystallographers work with exactly the same idea of what constitutes a secondary structure. Even if a standard measure is adopted, a problem remains as to the standard with which secondary structure prediction should be compared. For example, hydrogen bond patterns of secondary structures are not in dispute, but the criteria for hydrogen bonds themselves often are. This is because the bonds vary continuously in length and angle, and threshold values define their existence. According to Kabsch and Sander [Kabsch84], there is no generally correct hydrogen bond definition, as there is no sharp border between the quantum-mechanical and electrostatic regimes, and no discontinuity of the interaction energy as a function of distance or alignment. As the result, a definition of hydrogen bonds by one crystallographer may produce a long secondary structure, while another produces two pieces. The method by Kabsch and Sander [Kabsch84] is considered successful because they considered a full backbone representation and calculate a rough energy for each bond [Taylor87].

It is accepted that an algorithm for extracting structural features

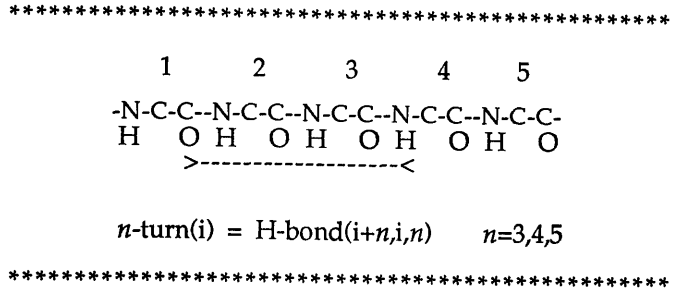
from atomic coordinates is best represented by a pattern recognition process. In order to determine whether a pattern is present or not in a continuum of possible atomic configurations, a threshold value should be used. There are two methods. One is to use the backbone dihedral angles phi and psi. The other is to use hydrogen bond patterns.

The idea of using hydrogen bond patterns to define α-helices and β–sheets were first used by Linus Pauling and Robert Corey in 1951 [Kabsch84]. Since then, hydrogen bond patterns have been used widely to assign such protein structures from their atomic coordinates. Kabsch and Sander [Kabsch84] developed a pattern recognition process for hydrogen-bonded secondary structures. They are recognized as repeats of elementary units of hydrogen bonding patterns *"turn"* and *"bridge"*. Repeating *turns* are helices, repeating *bridges* are 'ladders', and connected ladders are 'sheet'. Milner-White and Poet [Milner87II] have recognized hydrogen bond patterns for various other secondary structures such as gamma-turn, alpha-beta loop, alpha-alpha loops, beta-beta loops, paper-clip, beta-bulge and as well as alpha helix and beta sheet.

4.3.2 Hydrogen bond pattern

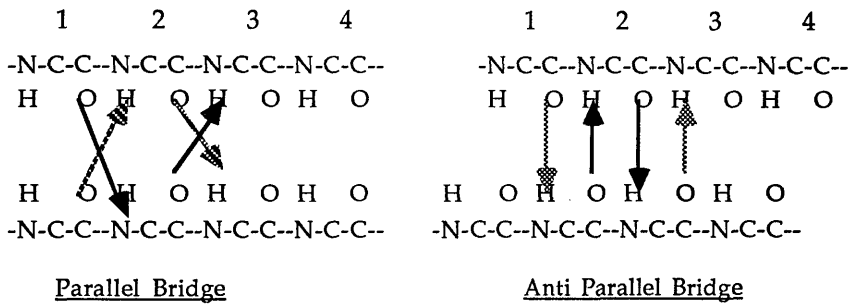
4.3.2.1 Elementary hydrogen bond patterns

The elementary hydrogen bond patterns can be represented as turns and bridges. The basic turn pattern is a single hydrogen bond of type (i+n,i), that is a hydrogen bond from CO of amino acid 'i' to NH of amino acid 'i+n' as below:



Diag. 4.3.2.1(a): Hydrogen bonds for turn

The basic bridge pattern is when there are nonoverlapping stretches of three residues each, (i-1, i, i+1) and (j-1,j,j+1), and each hydrogen bond has 'n' greater than 5. The nonoverlapping stretches form either parallel or anti-parallel bridges, as shown in Diag.4.3.2.1(b).



Diag 4.3.2.1(b): Hydrogen bonds for bridge

4.3.2.2 Patterns for Alpha Helix

An alpha helix is usually represented as a cylinder. Alpha helices can be further classified into three groups according to the types of their *"turns"*, i.e. alpha_3, alpha_4, and alpha_5. Alpha_3 is not uncommon, but usually has two or three weak hydrogen bonds. Alpha_5 is extremely rare. Alpha_4 is the most common alpha helix, but alpha helices themselves are rarely 'pure' (i.e. conform to one of these simple patterns): numerous hydrogen bonds in them are bifurcated, i.e. (i+4, i) and (i+3, i) or (i+5,i); the ends of them are often

overwound, ending in a 3-turn or alpha_3, or, underwound, ending in a 5-turn. The h_bond patterns for all types are shown as below:

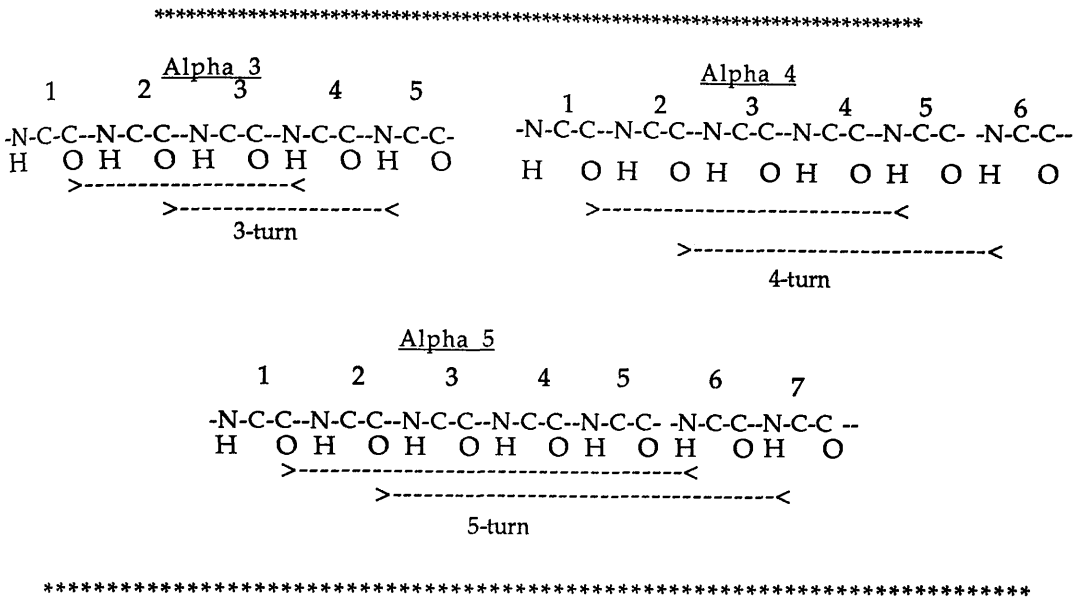


Fig:4.3.2.2: h_bond pattern for alpha helix

A minimal helix is defined by two consecutive n-turns. Longer helices are defined as overlaps of the minimal helices. The residues at both ends are not considered as part of the helix. So the minimum number of residues for alpha_3 is three(3), alpha_4(4), and alpha_5(5).

In this work, an additional criterion apart from hydrogen bonds is for the dihedral angle ϕ to be negative for the middle residues. This is to avoid including left-handed helices. All alpha helices are right-handed. An alternative criterion for helices is the use of main chain dihedral angles. The condition for dihedral angle ϕ is to be between -85° and -45° while dihedral angle ψ is to be between -60° to -10°. At least three consecutive residues must obey these conditions.

4.3.2.3 Patterns for beta strands

Beta sheet is grouped into three classes. The hydrogen bond patterns for each class are described below. In this work, the distance

between the hydrogen-bonded amino acids is set to be greater than five. As for alpha helices, dihedral angles are also used to define beta strands. The reason for this is that the hydrogen bonds can be between other secondary structures and may not solely belong to the beta strand. The valid length for a beta strand is greater or equal to three amino acids. .

```
beta_1 :  h_bond(1,P2,L) and (P2,3,M)
beta_2:   h_bond(1,P1,L1) , h_bond(P1,1,L2),
          h_bond(3,P3,N1), h_bond(P3,3,N2).
beta_3:   h_bond(P1,1,L1),  h_bond(3,P3,L2).
```

The range for dihedral angles is shown below:

$-180° \leq \phi \leq -30°$ $\qquad$ $170° \leq \phi \leq 180°$

$60° \leq \psi \leq 180°$ $\qquad$ $-170° \leq \psi \leq -180°$

4.3.2.4 Pattern for beta_turn

A beta turn is defined as having a hydrogen bond pattern as shown Fig.4.3.2.4, and the distance between Cα of the first and forth amino acid is less than 7°A. The definition is valid if it is not part of an alpha helix. The dihedral angles condition is restricted to the middle of the beta turn. However, due to the variation in dihedral angles, beta turns are separated into several classes. The dihedral angles are based on the classification given by Wilmot[Wilmot88] as shown in Fig.4.3.2.4. The angles are allowed to vary (+/-) 30°. A beta turn is only considered to exist when the middle amino acids at positions (i+1,i+2) satisfy one of the sets of angles.

beta turn	Position i+1		Position i+2	
	ϕ	ψ	ϕ	ψ
I	-60	-30	-90	0
I'	60	30	90	0
II	-60	120	80	0
II'	60	-120	-80	0
VIa	-60	120	-90	0
VIb	-120	120	-60	0
VIII	-60	-30	-120	120

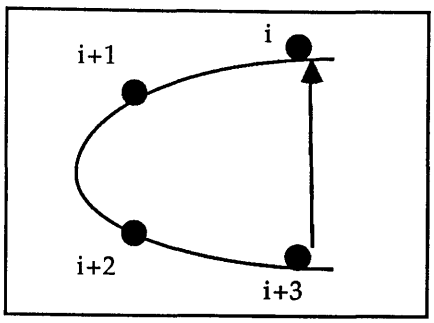


Fig: 4.3.2.4: Hydrogen bond patterns for beta turns and sets of dihedral angles at middle positions

4.3.3 Generating the 'propagation factor' for secondary structure

Once the criteria and patterns for secondary structures are formulated, their locations in the amino acid sequence can be determined. Beside the locations, classes of all the secondary structures in the protein are also produced. However, for the purpose of calculating the 'propagation factor', the information about the classes is not important. So, the overlap locations between several classes of secondary structures are resolved. This allows for each secondary structure to be represented by single class only.

The 'propagation factors' are derived from the information about the amino acids in the secondary structure. The formula to calculate this is shown as follows:

$fmotif_i(A)$ = frequency of amino acid i in the secondary structure A .

$fprot_i$ = frequency of amino acid i in the protein.

totmotif(A) = total residues in secondary structure A.

totprot = total number of residues in protein.

$prob_i(A)$ = probability of amino acid i belong to secondary structure A

$$= fmotif_i(A)/fprot_i \quad (1)$$

$$Ratio = totmotif/totprot \quad (2)$$

$$propensity_i(A = prob_i(A) / Ratio \quad (3)$$

The values of the 'propagation factor' for each amino acid for each type of secondary structure are shown in Fig 4.3.3. The secondary structure alpha_helix, beta strand, beta turn and not_abt(not alpha/beta/beta turn).

****

Amino	PF(alpha)	PF(beta)	PF(turn)	PF(not_abt)
cys	0.8296	1.1727	0.9612	0.9993
met	1.22283	1.116	0.5831	1.0364
asp	0.9047	0.3976	1.4011	1.1173
arg	1.0509	1.0607	0.9648	0.9399
phe	1.2191	1.5374	0.9357	0.6823
tyr	0.8607	1.3673	0.8675	1.0075
his	1.0075	0.9109	1.1636	0.9664
gly	0.3679	0.6266	1.7365	1.2951
val	1.0502	1.8251	0.5149	0.8292
ile	1.1668	1.7216	0.51196	0.7742
pro	0.6083	0.4026	1.3374	1.3192
asn	0.7563	0.4322	1.0869	1.3638
glu	1.5007	0.5616	0.9204	0.8180
gln	1.2122	0.9488	1.0093	0.8777
lys	1.2513	0.6983	0.9911	0.9791
ser	0.6227	0.9441	1.2333	1.204
ala	1.4159	0.8311	1.0066	0.8013
thr	0.7903	1.2912	0.820.4	1.1064
leu	1.4592	1.3205	0.4812	0.7780
trp	1.2185	1.5507	0.6397	0.7703

****

Fig.4.3.3: Values for Propagation factor

4.4 Tertiary and Super Secondary Structure

Protein structure analysis becomes more sophisticated at a level higher than secondary structure. Grouping of the secondary structure are known as super-secondary structure. This is less well defined than secondary structure[Lathrop87]. Super-secondary structure is at a level between secondary and tertiary structure. It consists of a few elements

of secondary structure that may or may not be sequentially adjacent and pack together in a regular way. As mentioned by Taylor [Taylor87], these motifs, which dominate the tertiary folds of most globular proteins, are probably energetically favourable and may provide stable nucleation centres in folding. The tertiary structure of a protein on the other hand, is usually unique for a particular protein. The three-dimensional shape of a protein directly determines its biochemical activity.

4.4.1 Creating Tertiary Structure for PREDMOLL

The difficulty of providing genuine rules for tertiary structures was highlighted in Chapter One. As the result, some forms of rules have to be provided for the implementation of PREDMOLL. In this work, the rules are created based on regular structures formed by a sequence of secondary structure. In this work, structures at levels higher than secondary structure, are to be called 'tertiary structure'. They are created based on their spatially adjacent characteristics observed in the OGSS. The spatially adjacent is characterized by hydrogen bonding between secondary structures. The tertiary structure for this work is classified into three types and two of them are shown in the analysis of protein 'b_act' as below:

```
**********************************************************
    /* group */
    group_order(1,not_abt,1,3).
    group_order(2,beta_strand,4,6).
    group_order(3,beat_turn,7,11).
    ......
    ......
    /* adjacent */
    adjacent(3,31,1).
    adjacent(3,4,1).
    ...
    adjacent(9,16,1).
    adjacent(16,20,2).
    ....
    ....
```

```
    /* tertiary structure */
    multiple_pattern(beta_strand,9,20,[9,16,20]).
    single_pattern(beta_strand,26,30).
**************************************************
```

Fig.4.4.1: Creating tertiary structure for PREDMOLL

4.4.1.1 Single Pattern

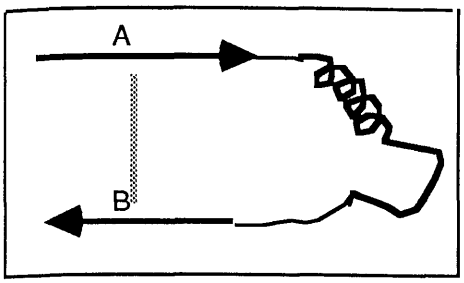
The first type of tertiary structure is called 'single pattern'. It consists of between three to eight secondary structures; of this the first and the last in sequence are spatially adjacent. An example of this structure is shown in Diag.4.4.1.1. The 'single pattern' in the diagram consists of a sequence of secondary structure: beta_strand, not_abt, alpha helix, beta turn, not_abt, and beta strand. The dotted line indicates that at least one hydrogen bond exists between the secondary structures A and B.

4.4.1.2 Multiple Pattern

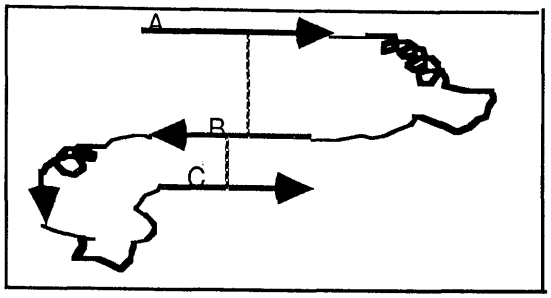
The second type of tertiary structure is called 'multiple pattern'. It may consists of three or more spatially adjacent secondary structures elements, each separated by one or more further secondary structural features. In the 'multiple pattern' diagram, the example given can be thought of as a combination of two 'single pattern'. The additional secondary structures are not_abt, alpha helix, beta strand, not_abt, beta turn, not_abt and beta strand. It can be seen that a minimum criteria for the type multiple pattern is as follows:

A adjacent to B, B adjacent to C ------> multiple pattern(A,C,[A,B,C])

In Diag 4.4.1.2, It shows that there is a multiple pattern for secondary structures A to C with interactions(indicated by dotted line) between A and B, and B and C.



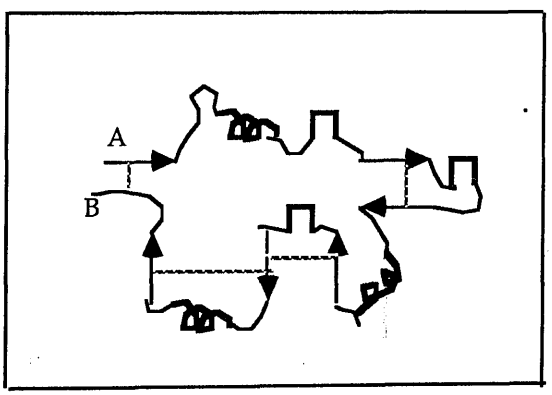
Diag. 4.4.1.1: single pattern



Diag. 4.4.1.2: Multiple Pattern

4.4.1.3 Independent

The third type of tertiary structure is called 'independent' and it consists of a group of secondary structures in which the first and the last element are adjacent in space. The distinct characteristic of this type is that no hydrogen bonds are allowed between amino acids in the group and amino acids outside the group. An 'independent' structure can consist of several independent groups, and even of groups type one and type two. In the 'independent' diagram, a 'multiple pattern' and a 'single pattern' is also part of the structure.



Diag. 4.4.1.3: Independent.

4.4.2 Rule Generation

In Chapter one, it was discussed that, once the secondary

structures have been predicted, the sequences of the secondary structures are compared with the sequences of secondary structures of proteins with known conformations (represented by the tertiary structures). All parameters, which represent the properties such as its location in the protein, the secondary structure sequence that forms its tertiary structure, and the relative orientation of the secondary structures, are stored. This information is represented in the rules of tertiary structure. The total number of rules for each type of tertiary structure out of 54 proteins are as follow:

****

Type	Total
single pattern	44
multiple pattern	59
independent	17

****

Fig.4.4.2(a): Total number of rules out of 54 proteins

Due to the large number of tertiary structures available out of 54 proteins, an automatic means of generating the rules is needed. The format of the rules is similar to the knowledge source for PREDMOLL as discussed in Chapter Two. An example of rules for predicting tertiary structure which are generated directly from the protein data base is shown as follows:

```
**********
ks_name(multiple71, 'Rule for multiple pattern').                    ...(a)
ks_problem(multiple71,predict_tertiary).
ks_blackboard(multiple71, [domain,tertiary]).
                              /* trigger conditions */
ks_trigger(multiple71) :-
        amino_unknown(Lq_Amino),
        secondary_unknown(Lq_Second),
        amino_known(b_sod, Lp_Amino),                                ....(b)
        secondary_known(b_sod,Lp_Second),

        exact_homology(Lq_Secondary,[beta_strand,.....,beta_strand]),  ...(c1)
        location_second_hom(From_Group,To_Group,List_Sec_Hom),
        ......
        ......
        inexact_homology(Listq_Amino, Listp_Amino, Value_primary),   ....(c2)
```

```
        inexact_nabt( [From_Group,To_Group,Lq_Second, Lq_Amino],          ...(c3)
                [3, 21, Lp_Second, Lp_Amino], Value_nabt),

        ksar_bind[ [ ksar_ks, multiple71],
                   [ksar_protein_known, b_sod],
                   [ksar_groupsecond_known, [3,21] ],
                   [ksar_groupsecond_unknown, [From_Group, To_Group]],
                     [ksar_similarity_primary, Value_primary],
                    [ksar_similarity_nabt, Value_nabt] ],
        fail.

.....
                          /* Action part of the knowledge source */
ks_action(multiple71) :-
        chosen_ksar(KSAR),
        ......
        ksar_groupsecond_unknown(KSAR,[N,M])
        list_from_pattern(N, [0,2,9], New_List),                          .....(d)
        assertz(tertiary_uncertainty([multiple,N,M],  New_List,KSAR)),
        .......
        get_locsecond(N,0,beta_strand,Lq_Second,L1),                      .....(e)
        get_locsecond(N,1,beta_strand,Lq_Second,U1),
        assertz(adjacent(KSAR,L1,U1)),
        .....
        ....
*****************************************
```

Fig.4.4.2(b): An example of a rule for tertiary structure

4.4.2.2 Description of the Rule

In this Chapter, we describe briefly functions of some predicates of the above rule(indicated by a, b,..). The objective is to provide information on how this knowledge source is being used to predict a tertiary structure in the implementation of PREDMOLL.

Indicator	Function
(a)	*Rule of Nomendature*: Each rule is named according to its order of creation by the system. In the example given, the rule is the 71th of the multiple pattern in the system. The next rule for the multiple pattern will be called *multiple72*.
(b)	*The name of protein where this rule is created* : The name

of the protein analysed in the above example is 'b_sod' which is shown in predicate 'amino_known'. The amino acid sequences and the sequence of secondary structures for both proteins ('b_sod' and protein in which the structure to be predicted) are represented by variables with their relevant predicates.

(c1) The aim of this function is to locate patterns that are similar to those pattern of secondary structure in protein 'b_sod'. Any exact similarity found in the unknown protein is considered as successful and the range of groups are recorded as shown by 'From_Group' and 'To_Group'. The objective of this matching is to locate regions in the unknown protein where primary structure homology can be performed for their alignment.

(c2) 'Inexact matching' for primary sequence homology between location in the unknown protein which is determined in (c1), and protein 'b_sod' at location group 3 to 21. The highest value of similarity for the homology is used to represent the alignment for the location.

(c3) The 'inexact matching' for primary sequence homology for 'coils' only at the same location as in (c). The objective is to find any similarity between such regions if the secondary structure is assumed to perfectly aligned.

(d) *Transform group*: This part is to transform the groups numbers used in 'b_sod' to the group numbers in predicted protein. The function to

do the transformation is:

list_from_pattern(N,[0,2,9], New_List)

The function indicates that there are three beta strands in this motif. The group numbers as mentioned before were from 3 to 21 but in this case, it was represented'[0,2,9]'. The numbers in the list represent the secondary structures(without secondary structure 'not_abt'). In the example, it indicates that beta sheets are formed by beta strands [0,2] and [2,9]. The transformation in the new list for the unknown conformation protein includes the group 'not_abt'.

(e) *Other Transformations*: In (d), the transformation is made on secondary structures that form the definition of the tertiary structure. However, within the location, there are various groups that are adjacent one to another. These orientations are transformed according to the group number of the predicted protein. The transformation is similar to the transformation in (d).

4.4.3 Primary structure homology

There are many ways of calculating the similarity value for two sequences. Blundell[Blundell87] stated that most of the methods are based on Waterman and Smith[Waterman76]. More comprehensive discussions on matching algorithms for primary structure homology are given by Collins[Collin87]. For the purpose of this work, the simplest method for calculating the similarity value is used to indicate a similarity alignment of a pair of subsequences.

```
**************************************************
SEQ1          - - A B C X Y Z D E - -
                  | | |       | |
SEQ2          s t a b c - - - d e j k
**************************************************
```

Fig4.5.3: alignment of two sequence

Fig.4.5.3 shows two sequences corresponding to the upper-case and lower-case letters. If SEQ1 is to be transformed to SEQ2, the letter s,j,k,t have to be added and X,Y,Z, deleted. In this work, the similarity value is expressed by the total score for each of pairwise comparison of residues(and for each comparison of a residue with a deletion). The similarity value used in the tertiary prediction to represent the uncertainty of a rule is calculated as follows:

$$\text{uncertainty} = \frac{\text{positive}}{\text{positive+delete1+added2}}$$

$$= \frac{5w_1}{5w_1 + 3w_2 + 4w_3}$$

w1 = matching weight
w2 = deleting weight
w3 = adding weight

The formula allows users to indicate the weight/score when a match is positive or when a residue has to be deleted or added. The simplest method, adopted in this work, is to assign the weight as 1 for both situations. In the example, the similarity value calculated by this technique gives the answer as 0.416.

4.5 Facilities of Locating Pattern

4.5.1 Motivation

PROLOG is useful when the user wishes to manipulate, search, or draw inferences from large body of data. This section concerns with

PATTERN, a program written in PROLOG to facilitate the locating of pattern for amino acid sequence.

4.5.2 Work by Morffew and Todd

Previous workers have used PROLOG to search for particular amino acid sequences. Morffew and Todd [Morffew86] used PROLOG for their query language to facilitate the the following queries:

ALA-GLY-ALA	(i)
ALA-GLY-any-ALA	(ii)
ALA-hydrophobic-GLY	(iii)

The example in (i) is to locate the pattern based on residues only. In (ii), 'any' indicates that it can take any one residues. The example below, clarifies the purpose of 'any':

ALA GLY - ALA	: no residue
ALA GLY ASP ALA	: 'any' refers to ASP
ALA GLY ASP ASN ALA	: 'any' refers to ASP ASN

In example (iii), the properties of the amino acid is used as a function. For example:

ALA MET GLY : residue MET is checked whether it is hydrophobic.

4.5.3 Facilities in PATTERN

PATTERN provides facilities as explained in section (5.1.3). Besides, several additional features are provided. The format is as follows:

```
**********************************************************************
pattern              :   < pattern(Name_or_pattern, List_of_member) >
List_of_member       :   [X | Tail]
```

```
X   :  name of amino acid
       | 'any_a'              any residue          (1)
       | 'any'                any                  (2)
       | physical/chemical    function             (3)
       | pattern(Name)        another pattern      (4)
       | range(RP)            range_pattern        (5)

RP  :  [pattern(Name),[Lower,Upper]]
Lower :                      number
Upper :                      number
```

Fig.4.5.3: Facilities in PATTERN

The types 'X' for (1),(2) and (3) are similar to the work by Morffew and Todd. Types (4) and (5) are additional features provides by PATTERN. Type (4) is the facility to call another pattern while type (5) is intended for contiguous patterns. Number of patterns in type (5) should be within the range specified by variables 'Lower' and 'Upper'. An example is as follows:

```
10   11   12   13   14   15   16   17   18   19   20
ala  gly  ala  val  val  asp  ile  lys  ser  gln  gly
```

Type (4):

```
pattern( pattern1, [ ile, lys, ser ]).
pattern( pattern2, [ ala, val, val, asp, pattern(pattern1), any, gly]).
location for pattern1: [ 16, 18]
location for pattern2: [12, 20]
```

Type (5):

```
pattern(pattern3, [val] ).
pattern(pattern4, [ range([pattern(3), [1,3]), any, pattern(pattern1)]).
location for pattern4 : [13,18].
range([pattern3, [1,3]) satisfies for sequence ' val, val' because it indicates
two of  pattern3, which is within the range of [1,3].
```

Fig:4.5.3(b): Example using PATTERN

The main advantage of PATTERN is the ability to use a complex structure as part of its pattern. This is in contrast to the work in (5.1.1) where the pattern is only in the form of the primary structure.

4.6 Summary

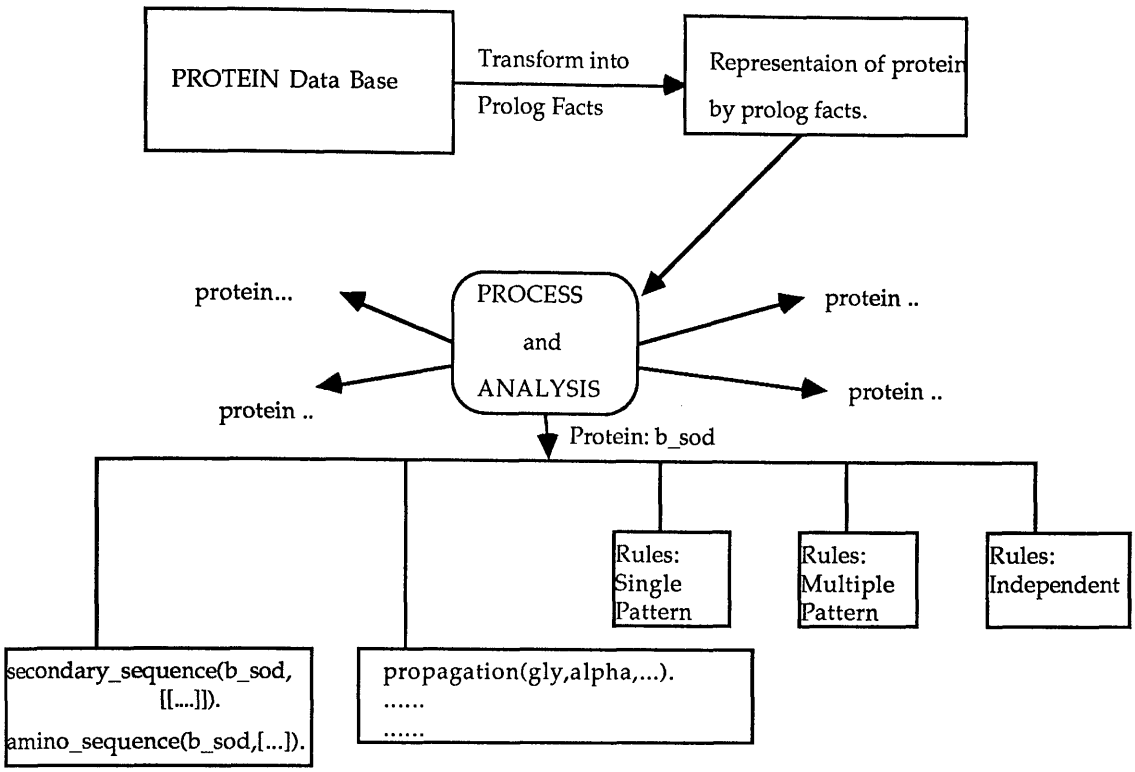
This Chapter described four types of information generated and used in the implementation of PREDMOLL. They are as follows:

a) For a given protein, each secondary structural element is located and ordered within the sequence. This information and the primary sequence are stored in one file.

b) The propensity values for each amino acid to adopt various secondary structures are calculated. The values are stored in the form of PROLOG facts and are used in predicting secondary structures by the method 'statistic'.

c) Rules are generated from the tertiary structures found in proteins in the protein data base. They are used for predicting structures higher than secondary structures. A simple method for calculating similarity value for primary structure homology is briefly described.

d) Facilities are for locating patterns in amino acid sequence using complex structure.



Diag.4.7: Information generated from protein data base

The consequence of information generated in (c) is the creation of rules and files which are required for implementation of tertiary structure prediction using PREDMOLL. This is shown in Diag. 4.7, where the process is shown from the protein data base up to the creation of rules and other necessary files. The advantage of (d) is the creation of facilities to locate patterns imposed by complex rules. Using PATTERN, the complexities of implementation are reduced.

CHAPTER FIVE
GENERATING STATISTICAL VALUES

5.1 Introduction

This Chapter concerns the implementation of the individual methods with the objective of generating statistical values to be used when the methods are combined. The values generated from this Chapter are used as the basis for making decisions during the implementation of PREDMOLL. The objectives of this Chapter are:

a) To generate precision and reliability values for each method. The methods are based on the heuristics rules (method 'statistic'), Lim's[Lim74] (method 'stereochemistry') and inductive learning[King87] (method 'inductive');
b) To study the effects of reliability values of methods 'stereochemistry' and 'inductive' when the uncertainty technique is being used on their predicted locations.
c) To generate the reliability values at overlap locations when methods are combined.

5.2 Reliability of Individual Method

In order to facilitate understanding of the performance of each method of prediction, the parameters of reliability and precision are used. Precision is defined as the ratio of the number of locations correctly predicted by the method to the total actual locations in proteins. Reliability is defined as the proportion of total locations correctly predicted. The values for these parameters can be calculated as follows:

N_T = total number of locations known to belong a

secondary structure in protein;

N_p = total number of locations predicted to belong to a secondary structure in protein;

N_c = total number of locations correctly predicted.

$$\text{Reliability} = N_c / N_p \qquad \text{Precision} = N_c / N_T$$

In some methods, the reliability of the method is more relevant. This is because the rules are used to recognize a particular pattern of a particular secondary structures. Thus, the method does not usually have a complete set of rules to be used for all patterns and all secondary structures for a protein. This is true in cases like the rules in method 'inductive' based on work by King [King87]. In that method, the rules are generated separately and based on their individual ability to predict secondary structure. There is no means of combining such rules so that they can be used as general prediction techniques. Currently, several rules are used to detect a particular secondary structure and no means is provided to overcome the overlap problem. Beside the problem of overlaps between similar secondary structures, overlaps between two different secondary structures are also not solved. In our work,they are treated separately and the evaluation is based on reliability for their individual rules.

We consider that predicting many locations but risking a large number of errors is not a preferable choice in prediction. So, the balance between precision and its reliability in this work is struck. Furthermore, investigations to reduce the number of locations predicted by the two methods 'stereochemistry' and 'inductive learning' are carried out to see the effect of using uncertainty technique to improve the reliability of their prediction. The implication of the investigation is to rely on a smaller number of locations but with more reliability. The reduction of the number of locations being predicted is compensated by other methods used in the same prediction. So, the combination of methods is directed to provide more rules which are

reliable enough to improve the precision.

In this work, three secondary structures are intended to be predicted. They are alpha helix, beta strand and beta turn. However, among the three methods that have been suggested, only the method 'statistic' has the potential to predict all the three. The prediction of the other two methods, i.e. method 'stereochemistry' and method 'inductive', is limited to alpha helix and beta strand only. Thus, in evaluating the performance of each method, the reliability and precision for each secondary structure is discussed.

5.2.1 Results of the method 'statistic'

5.2.1.1 Background of the method

The result is based on the implementation of several simple heuristic rules created in this work. Implementation of these rules in PREDMOLL is discussed in Chapter Six section (6.2). The rules are created based on the approach of creating 'islands' to predict the possible locations of the secondary structures (section 3.2.2.2). The rules for secondary structures are created based on different cases of connecting location between 'breaker' and 'former'. The boundary of a secondary structure is determined by the value of 'relative probability' of the location which is based on the values of the 'propagation factor' for each amino acid.

5.2.1.2 Analysis of result

The result of prediction is based on a series of experimental rules and threshold values. There are three simple experiments to be discussed in this section. The rules are created to determine the boundary in location between two 'islands'. The 'islands' are indicated by either 'breaker' or 'former'. Six basic rules used in the experiments are as follows:

1. gb_b : location between 'breaker' and breaker;
2. gb_temp : location between 'breaker' to 'former';
3. gtemp_b : location between 'former' to breaker;
4. gtemp_temp : location between 'former' to 'former' (space < 4);
5. gtemp_temp_f : location between 'former' to 'former' and spaces =< 4, and it is forward direction;
6. gtemp_temp_b : location between 'former' to 'former' as in rule(5) but the direction is backward.

5.2.1.2a) Threshold value 0.5

In the first experiment, the above rules are applied in general to all secondary structures. There are two conditions for the location of a sequence of amino acids to be accepted as adopting a particular secondary structure. They are as follows:

a) The uncertainty for the location to adopt the hypothesized secondary structure should be the highest;
b) The uncertainty must be greater than 0.5;
c) conflict structure is resolved by the uncertainty value.

The predictions are performed on 53 proteins and the overall result is shown in Fig. 5.2.1.1(a1). It indicates that the reliability of all the structures predicted are almost similar, that is in the range from 0.45 to 0.48. However, the precision predicted varies widely from one secondary structure to another. The precision for alpha helix is 0.59 which is interpreted to be comparatively not bad, while precision for beta strands and beta turns are poor. The poor result for beta strands and beta turns as shown in Fig.5.2.1.2(a) is possibly due to: the small number of locations being predicted by this method; and the rules are not suitable in predicting these structures.

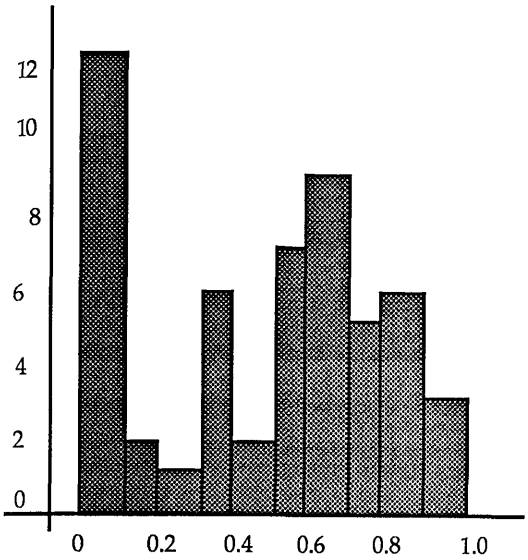
****

	alpha	beta	beta turn
locations/actual	2277	1548	1438
location/predicted	2984	456	209
correct	1348	221	98
precision	0.59	0.14	0.068
reliability	0.45	0.48	0.46

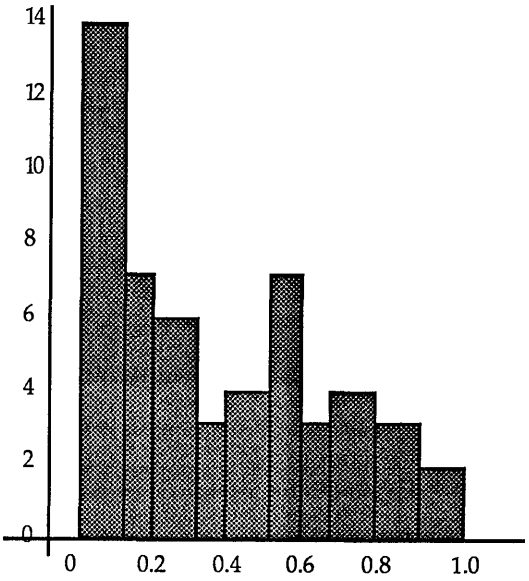
****

Fig:5.2.1.1.2(a1): precision & reliability threshold 0.5

The distribution of precision and reliability of alpha helix prediction for the proteins are shown in the diagram Diag.5.2.1.2(a2). It indicates that more than half of the proteins record precision greater than 0.5(50%). Out of the thirteen proteins with precision between 0 to 0.1, six of them (d_rxn, b_ebx, c_tln, a_pcy, b_rhe, b_sod) do not contain alpha helix structure in their protein. If the six proteins are excluded from the analysis, the percentage of proteins with precision higher than 0.5 is higher than in the above result. In the analysis for reliability, out of 14 proteins with reliability less than 0.1, six previous proteins are included. Without these proteins in the analysis, the diagram in Diag 5.2.1.2(a3) appears satisfactory.



Diag 5.2.1.2(a2): Precision (T= 0.5)



Diag 5.2.1.2(a3):Reliability (T=0.5)

The distribution of precision and reliability for beta strand prediction for proteins is shown in Fig.5.2.1.2(a4). Out of thirty four of proteins with precision between 0 to 0.1, thirteen of of them do not contain beta strands. For reliability, the thirteen proteins are part of twenty five proteins with their reliability between 0 to 0.1. This indicates that without the thirteen proteins, the distribution of reliability among the proteins is better than in Fig. 5.2.1.1(a4). However, the distribution of precision among the proteins is still inclined to be at lower end and none has a value greater than 0.6.

	0-0.1	0.1-0.2	.2-.3	.3-0.4	.4-.5	.5-.6	.6-.7	0.7-.8	.8-.9	.9-1.0
Precision	34	8	6	1	2	2	0	0	0	0
Reliability	25	1	3	0	4	5	4	3	4	4

Fig.5.2.1.2(a4): distribution of precision and reliability for beta strand(T=0.5)

5.2.1.2(b) Highest Certainty

In the first experiment, the poor precision of beta strand and beta turn prediction is thought to be due to the small number of locations being predicted by the method in section 5.2.1.2(a). So, as part of the strategy to increase the locations being predicted for those two secondary structures, some of the conditions that embodied the first experiment are dropped. Thus, in the second experiment, the condition for threshold value 0.5 (condition ii) is dropped. However, the two other conditions (condition i and iii) are still used. The overall result for each secondary structure in the second experiment is shown as in Fig. 5.2.1.2(b).

****

	alpha	beta	beta turn
locations/actual	2277	1548	1438
location/predicted	2995	763	241
correct	1307	299	110
precision	0.57	0.19	0.076
reliability	0.44	0.39	0.46

****

Fig:5.2.1.1.2(b): precision & reliability threshold 0.5

The slight difference between the results from section 5.2.1.1(a) and this section is obvious for each of secondary structures. More locations are predicted for all three secondary structures. For alpha helix, the locations predicted increase, but, among them, the locations that are correctly predicted are lower compared to the one in the first experiment. As the result, the precision is lower (from 0.59 to 0.57), as well as the reliability (from 0.45 to 0.44).

In the second experiment, the number of locations predicted for beta strands increase by 67% from the previous sections (from 456 to 763). Among them, the locations predicted correctly also improve

(from 221 to 299). As a result, the precision increases from 0.14 to 0.19, that is about 35% more than the previous section. However, the increase in the number of locations of beta strands predicted, also increases the error of prediction. The reliability of predicting beta strands using this method decreases from 0.48 to 0.39. For beta turns, the absence of a threshold value as one part of the conditions improves its prediction result. The number of locations predicted increases as well as the amount correctly predicted. So, in beta turn prediction in the second experiment, the precision increases from 0.068 to 0.076 while its reliability remains at 0.46 as in the previous method.

5.2.1.2(c) Additional Rules

In section 5.2.12(b), the improvement in beta strand and beta turn prediction is based on the threshold value 0.5 being dropped. The prediction can possibly be improved if some simple modifications are applied to the current rules. So, in the third experiment, the strategy is directed by adding rules to improve the prediction result.

In section 5.2.1.2(a) and 5.2.1.2(b) the methods are based on six general rules applied for all secondary structure prediction. In this section modifications are made to three of the rules: 'gb_temp', 'gtemp_b' and 'gtemp_temp'. They are broken into two classes: rules dedicated for alpha helix; and rules for other secondary structures (beta strand and beta turn). Rules for alpha helix are similar as the rules before without any changes but slight modifications are made to rules for other structures. In the changes in rules for beta strand and beta turn, the space to be connected between 'island's is reduced to be not more than one space (more than one spaces for alpha). In addition, the condition 'the highest uncertainty' is imposed on each location if the island is connected (not in previous rules). These measures increase the chances of beta strand and beta turn being selected.

As the result of three extra rules for beta strand and beta turn,

nine rules are used in the third experiment. The result of this experiment is shown as in Fig.5.2.1.2(c).

****

alpha	beta	beta turn	
locations/actual	2277	1548	1438
location/predicted	2960	763	339
correct	1321	325	140
precision	0.58	0.21	0.10
reliability	0.45	0.43	0.41

****

Fig:5.2.1.1.2(c): additional rules

The precision of alpha helix is almost the same as that in the method used in section 5.2.1.2(a), i.e. 0.58 compared with 0.59. However, the reliability for both methods is the same (0.45). If the result is compared to that of the method in the second experiment, the extra rules in the third experiment improve the values by 0.01 for both reliability and precision.

Extra rules and without a threshold value in the third experiment provide several improvements in precision in beta strand and beta turn prediction, compared to the first experiment. For beta strands, precision increases from 0.14 in method 5.2.1.2(a) to 0.21, which is about 50% improvement. A similar situation is found for beta turns where its precision increases from 0.068 to 0.10, which is about 47%. However, the reliabilities for both are lower compared to their reliability in method 5.2.1.2(a).

A similar comparison for beta strand and beta turn prediction is made between the method in third experiment and that in the second experiment. The latter produces a better result in both secondary structure predictions compared to the similar prediction for the method in section 5.2.1.2(b). The precision for beta strands increases

from 0.19 to 0.21 while its reliability increases 0.39 to 0.43. However, in beta turn prediction the precision increases from 0.076 to 0.10 while its reliability is down from 0.45 to 0.41.

5.2.1.3 Summary of the result and method

The prediction in method 'statistic' is done using a series of experimental rules and conditions. Although the experiment does not truly represent the true capability of this method, the prediction result shows potential improvement if the rules and conditions are improved. This is because with the facilities in PREDMOLL and the uncertainty method used there is the possibility of being able to create many rules and control the prediction process easily. The result of these three experiments are summarized as below:

a) the reliability of the method (threshold value 0.5) is not very different from other empirical methods as discussed by Taylor[Taylor87], that is 45% to 50%.
b) the overall precision for alpha helices is acceptable, eventhough the number of rules are small. The precision for beta strands and beta turn is less encouraging, but can be further improved by modification and addition of the rules. This has been shown by the series of experiment in this section;
c) The islands are allowed between breakers only. As the result, boundaries beyond breakers are not explored;
d) Proteins are treated without classification during the generation of their propensity values. As the result, the variations in propensity between proteins is greater;
e) The number of rules used may be too small. Perhaps more rules are required to achieve a better precision.

5.2.2 Result for method 'stereochemistry'

5.2.2.1 Background of the method

The method is based on rules developed by Lim[Lim74]. A brief description of the method is explained in Chapter One (section 1.1.3b). The method is used to predict alpha helices and beta strands for globular proteins. There are two stages: the first is to predict alpha helices; in the second , and the remaining regions are used to predict beta strands.

The method involves a large number of complicated rules. The rules are not only difficult to implement but sometimes can be confused due to large number of parameters involved. In the method, the locations of alpha helices are hypothesized to be around linked hydrophobic pairs(1,5). This can be extended by other hydrophobic pairs or mixture of hydrophobic and hydrophilic. This is shown in Diag. 5.2.2.1.

```
**************************************************************
1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
   G  H  M  H  H  A  H  H  S  H  H  A  H  A  A  H  G
            xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      yyyyyyyyyyy                        yyyyyyyyyyy
   zzzzzzzzzzzz                                zzzzz

H = large hydrophobic,     G = large hydrophilic
**************************************************************
```

Diag.5.2.2.1: Segments for alpha helix

The definitions of segment X, Y and Z are as follow:

segment X : a chain fragment of five residues which has one hydrophobic pair (1-5), or,
a fragment of all linked hydrophobic pairs(1-5);

segment Y : there is hydrophobic pair (1-4) overlap at both end of

segment X, one or both external position of this pair have G type residue. The segment X and the extensions are called segment Y;

segment Z : extend the segment Y to residue type G for one or both end. It is done if G is from:

a) hydrophobic-hydrophilic pair (1-2),(1-4) or (1-5); or,

b) an electrostatic pair (1-5)[1];

region joined to X and Y-fragment must not have H residue.

Segment X, Y and Z are hypothesized as alpha helices if they obey several other conditions. Beside this rule, there are five other complicated rules to be used for alpha helix prediction in this method. The remaining regions after alpha helix prediction are used for beta strand prediction. There are another eight complicated rules to be used in beta strand prediction. The implementation of all the sophisticated rules in this method is under way using PATTERN, a facility for locating patterns for amino acid sequence. The result presented in this work is based on the first two major rules for alpha helix prediction and the first four rules for beta strands. The result is sufficient in this work at the moment because it provides an additional method for implementing PREDMOLL.

5.2.2.2 Result of prediction

The prediction for this method is carried out for 53 proteins. The result of the prediction is compared with actual structures of the proteins. The overall reliability and precision for the prediction in this method is shown in Fig.5.2.2.2 as follow:

[1]electrostatic pair is hydrophilic pair(1-5) and within it there is only one hydrophobic residue

****

	alpha	beta
locations/actual	2277	1548
location/predicted	3614	539
correct	1202	189
precision	0.53	0.12
reliability	0.33	0.35

****

Fig:5.2.1.2(b): precision & reliability for method 'stereochemistry'

The result as shown in Fig.5.2.1.2(b) is not encouraging. Perhaps, this is because the result does not represent a complete set of rules for this method. The precision of alpha helix is not too bad, but precision for beta strand prediction is considered poor. Reliability for both structures are much lower compared to the method 'statistic' discussed before.

If the result of this method is compared to method 'statistic', the result of this method is disappointing. This is because, eventhough the results represent only part of the complete rules, the rules implemented in this work do represent the core of the method. Considering the difficulty of implementing the rules in this method compared with the simple rules in the method 'statistic', the lower values of its precision and reliability are not expected (However, Taylor[Taylor87] did mention lower result for this method by some workers).

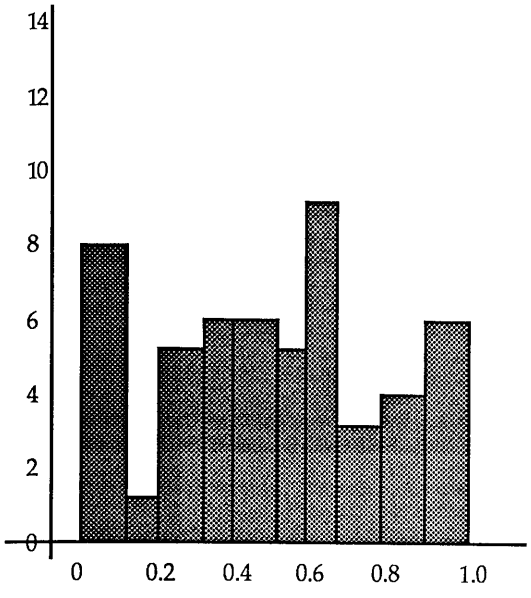


Fig.5.2.1.2(b1): Distribution for Precision

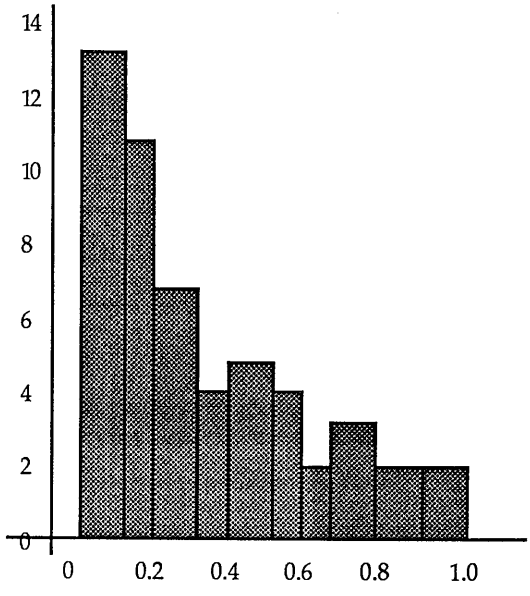


Fig.5.2.1.2(b2): Distribution for reliability

The distribution of precision and reliability of alpha helix prediction for 53 proteins are shown in the diagram Diag.5.2.2(b1) and Diag.5.2.2(b2). More than half of the proteins recorded a precision greater than 50%. As discussed earlier for the result of method 'statistic', six out of eight proteins with precision between 0 to 0.1 do not contain alpha helix. For reliability, considering six out of 14 proteins have values less than 0.1, the results are adequate.

5.2.3 Result for method inductive

5.2.3.1 Background of the method

A brief description of the method is discussed in Chapter One. In this method, two sets of rules were generated by King[King87] to be used in prediction. The first set consists eight rules to be used to predict alpha helix while the second set, consists of five rules for predicting beta strands. The rules are as follows:

a) ***Alpha helices***

1. [tiny_or_polar,large,aromatic_or_m,large,large_and_not_negative]
2. [negative,large,large,large_and_not_negative]
3. [large,large,small_and_not_p,large,large,large,large,large,large]
4. [large_and_polar,hydrophobic,large,large_and_not_aliphatic,large,
aromatic_or_aliphatic]
5. [polar_and_not_aromatic_or_charged,large,large,
small_p_or_polar_and_not_aromatic,large,large,aromatic_or_m]
6. [all_minus_p,charged,large,large,small_p_or_polar_and_not_aromatic,
large_and_polar,large,large]
7. [tiny,tiny_or_polar,large_large,positive]
8. [large,large,small_and_not_p,large_and_polar,large,large,tiny_or_polar_and
_not_aromatic,small_or_polar]

b) ***beta_strand***

9. [aliphatic,aromatic_or_aliphatic_or_m,aliphatic]
10. [tiny_or_polar,aliphatic_or_large_and_non_polar,large_and_not_negative,
aliphatic_or_large_and_non_polar,small]
11. [large_and_not_aliphatic,aliphatic,large_and_not_aliphatic,
aliphatic_or_large_and_non_polar,small_and_not_p_or_polar]
12. [large,aliphatic,large_and_not_negative,aliphatic_or_large_and_non_polar,
small_and_not_p_or_polar]
13.[aromatic_or_aliphatic_or_m,hydrogen_bond_doners,aliphatic_or_large_and
_non_polar,tiny_or_polar,aromatic_or_aliphatic_or_m].

The rules are easily implemented using the facilities in PATTERN. When several rules are used, two cases are bound to happen as follows: overlap locations with similar secondary structure; overlap location with different structures. Since no technique is discussed to solve the problems arising from overlapping, the rules are treated separately. So, the reliability for each rule is used to measure their performance while precision is not relevant due to the small number of predicted locations for all of the rules.

5.2.3.2 Result of the method

The results of prediction are shown in Fig 5.2.3.2. It appears that none of the rules achieve the expected reliability of 0.63 as suggested for the method[King87]. Six out of the thirteen rules achieve around 0.5, while two rules, king_alpha_6 and king_alpha_8 could not predict

one location in 53 proteins.

	α1	α2	α3	α4	α5	α7	β1	β2	β3	β4	β5
Tot. Predict	288	380	81	147	98	250	247	187	709	246	179
Correct	120	187	27	76	41	124	126	100	174	94	92
Reliability	0.41	0.49	0.33	0.51	0.42	0.5	0.51	0.53	0.245	0.38	0.51

Fig. 5.2.3 Reliability of inductive learning.

5.3 The effect of uncertainty technique on the additional methods

5.3.1 Motivation

In the previous section, the result for individual methods is shown and stored in their respective files. In the method 'stereochemistry', the result showed that many locations are predicted for alpha helices, but at the same time the higher error reduces reliability. It has been mentioned elsewhere in this work that few but reliable predictions are preferable than to many and unreliable. In that case, the locations that have been predicted should be screened to see whether they can be accepted or rejected.

The screening approach is used to reject locations which suspected to be untrue. However, in this work it is not part of prediction. It is applied after prediction when a particular method is completed. So, it does not recommend any additional boundary for the locations because it only investigates the result of prediction. As an example, in prediction using the method 'stereochemistry', the reduction of locations for alpha helices does not increase the locations for beta strands because screening is not part of prediction. As the result of screening, the number of locations used as the basis for calculating precision and reliability after the screening is reduced. Thus

the values of precision and reliability for the method are expected to be affected.

This approach provides the opportunity to investigate the use of the uncertainty technique in order to improve the reliability. In this section the reliability for methods 'stereochemistry' and 'inductive' in predicting secondary structure alpha helix and beta strand are investigated. The result of this work will be used during the implementation of PREDMOLL.

5.3.2 Method of application

The application of this method is done in two stages. In the first stage the locations are predicted as usual for the individual method. In the second stage, the locations that have been predicted are tested. The location is accepted for adopting a particular secondary structure if its uncertainty value ('relative probability') is greater than a pre-determined threshold value(in this case the value is 0.5). The example is shown in Fig.5.3.2. as below:

```
**************************************************************************
          1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
Predicted:   α  α  α  α  α     β  β  β     α  α  α  α  α
Certainty:      L1: 0.65          L2: 0.55          L3: 0.35
Result:      α  α  α  α  α     β  β  β
**************************************************************************
```

Fig 5.3.2: Screening predicted locations

In Fig.5.3.2, three locations in the sequence are predicted to be alpha helices at location [2-6], beta strands at location [8-10] and another alpha helices at location [12-16]. For each location, its uncertainty value is calculated. The values are 0.65, 0.55 and 0.35 consecutively. Since the threshold value is 0.5, locations [2-8] and [8-10] are accepted, while location [12-16] is rejected.

5.3.3 Effect on method 'stereochemistry'

The screening technique as mentioned in previous section is used for predicting the results of 53 proteins using method 'stereochemistry'. The reliability and precision for overall proteins are shown in Fig.5.3.3(a) as below:

****

	alpha	α reduced	beta	β/reduced
locations/actual	2277	2277	1548	1548
location/predicted	3614	1194	539	57
correct	1202	684	189	20
precision	0.53	0.35	0.12	0.013
reliability	0.33	0.57	0.35	0.35

****

Fig:5.3.3(a): precision & reliability threshold 0.5

The result shows a large reduction in locations compared to those predicted earlier by the method. For alpha helix, the number of locations predicted is reduced from 3614 to 1194. One of the consequences is reduction in precision from 0.53 to 0.35. However, the technique achieves the objective with tremendous improvement on its reliability, that is from 0.33 to 0.57. For beta strand, the technique does not improve its reliability despite the reduction of its precision.

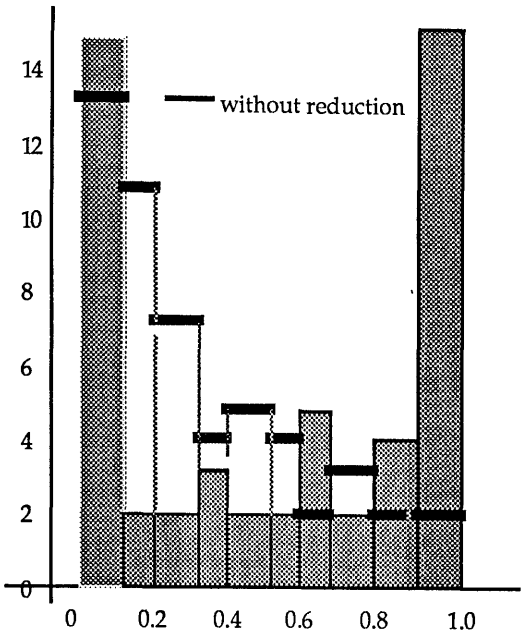


Fig:5.3.3(b): Distribution of Reliability for proteins in method 'stereochemistry'

The distribution of reliability among proteins is shown in Diag.5.3.3(b). The distribution is compared before and after screening. It indicates an improvement of the distribution for alpha helix prediction. There are more proteins at the better end of reliability after screening, compared with before. This shows that screening improves the reliability of alpha helix prediction in the method 'stereochemistry'.

5.3.4 Effect on the method 'inductive'

A similar technique is used to screen the result of predictions in the method 'inductive'. The effect of this technique is shown in Fig.5.3.4(a) and Fig 5.3.4(b). In beta strand prediction, a large improvement in reliability for rules β_3, and a slight improvement in reliability for rule β_1 and rule β_2 are seen. The reliability for rules β_4 and β_5 record a slight reduction. In alpha helix prediction, all rules except rule α_3 record improvements in reliability compared to

that without screening. Rule α_3 records a little reduction. So, the overall performance for the rules in the method 'inductive' show that the screening technique is useful.

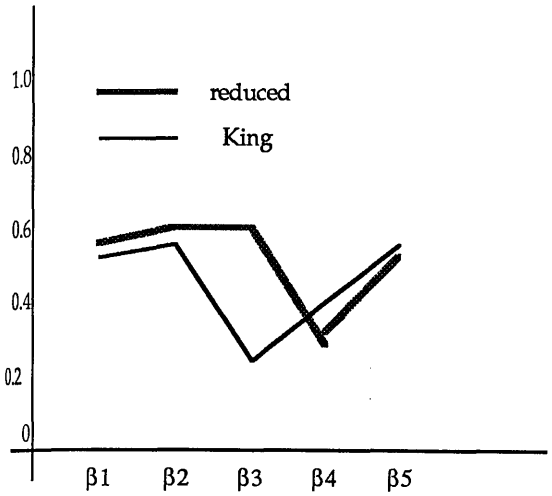


Fig 5.3.4(a): results of screening technique for beta strand prediction.

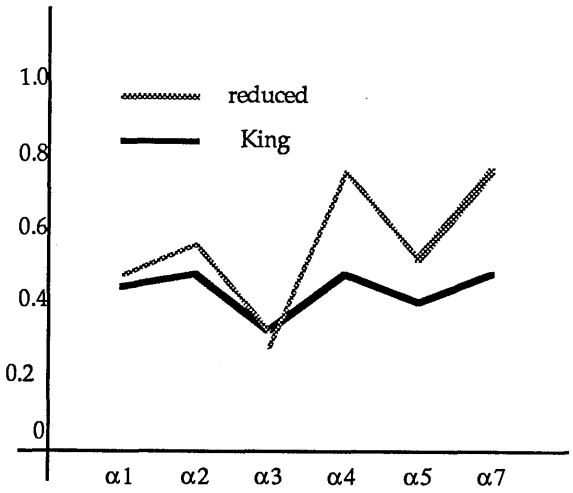


Fig 5.3.4(b): results of screening technique for alpha helix prediction.

5.4 Combination of methods

5.4.1 Motivation

When several methods are used for prediction, overlapping is not avoided. This is because the methods are not always mutually exclusive, so overlapping is to be expected. In this work, overlap between locations predicted by different methods is investigated. In many situations, overlap with a similar secondary structure at a particular location, increases the strength of the hypothesis for secondary structure. On the other hand, an overlap location with two different secondary structure predicted by two different methods, is an indication of the need to be more caution about adopting the hypothesized structure. It encourages the prediction technique to look for other evidence before a decision is made.The result of the investigation at overlap locations is used in the implementation of PREDMOLL. In the implementation, reliability for supporting the

hypothesis (overlaps with similar secondary structure) is used as the criteria in selecting the preferable structure for locations.

5.4.2 Method and calculation.

In the present work, overlaps are considered for locations with similar structure only. The reliability for the overlap locations is calculated for each secondary structure. Although the rules for method 'inductive' are treated independently, they are considered to represent a single method. Thus, the calculation for overlap locations between the rules in method 'inductive' are not done. An example for the calculation of reliability when the locations are predicted by two different methods is shown in Fig.5.4.2 as below:

```
**************************************************************************
          1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17
 Real:          α  α  α        β  β  β     α  α  α  α  α
 Method1: α  α  α  α        α  α  α     β  β  β     α  α  α
 Method2:    α  α  α  α  α     β  β  β  β  β     α  α  α  α
**************************************************************************
```

Fig 5.4.2: Screening predicted locations

In the example, the calculation for reliability is as follows:

total overlap for alpha[Method1 and Method2]	=	6
correct	=	4
P(alpha true | Method1,Method2)	=	0.67
P(alpha true | Method1,¬Method2)	=	0
P(alphatrue | ¬Method1,Method2)	=	0.67

In the above example, the reliability for alpha helix at overlapping locations for 'Method1' and 'Method2' is 0.67. However, the reliability of alpha helix prediction for 'Method1' without overlap locations by 'Method2' is 0. Conversely, the reliability of alpha helix

prediction for 'Method2' without overlap locations predicted by 'Method1' is 0.67.

5.4.3 Combination of two methods

A similar calculation is done for overlap locations predicted by method 'statistic' and 'stereochemistry'. The reliability for alpha helix and beta strand are shown in Fig. 5.4.3(a). For method 'statistic', the prediction result based on the discussion in section 5.1.1.2(a) is used. For method 'stereochemistry', the reduction is done based on threshold value 0.5.

	alpha	α reduced	beta	β/reduced
r(True | statistic,stereochemistry)	0.51	0.60	0.58	0.50
r(True | ¬statistic,stereochemistry)	0.211	0.4	0.31	0.15
p(True | statistic,¬stereochemistry)	0.39	0.38	0.46	0.48
r(stereochemistry)	0.33	0.57	0.35	0.35
r(statistic)	0.46	-	0.48	-

Fig:5.4.3(a): precision & reliability threshold 0.5

The reliability for alpha helix and beta strand prediction at overlapping locations show much improvement if compared with the reliability for their individual methods. For method 'statistic', reliability for alpha helix increases from 0.46 to 0.51 (0.6 if reduced) while for beta strand it is increased from 0.48 to 0.58. Larger improvements are shown for method 'stereochemistry' with the increment for alpha helix from 0.33 to 0.51 while for beta strand, it increases from 0.35 to 0.58. The reliability for locations predicted by method 'stereochemistry', but without prediction by method 'statistic', shows lower reliability : 0.21 for alpha helix and 0.31 for beta strand. On the other hand, the reliability is reduced but not much for alpha helix

if the prediction is without support from method 'stereochemistry' (0.38). For beta strand, the reliability is not much different(from 0.48 to 0.46).

The reliability at overlap locations predicted by method 'statistic' and 'inductive' is shown in Fig.5.4.3(b). It shows improvement for method 'statistic' (from 0.46 for alpha helix and 0,.48 for beta strand) in all rules except rules α_3, β_4 and β_5. Among the three rules, only rule β_5 records larger reduction in its reliability (from 0.48 to 0.22). Larger differences in reliability at location predicted by different methods and overlap, produces a better information. An extreme result provides a significant indication whether a structure should be adopted at the location or not.

The reliability for locations predicted by method 'statistic' but not by method 'inductive' is in the range of 0.44 to 0.50. This is perhaps due to fact that the number of locations predicted by rules in method 'inductive' are small. On the other hand, the reliability for the rules when they are not supported by method 'statistic' produces a lot of variations. The variation is useful where lower reliability of locations demand rejection and higher reliability demands acceptance.

	α1	α2	α3	α4	α5	α7	β1	β2	β3	β4	β5
[T | P, K]	.51	.55	.41	.65	.51	.60	.55	.76	.57	.46	.22
[T | ¬P, K]	.24	.34	.2	.41	.35	.24	.49	.51	.20	.36	.57
[T | P, ¬K]	.44	0.44	0.45	0.45	0.45	0.44	0.47	0.47	0.47	0.48	0.50

Fig.5.4.3(b): Reliability for combination between method 'statistic' and 'inductive'

Similar work is carried out for the method 'inductive' and 'stereochemistry' and the reliability at overlap locations is shown in

Fig.5.4.3(c). The difference in reliability between the overlap and individual method of 'stereochemistry' is similar, as in Fig 5.4.3(b). Thus the effect of the method 'inductive' to the single method 'inductive' or 'statistic' is not large. Perhaps this is contributed by the small number of locations being predicted for each rules in method 'inductive'.

	α1	α2	α3	α4	α5	α7	β1	β2	β3	β4	β5
[T | L, K]	.50	.58	.38	.80	.56	.76	.57	.75	.64	.17	.33
[T | ¬L, K]	.26	.34	0.0	0.0	.45	1.0	0.49	1.0	.60	.49	.54
[T | L, ¬K]	.44	0.44	0.45	0.45	0.45	0.44	0.47	0.47	0.47	0.48	0.50

Fig.5.4.3(c): Reliability for combination between method 'stereochemistry' and 'inductive'

5.4.4 Combination for three methods

In this section, the effect of adding rules from method 'inductive' at overlapped locations predicted by method 'statistic' and method 'stereochemistry', is evaluated. The result of their reliability is shown in Fig. 5.4.4. and can be compared with the reliability result in Fig.5.4.3(a). For alpha helix prediction where the reliability is 0.51, it shows improvement in four of the rules (α_2, α_, α_5, α_7) and slight decreases in rule α_3. For beta strand where the reliability is 0.58, all rules record improvement.

The overall result shows improvement with the additional rules from method 'inductive' at overlapping location predicted by methods 'statistic' and 'stereochemistry'. However, this result is overshadowed by the small number of the overlap locations being observed when the three prediction methods are combined.

	α1	α2	α3	α4	α5	α7	β1	β2	β3	β4	β5
[T | P,L, K]	.50	.61	.42	.67	.67	.65	.71	1*	.78*	.67	1*
[T | ¬P,L, K]	.35	.36	0.2*	0.41	.47	.26	0.64	.61	.26	.34	.67
[T | P, ¬L,K]	.51	0.51	0.4*	0.60	0.1*	0.54	0.46	0.67	0.52	0.41	0.17
[T | P, L, ¬K]	.51	.5	.51	.5	.5	.5	.54	.56	.54	.57	.57

Fig.5.4.4: Reliability when prediction location from three methods overlap

5.5 Conclusion

The precision for alpha helix prediction using method 'statistic' is found to be more than 0.55 while the precisions for beta strand and beta turn prediction have the potential to be improved further. The average reliability for all secondary structure prediction in method 'statistic' is higher than 0.4 and perhaps with proper rules can be further improved. The prediction result for method 'stereochemistry' is poorer than the prediction result in method 'statistic'. The reliability of the rules in method 'inductive' is less than 0.60 as stated for the method. Most of the rules have a reliability below 0.50. In general however, the screening technique and combination of predictive methods improves the reliability of individual method.

CHAPTER SIX
PREDMOLL Implementation

6.1 Introduction

In Chapter Two, the development of PREDMOLL as a tool for an intelligent system was discussed. PREDMOLL is equipped with generic knowledge sources to implement some of its control problems. Several user- defined knowledge sources for control problems are needed in the implementation. This Chapter is concerned with the implementation for protein structure prediction. The objective is: to show how PREDMOLL implements its control strategy in the problem-solving process as planned by the users.

To achieve the objectives, this Chapter discusses the implementation of PREDMOLL in a series of protein structure predictions. They are: a secondary structure prediction based on the method 'statistic'; secondary structure prediction based on combining several predictive methods; and finally protein structure prediction which includes secondary and tertiary structure prediction. Knowledge sources for domain and control problems for each prediction are discussed.

6.2 Secondary structure prediction by the method 'statistic'.

6.2.1 Motivation

In Chapter One, one strategy to improve performance in secondary structure prediction is to combine several existing predictive methods. Two established methods, one based on Lim[Lim74] and another based on King[King87] have been discussed. Results of their prediction on 53 proteins are shown in Chapter Five which include their reliability as combined methods. A method to be developed in

this work is based on the statistical approach and its knowledge sources are created. Throughout this thesis, this method is referred to as the method 'statistic'.

The main reason to discuss prediction for the method 'statistic' separately, is to show how secondary structure prediction is done by using several simple heuristic rules. The rules in this method are different from rules in the other two methods. This is because it does not require deep knowledge about the theory of biochemistry. Instead, it is based on a simple way to connect several potential locations for secondary structure. The technique is made possible with the development of uncertainty technique which provides the facility for making decisions. Indeed, prediction by this method has a lot of potential for further improvements by rule manipulations and protein classifications[Taylor84].

Results of secondary structure predictions for the method 'statistic' are based on a series of rules and conditions manipulations as discussed in Chapter Five. In this chapter, discussions are focussed on how those rules are implemented in PREDMOLL. Besides the discussions on domain rules, the implementation of those rules in the problem-solving is discussed in detail. These tasks are performed by knowledge sources for control problems which are the central issue in this Chapter. Several user-defined knowledge sources required to solve these control problems are discussed.

6.2.2 Knowledge Sources For Domain Problems

6.2.2.1 Levels And Attributes

Domain problems for secondary structure prediction are divided into four levels of abstraction: *primary, island, second_certainty* and *secondary*. Groups of amino acids in each level are represented by different predicates such as: 'group_prop' for level *primary*,

'group_temp' for level *island*, 'group_certainty' for level *second_certainty* and 'group_second' for level *secondary*. The representation of different groups at different levels is to provide a unique representation at each levels. Furthermore, the creation of knowledge sources are also made easier. Each predicate for these groups has similar attributes as follows:

```
*********************************************************
Start          :  location for beginning;
End            :  location for end;
Struct         :  secondary structure adopted;
KSAR           :  KSAR that created the group;
Uncertainty/   :  uncertainty for Struct or
 'island'      :     identification of 'island';
Status         :  the status of the group,i.e 'active' or 'inactive'.
*********************************************************
```

Fig:6.2.2.1(a): Attributes for a group of amino acids

As an example, if the group is assigned as 'active' for its attribute 'Status', then the group can actively participate in the problem-solving process. However, when the attribute of the group is 'inactive', then it is no longer useful in the solution of the problem-solving.

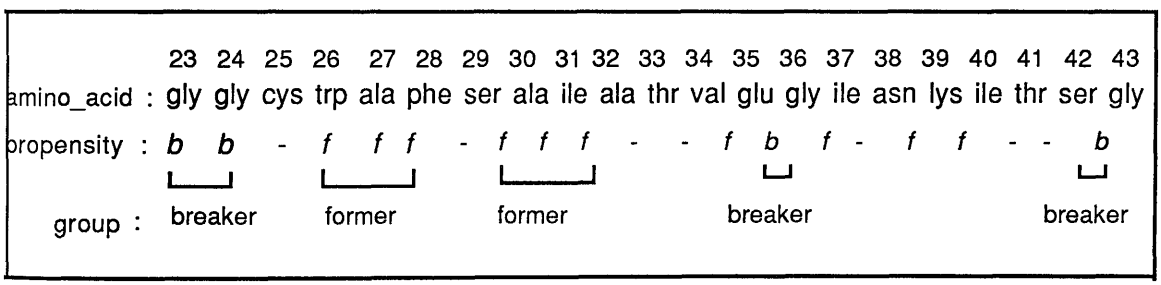
Attributes of KSARs are essential to provide means to refer to the actual KSAR involved in the process. It provides the facility for retracing the reasoning process on how the solution is achieved. The minimum attributes of KSARs for domain problems in this prediction are as follows:

```
*********************************************************************
attribute       argument          description
ksar_ks                           knowledge source for that KSAR
KSAR_group      [No,Start,End]    location for the group, 'No' is the
                                  identifier of the group.
KSAR_struct     [No,Struct]       Struct for the group
KSAR_certainty  [No,Certainty]    Certainty for the group
KSAR_ksar       [No,ksar]         ksar from the group triggered.
*********************************************************************
```

Fig.6.2.2.1(b) : Attributes for KSAR in domain problem

6.2.2.2 knowledge sources for Level *primary*

This is the lowest level of abstraction of domain problems. It consists of the primary sequence of amino-acids as the input. In prediction for the method 'statistic', there is only one knowledge source at this level. Its function is to 'create islands', that is to hypothesize possible locations of secondary structures in the sequence. The values to be used are based on 'propagation factor' for secondary structures as shown in Fig.4.5.5. At least three consecutive values greater than 1.00 are needed to form a potential structure for the location (island *former*). On the other hand, any amino acid having value less than 0.4 is considered as *breaker* for the structure. The knowledge source 'create island' deals separately for each of secondary structures. An example of the output for the secondary structure alpha helix is shown in Diag. 6.2.2.2.



Diag 6.2.2.2 : Island for secondary structure alpha.

In the blackboard, the result is represented in the form of group 'group_prop' as follows:

```
**************************************
group_prop(23, 24, alpha, KSAR, breaker, active).
group_prop(26, 28, alpha, KSAR, former, active).
.....
.....
**************************************
```

The fifth argument in predicate 'group_prop' is the identification of 'island'. This is because, the uncertainty value is not useful as yet at this level.

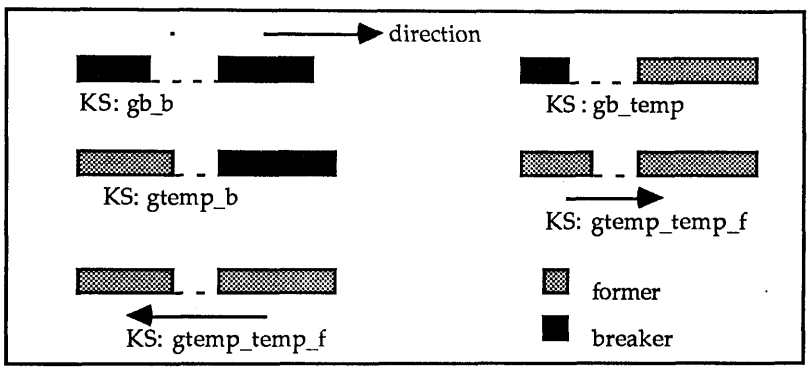
6.2.2.3 knowledge sources for domain level *island*

The objective of this level is to connect 'islands' and determine the most likely boundary of its secondary structure. Several knowledge sources are assigned to perform the tasks.The first task to perform at this level is to change the group of amino acids from 'group_prop' to 'group_temp'. The following tasks are based on the predicate 'group_temp' and when all the tasks at this level are completed, the uncertainty value replaces the identity of 'island'(*former*). The uncertainty value is used to determine the strength of belief for the locations to adopt a secondary structure.

Several knowledge sources are used to deal with different situations of connecting 'islands'. Five of the basic situations in connecting 'islands' are shown in Diag.6.2.2.3. The sign of direction in the diagram indicates the direction to be carried out in determining the boundary of the secondary structure by the knowledge source. The creation of knowledge sources and their name are based on their situations as shown in the diagram. Further extension is based on the following criteria:

a) the secondary structure for the islands to be connected;
b) number of spaces to be connected;
c) fulfillment of any conditions before the islands can be connected;
d) the islands and secondary structures surrounding them.

If the combination of criteria are taken into account, the number of knowledge sources that can be created are very large. However, larger numbers of knowledge sources created for these tasks can possibly produce a better precision for prediction. This is because it provides more a specific situation for connecting those 'islands'. This is shown in the analysis of the method in Chapter Five. In that analysis, three knowledge sources are added to provide a specific condition to separate alpha helices from other secondary structures. As a consequence, there are improvements in the overall result of the prediction.



Diag. 6.2.2.3: Problems to determine boundary

6.2.2.4 knowledge sources in level *second_certainty*

In secondary structure prediction for the single method 'statistic', there is no important task at this level except to change the predicate for groups of amino acids. This is because the information stored at level *island* uses the predicate 'group_temp' while at level *second_certainty* the predicate to be used is 'group_certainty'. The knowledge source responsible for this task is called 'temp_to_certainty'.

6.2.2.5 knowledge sources for Level *secondary*

The purpose of knowledge sources at this level is to choose the most possible locations for secondary structures among the groups of

amino acids at level *second_certainty*. Once a group is decided, it is posted to this level. Locations are stored in the predicate 'group_second'. When all locations for secondary structures have been decided, locations and their secondary structures are sorted and stored as a list in an appropriate file for further use such as for displaying the result, comparison and others.

Several knowledge sources can be created at this level. However, there are two important knowledge sources which are prominent at this level. They are used to make a decision on the possible groups to be selected for two important and separate cases. The first case is for structures with their locations are not overlapped with other locations of different structure. The task is carried out by a knowledge source called 'isolate'. It checks the uncertainty for the locations and if its value is greater than the cutoff value (or any other conditions laid down by the user as discussed in Chapter Five), then the secondary structure for the location is adopted.

In the second case, a knowledge source called 'highest_overlap' is used to deal with overlapping locations with different secondary structures. In order to select one of the overlapping groups, the knowledge source registers all the groups into an agenda. Depending on the policy of the problem-solving at that time, a group is chosen. Once a group is selected, the overlapping groups are no longer valid. The situation is shown based on Fig.6.2.2.5 as below:

****

policy: highest value of uncertainty

Group	Uncertainty	structure
A	0.7	beta strand
C	0.68	alpha
D	0.65	alpha
E	0.75	beta turn
B	0.8	beta strand

Selected:

Cycle 1: Group B

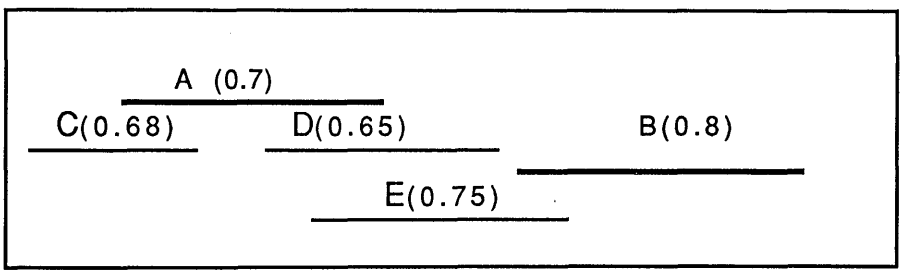
Cycle 2: Group A

****

Fig 6.2.2.5: locations with different secondary structure

In the example, based on the policy 'highest value of uncertainty', it chooses group B in the first cycle because its uncertainty value is the highest. In the second cycle, group E is no longer valid because it overlaps with group B. So group A is chosen because its uncertainty value is the highest among the remaining groups.

There is another knowledge source called 'overlap_turn' at this level to predict beta turn in a location between two closer beta strands. It is accepted even if they are overlapping. The situation is shown between group A-E-B where group E is hypothesized to be beta turn. If one of policies at this level is 'turn_between_beta' which prefers this situation, then group E is immediately selected after the groups A and B.



Diag 6.2.2.5: Highest Overlap

6.2.3 Knowledge Sources For Control Problems.

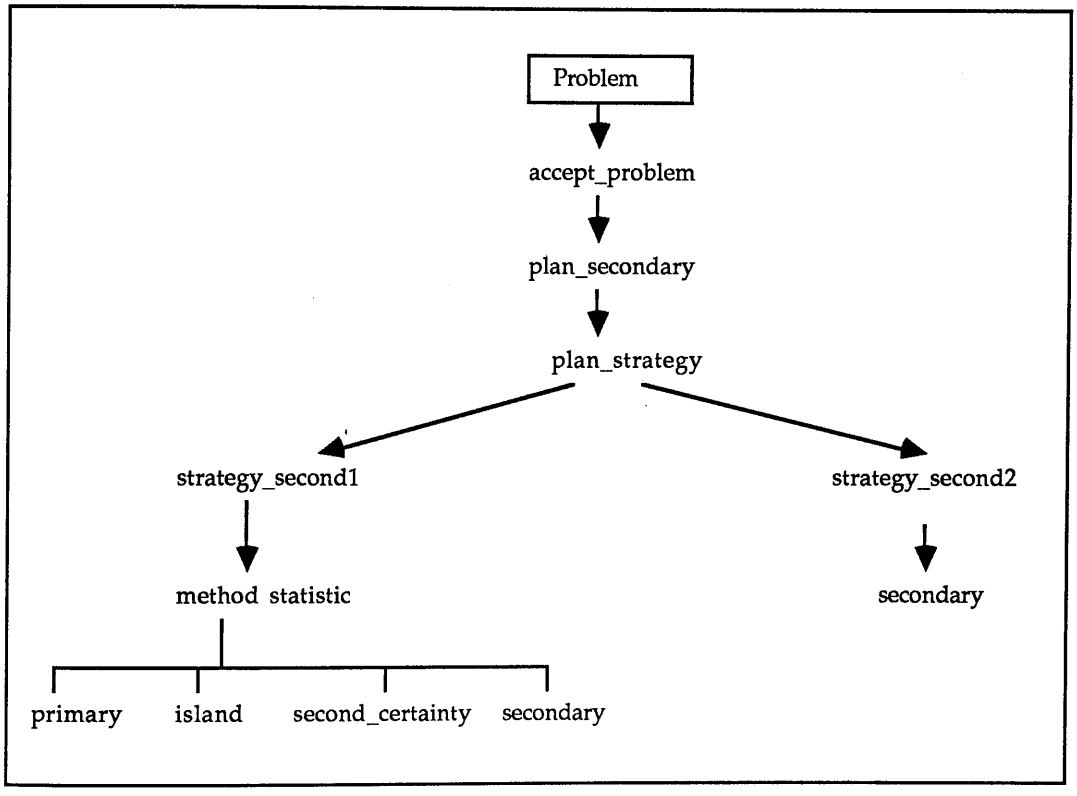
6.2.3.1 Approach to the problem

An approach to solve domain problems is from bottom to top. This is done by completing all knowledge sources for domain problems at the lower level before knowledge sources at upper levels are consulted. There are two strategies to be used for this prediction. For the first strategy, the prediction is performed by executing all relevant knowledge sources from level *primary* until level

secondary. At the end of the first strategy, secondary structure are predicted at all locations. The second strategy is employed once the first strategy is completed. Its objective is to store predicted secondary structures and amino acids as lists. Policies to be adopted throughout the problem-solving are:

a) prefer the most recently triggered knowledge source;

b) policy to calculate rating value is based on 'integrate_sum_weight'.



Diag.6.2.3 : controls to perform secondary structure prediction by a single method

The above approach is shown in the Diag.6.2.3. There are two strategies to be used and are performed in a sequence, i.e. one after another. Focus decisions for strategy 'strategy_second1' are also planned in a sequence. As most of the control decisions are planned in a sequence, they can be performed easily by generic knowledge

sources provided in PREDMOLL.

6.2.3.2 User-defined control knowledge sources

Although several control decisions can be performed by several generic knowledge sources provided by PREDMOLL, other knowledge sources have to be provided by users. These knowledge sources are usually involved in planning and lay out strategies to solve the problem. Besides that, several knowledge sources at level policy have to be developed. In this section, several important user-defined knowledge sources are discussed:

a) Level *plan*

'plan_secondary' is the only knowledge source defined at this level. It is triggered by an event 'accept problem predict secondary'. In the action part: it installs 'predict secondary' as the current problem and sets criteria for its termination; and plans a sequence of the strategies to solve the problem. The action part of the knowledge source is shown as follows:

```
**********************************************
Action:
        current problem: predict_secondary
        criteria_problem:
                complete all strategies,
        strategy_plan:
                first strategy: strategy_second1
                last strategy: strategy_second3.
**********************************************
```

Fig.6.2.3.2(a): action part of KS 'plan secondary'

b) Level *strategy*

Two knowledge sources for the two strategies as planned in 'plan secondary' are defined by users at this level. Summary of the action

part of both knowledge sources are discussed below.

```
**********************************************************
Action:
        current_strategy: strategy_second1
        criteria:
                complete all focus decisions,
        focus_plan:
                focus_series: strategy_second1,
                first_focus: method(statistic),
                last_focus:method(statistic),
        focus_plan:
                focus_series: method(statistic),
                first_focus: domain(primary),
                last_focus: domain(secondary),
                next_focus: one_up,
        focus_criteria:
                knowledge sources in the current focus are completed.
**********************************************************
```

Fig.6.2.3.2(b): knowledge source for 'strategy_second1'

In the action part of the knowledge source 'strategy second1', there are two 'focus plans' to be implemented. In the first focus plan, the strategy for 'strategy_second1' is to concentrate on the 'method(statistic)' to solve the problem. Another focus plan in 'strategy second1' is inherited from the decision to focus 'method(statistic)'. For 'method(statistic)', there are a sequence of focus decisions to be implemented, beginning with solving domain problems at level primary and ending up at level secondary. Criteria for completion of this strategy are the completion of all the focus decisions. For each the focus decision, criteria for completion are the absence of any knowledge source of decisions in the trigger list.

In the knowledge source 'strategy_second3', the strategy 'strategy_second1' has to have been completed before it can be scheduled. Its strategy is to focus at domain problems at level *secondary*. In contrast to strategy 'strategy_second1' where criteria for termination is simply based on the completion of its focus decisions, criteria for termination of 'strategy_second3' is based on some facts

generated in the domain blackboard. It terminates its strategy if there are facts of lists for predicted secondary structure and amino acid sequence.

```
*****************************************
feasible: strategy_second1 completed.
Action:
        current_strategy: strategy_second3,
        criteria:
                list_second_q(L),
                list_amino_q(L),
        focus_plan:
                focus_series: strategy_second3,
                first_focus: secondary,
                last_focus: secondary.
*****************************************
```

Fig.6.2.3.2 c: Part of knowledge source 'strategy_second3'

In both knowledge sources, definitions for several functions are provided. These definitions include the order of which the focus decisions are to be performed. For an example, in the knowledge source 'strategy_second1', the order of focus decisions is domain level *primary*, followed by level *island*, level *second_uncertainty* and finally level *secondary*.

d) Level *policy*

There are several user-defined and generic knowledge sources at this level. They influence the scheduling and the triggering process of problem-solving. As mentioned in Chapter Two, in contrast to decisions at other control levels, knowledge sources at this level can appear at any level. In the problem-solving process, their knowledge sources are categorized under 'independent', which means that they are executed along with the chosen action at that cycle. In this section, three user-defined knowledge sources for three different functions are shown:

```
**************************************************************************
trigger:
        event: accept_problem predict_secondary.
action:
        schedule_policy:  recently,
        criteria:
                problem predict_secondary completed.
**************************************************************************
```

Fig. 6.2.3.2(d1): Part of knowledge source ' recently'

In knowledge source 'recently', the definition for scheduling policy 'recently' is provided. It prefers those knowledge sources which are the most recently triggered to be given higher scheduling priority. Under the current implementation, knowledge sources which are triggered in that cycle are given 120 points, while knowledge sources which were triggered in the previous one cycle, are given 110 points. Beyond the previous one cycle, no point is given. Criteria for the policy knowledge source shows that its policy is operational until the problem-solving of 'predict_secondary' is completed. So, this policy provides a general scheduling policy throughout the problem solving for knowledge sources from control problems as well as from domain problems.

```
**************************************************************************
trigger :
        event: current_focus(strategy(S), method(M)) installed,
action :
        trigger_policy:  method M,
        criteria:
                current_focus(strategy(S),method(M)) completed.
**************************************************************************
```

Fig.6.2.3.2(d2):knowledge source ' trigger_method'

Knowledge source 'trigger_method' is triggered when the above current focus is installed at that cycle. It installs its trigger policy which requires the basic control to consult knowledge sources from domain problems which belong to the method 'M' only. In that case, if there is

a knowledge source which belongs to another method, its trigger conditions will not be consulted. The policy remains operational until the current_focus decision is completed.

```
**************************************************************************
trigger :
        event:      current_focus(strategy(strategy_second1),
                        domain(secondary)) installed.
action:
        schedule_policy:
            1. highest_certainty value,
            2. turn_between_beta.
        criteria:
            current_focus(strategy(strategy_second1),domain(secondary))
                completed.
**************************************************************************
```

Fig.6.2.3.2(d3): knowledge source 'policy_secondary1'

The policy 'policy_secondary1' is triggered to provide one of its scheduling policies during the current focus decision at level *secondary*. This is part of its implementation for strategy 'strategy_second1'. In its policy, it favours knowledge sources which predict a beta turn between two close beta strands; and locations with the highest uncertainty value. Those policies remain operational until the above current focus decision has completed. So, the policy decision in this example provides policies at a specific focus decisions.

6.2.4 Problem Solving Process

In this section, an example for control decisions during the problem-solving process is shown and described. The example is part of the prediction of secondary structure for protein 'b_act' by the method 'statistic'. The example is divided into two parts. The first part is to describe decisions to solve control problems while in the second part is to solve the domain problems.

6.2.4.1 Problem-solving for control problems

An example of problem-solving process for control problems is discussed. The discussion is concentrated at decisions taken during the beginning of the problem-solving process. Although these discussions are restricted, nevertheless, it shows how control decisions represented by knowledge sources are being implemented in PREDMOLL. The rest of control decisions for the problem-solving are similar to the one given below:

```
**********************************
cutoff value for certainty ?0.5
display rating:yes or no?yes
display_each cycle: yes or no ?yes

CYCLE :1 Protein:b_act
KSAR:2 KS:accept_predict_secondary R:100 [control, problem]

CHOSEN ACTION:2 KS:accept_predict_secondary

**********************************
CYCLE :2 Protein:b_act
KSAR:3 KS:accept_problem R:100 [control, problem]

CHOSEN ACTION:3 KS:accept_problem
INTENTION:to_solve_problem(predict_secondary)
REASON:current_focus(predict_secondary)
        EVENT:[accept_problem(predict_secondary)]

**********************************
CYCLE :3 Protein:b_act
KSAR:5  KS:policy_plan  R: 120 [control,policy]
KSAR: 4  KS:policy_recently  R: 120 [control,policy]
KSAR: 6 KS:plan_secondary R:120 [control, plan]

CHOSEN ACTION: 5 KS:policy_plan
INTENTION: policy(policy_plan)
        EVENT:[installed,current_policy(policy(policy_plan))].

CHOSEN ACTION: 4 KS:policy_problem2
INTENTION: policy(policy_problem2)
        EVENT:[installed,current_policy(policy(policy_problem2))].

**********************************
CYCLE :4 Protein:b_act
KSAR: 6 KS:plan_secondary R:210 [control, plan]

CHOSEN ACTION:6 KS:plan_secondary
INTENTION:plan_problem(predict_secondary)
REASON:problem_need_a_planning
        EVENT:[problem_has_been_planned(predict_secondary)]

**********************************

CYCLE :5 Protein:b_act
KSAR:8  KS: end_policy  R: 210 [control,policy]
KSAR:9 KS:plan_strategy R:220 [control, plan]

CHOSEN ACTION: 9 KS: end_policy
```

```
INTENTION: terminate_policy(policy(policy_plan))

************************************
CYCLE :6 Protein:b_act
KSAR:9 KS:plan_strategy R:210 [control, plan]

CHOSEN ACTION:9 KS:plan_strategy
INTENTION: plan_strategy(plan_secondary)
REASON:implement(plan_secondary)

**********************************
CYCLE :7 Protein:b_act
KSAR:10 KS:strategy_second1 R:310 [control, strategy]

CHOSEN ACTION:10 KS:strategy_second1
INTENTION:to_use_strategy(strategy_second1)
REASON:to_implement_plan(plan_secondary)
        EVENT:[installed,current_strategy(strategy(strategy_second1))]

**********************************
CYCLE :8 Protein:b_act
KSAR:12  KS:policy_strategy  R: 120 [control,policy]
KSAR:13 KS:strategy_focus R:320 [control, focus]

CHOSEN ACTION: 12 KS:policy_strategy
INTENTION: policy_strategy(strategy(strategy_second1))
        EVENT:[installed,current_policy(policy(strategy_second1))].

**********************************
CYCLE :9 Protein:b_act
KSAR:13 KS:strategy_focus R:310 [control, focus]

CHOSEN ACTION:40 KS:strategy_focus
INTENTION:to_predict_using(method(statistic))
REASON:to_implement(current_strategy(strategy_second1))
        EVENT:[installed,current_focus(focus([strategy(strategy_second1),method(statistic)]))]

**********************************
CYCLE :10 Protein:b_act
KSAR:17  KS:policy_focus  R: 120 [control,policy]
KSAR: 16  KS:policy_trigger_method  R: 120 [control,policy]
KSAR:15 KS:focus_focus R:320 [control, focus]


CHOSEN ACTION: 17 KS:policy_focus
INTENTION: policy(method(statistic))
        EVENT:[installed,current_policy(policy(method(statistic)))].

CHOSEN ACTION: 4 KS:policy_trigger_method
INTENTION: policy(trigger(method(statistic)))
        EVENT:[installed,current_policy(policy(trigger(method(statistic))))].

**********************************

CYCLE :11 Protein:b_act
KSAR:15 KS:focus_focus R:310 [control, focus]

CHOSEN ACTION:15 KS:focus_focus
INTENTION:complete_focus(domain(primary))
REASON:implement_focus(method(statistic))
        EVENT:[installed,current_focus(focus([method(statistic),domain(primary)]))]

**********************************
....policies
....and knowledge sources at level primary

**********************************
CYCLE :14 Protein:b_act
```

```
KSAR:25 knowledge source:end_focus R:120 [control, focus]

CHOSEN ACTION:25 knowledge source:end_focus
INTENTION:end_current_focus(domain(primary))
REASON:implement_focus_plan(method(statistic))
        EVENT:[focus_completed,current_focus(focus([method(statistic),domain(primary)]))]

**********************************
...policies
...
**********************************
CYCLE :15 Protein:b_act
KSAR:27 knowledge source:change_focus R:310 [control, focus]

CHOSEN ACTION:27 knowledge source:change_focus
INTENTION:change_focus_to(domain(island))
REASON:implement_focus_plan(method(statistic))
        EVENT:[installed,current_focus(focus([method(statistic),domain(island)]))]

**************************************************
```

Fig.6.2.4.1: Part of problem-solving process for control problem

Cycle 1-2

The control problem is initiated by a problem 'predict_secondary' being recorded on the control blackboard by a knowledge source 'accept_predict_secondary'. This event triggers a knowledge source 'accept_problem'. Since there is no other control knowledge source in the feasible list, 'accept_problem' is scheduled and executed. It records an event as 'accepting problem predict_secondary'.

Cycle 3-6

The event triggers knowledge sources 'policy_problem2', 'policy_plan', and 'plan_secondary' at Cycle 3. The first two knowledge sources belong to control decisions at level *policy* while the third one belongs to control decisions at level *plan*. Since under the implementation of PREDMOLL, decisions at level *policy* have priority higher than any other control decisions while their knowledge sources are categorized as independent, thus, both knowledge sources for control decisions at level *policy* are scheduled and executed at Cycle 3.

Events created by executions of those knowledge sources at

level policy trigger the generic knowledge source 'end_policy' for each policy. However, the knowledge source 'end_policy' is not feasible yet because its criteria for the termination of its policy are not yet satisfied. So, in Cycle 4, 'plan_secondary' is the only knowledge source for which conditions are satisfied. Hence, it is scheduled and executed. It records an event as 'problem has been planned'.

The event triggers generic knowledge sources 'end_problem', and 'plan_strategy'. The generic 'end_problem' cannot compete until the criteria for the problem 'predict_secondary' is satisfied. The event at the same time satisfies the criteria for policy 'policy_plan'. Thus, the knowledge source 'end_policy' for policy 'plan_policy' can now be scheduled. So, in Cycle 5, there are two knowledge sources which compete to be scheduled. The scheduler selects 'end_policy' because it is unconditionally higher than other knowledge sources at other levels in both blackboards. In Cycle 6, the knowledge source 'plan_strategy' is the only one which is feasible, so the scheduler schedules it and executes it. It records an event as favouring its first strategy, that is 'strategy_second1'.

Cycle 7-11

The event in Cycle 6 triggers a knowledge source 'strategy_second1' and is scheduled and executed. It plans a sequence of focus decisions and records the event of its execution as 'current strategy has been installed'. The event triggers knowledge sources 'end_strategy', 'strategy_focus' and 'policy_strategy'. Knowledge source 'end_strategy' is not feasible until the strategy 'strategy_second1' is completed. In Cycle 8 'policy_strategy' is scheduled and executed while 'strategy_focus' is executed in Cycle 9. Knowledge source 'strategy_focus' plans the current focus as 'method(statistic)' and records the event as 'current focus has been installed'.

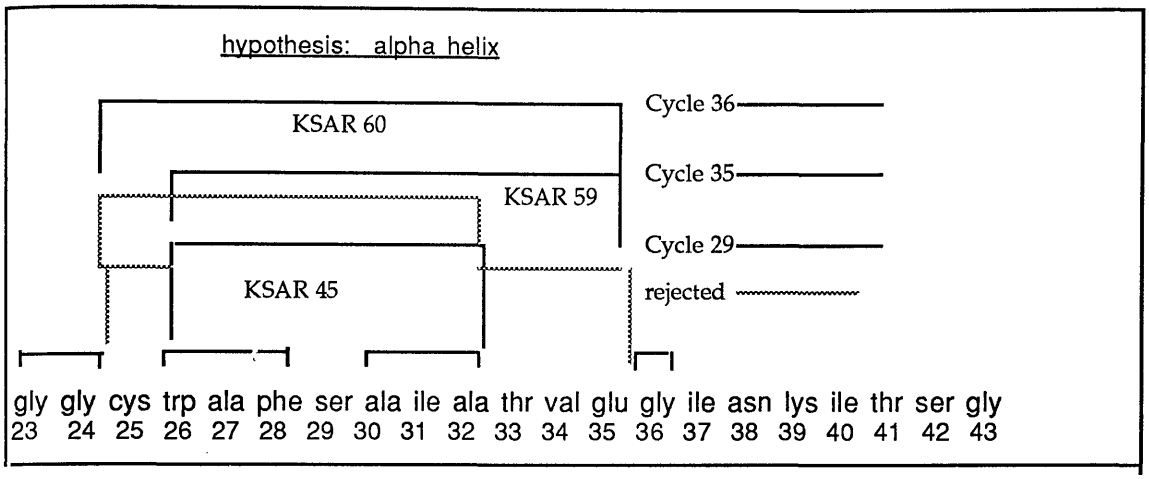
The event triggers generic knowledge sources 'end_focus' (is not be feasible until the focus complete) and 'focus_focus', and other several decisions at level *policy*. Executions at these knowledge sources are as shown in Fig.6.4.2.1.

Cycle 12-15

Several knowledge sources from domain blackboard are triggered. The scheduler selects a knowledge source which favours the current focus from the domain level *primary*. It schedules all knowledge sources from the level *primary* until there are no more knowledge sources at this level in the trigger list. This satisfies the feasible conditions of knowledge source 'end_focus' for focus 'domain(primary)'. Since this is the only knowledge source from control blackboard, it is scheduled and executed. The event triggers a generic knowledge source 'change_focus' and it installs a new focus 'domain(island)'. The process continues until the knowledge source 'end_problem' is executed.

6.2.4.2 Problem-solving for domain problems

In this section, a problem-solving process to solve the domain problems is presented. Since there are many levels and many cycles, an example is selected to solve a problem at location 23 to 43 of protein 'b_act'. The example uses several knowledge sources in several cycles at domain level 'island' to predict the secondary structure. In this implementation, in order to speed up the process and reduces the number of cycles, the trigger conditions of knowledge sources at level *island* stop looking other locations once there is a location that satisfies its condition. The process is even longer if trigger conditions of a knowledge source are allowed to search all locations that satisfy its conditions. Furthermore, many of these locations later on are rejected because some of them are no longer useful in the problem-solving process. The problem-solving process for the example is shown in Diag.6.2.4.2(a) and Fig.6.2.4.2(b).



Diag 6.2.4.2(a): Propagating islands.

A sequence of events for the example in Diag 6.2.4.2(a) is described in Fig 6.2.4.2(b). The cutoff value of the uncertainty is 0.5. If the location has an uncertainty value higher than the cutoff value for a particular secondary structure, then the structure is hypothesized for that location.

****

CYCLE :29 Protein:b_act
KSAR:45 KS:galpha_alpha R:420 [domain, island]
KSAR:44 KS:gtemp_b R:420 [domain, island]
KSAR:43 KS:gb_temp R:420 [domain, island]
KSAR:42 KS:gtemp_b2 R:410 [domain, island]
KSAR:41 KS:gb_temp2 R:410 [domain, island]
KSAR:39 KS:gtemp_b2 R:420 [domain, island]
KSAR:38 KS:gb_temp2 R:420 [domain, island]
KSAR:46 KS:prop_temp R:300 [domain, island]

CHOSEN ACTION:45 KS:galpha_alpha
Struct:alpha [30,32] [26,28]

****

CYCLE :30 Protein:b_act
KSAR:48 KS:gtemp_b R:420 [domain, island]
KSAR:47 KS:gb_temp R:420 [domain, island]
KSAR:46 KS:gtemp_b2 R:420 [domain, island]

```
KSAR:42 KS:gtemp_b2 R:410  [domain, island]
KSAR:41 KS:gb_temp2 R:410  [domain, island]
KSAR:39 KS:gtemp_b2 R:420  [domain, island]
KSAR:38 KS:gb_temp2 R:420  [domain, island]
KSAR:46 KS:prop_temp R:300  [domain, island]

CHOSEN ACTION:48  KS:galpha_alpha
Struct:alpha  [50,54]  [57,57]

**********************************
.....
.....

**********************************
CYCLE :35 Protein:b_act
KSAR:59 KS:gtemp_b R:420  [domain, island]
KSAR:58 KS:gb_temp R:420  [domain, island]
KSAR:56 KS:gb_temp R:410  [domain, island]
KSAR:54 KS:gb_temp R:300  [domain, island]
KSAR:50 KS:gb_temp R:300  [domain, island]
.........
.........
.........

CHOSEN ACTION:59  KS:gtemp_b
Struct:alpha  [26,32]  [36,36]

**********************************
CYCLE :36 Protein:b_act
KSAR:60 KS:gb_temp R:420  [domain, island]
KSAR:54 KS:gb_temp R:300  [domain, island]
....
....

CHOSEN ACTION:68  KS:gb_temp
Struct:alpha  [23,24]  [26,35]
ACTION SUCCESSFUL

**********************************
```

Fig:6.2.4.2(b): Problem-solving process for domain problems

Cycle 29-30

In Cycle 29, there are several knowledge sources that might be scheduled. However, the first three are the most recently triggered knowledge sources. Locations there are going to be connected are:

KSAR: 45 [26,28] - [30,32]

KSAR: 44 [30,32] - [36,36]

KSAR: 43 [23,24] - [26,28]

All those knowledge sources have similar ratings. By default, the scheduler chooses the first knowledge source in the list, i.e. 'galpha_alpha', to combine alpha_helix group 26 to 28 and alpha_helix group 30 to 32. The value of uncertainty from 26 to 32 is found to be 0.61, hence, it is greater than the cutoff value 0.5. So it connects locations from 26 to 32 as possible alpha helices. When those location are connected, other remaining tasks represented by KSAR 44 and KSAR 43 are no longer needed in the problem-solving process. So, theses tasks are rejected, eventhough they have not yet been scheduled. This is shown in the next cycle (Cycle 30) where those KSARs are no longer in the feasible list.

Cycle 35

In this Cycle, the two most recent knowledge sources enter the list. They are represented by KSAR 58 and KSAR 59 to connect the following locations:

KSAR 58: [23,24] - [26,32]

KSAR 59: [26,32] - [36,36]

At this cycle, the scheduler chooses KSAR 59 which causes knowledge source 'gtemp_b' to be executed. It attempts to extend the group forward with the maximum limit at amino-acid 35. Its strategy is to check the uncertainty value at amino-acid 35 and if fails then goes backward to amino-acid 34. The process continues until it reaches amino-acid 32. However in this location, the certainty value of amino-acid 26 to 35 is 0.63, thus, it is higher than the cutoff value of 0.5. So, the location of amino-acid number 26 to 35 is hypothesized to be an alpha helix.

Cycle 36

At this cycle, the knowledge source 'gb_temp' is chosen to extend backward the amino-acid group 26 to 35. It attempts to include amino-acid 25 into the group. The certainty value of amino-acid 25 to 35 is 0.62 which is higher than the cutoff value of 0.5. So, the location of amino acid number 25 to 35 is hypothesized to be an alpha-helix.

There are no more knowledge sources for an alpha helix at this location. The above example shows how several knowledge sources work together to predict a secondary structure at a particular location.

6.3 Secondary structure prediction by combination of methods

6.3.1 Motivation

In section (6.2), secondary structure prediction is based on the method 'statistic' only. In this section, two other methods are used in the prediction. Although most of knowledge sources remain as a single method, several knowledge sources for both domain problems and control problems are modified to suit these additional methods. Furthermore, several new knowledge sources are needed to implement new policies required to solve the problem of combining several knowledge sources from different predictive methods. This section discusses the modifications and new knowledge sources needed for solving the new problem at all levels in domain and control problems.

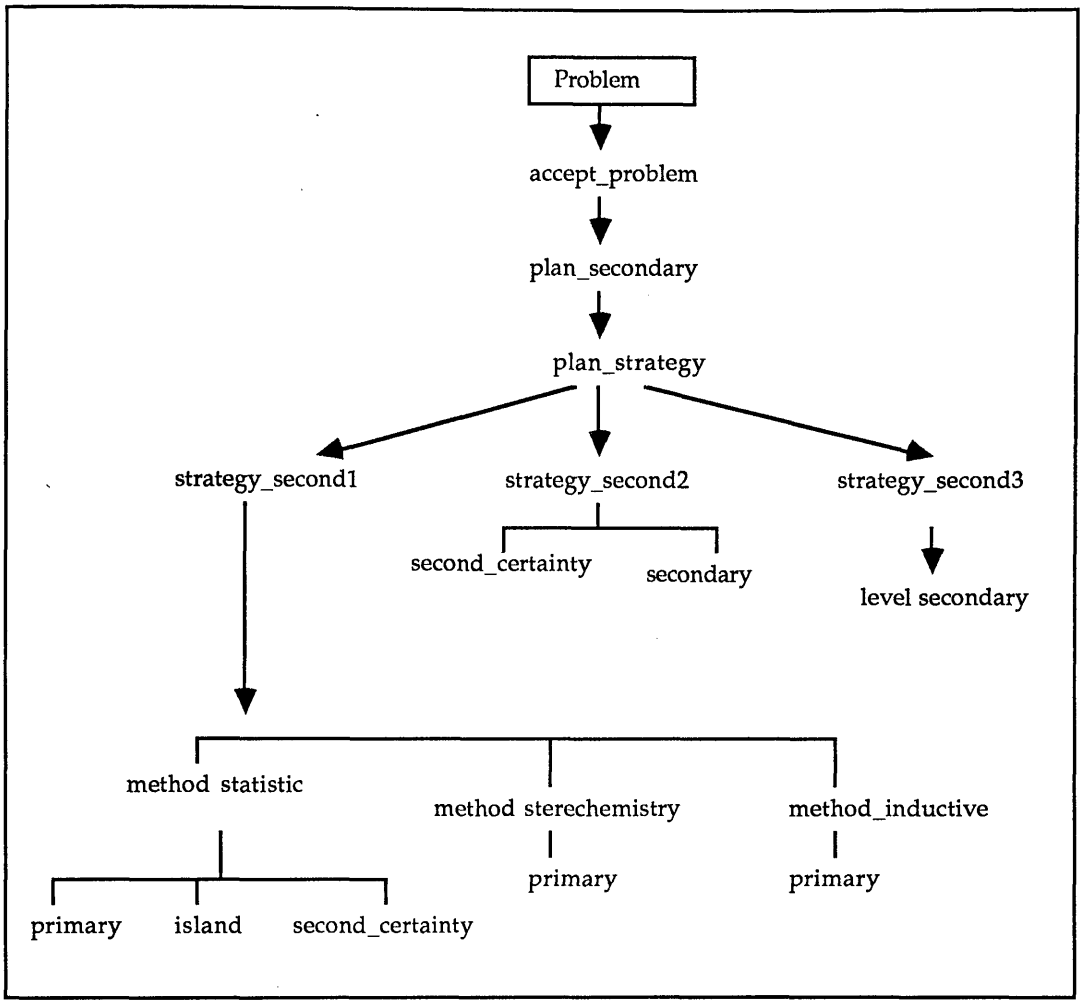


Fig.6.3.1: Controls to perform secondary structure predictions by several methods

6.3.2 Knowledge sources for domain problems

6.3.2.1 Level *primary*

There are additional knowledge sources at various levels for domain problems. At level *primary*, in addition to a knowledge source for the method 'statistic', there are thirteen knowledge sources for the method 'inductive', and one knowledge source for the method 'stereochemistry'. For the method 'inductive', its knowledge sources represent each of its thirteen rules: eight rules are for alpha helices and five for beta_strand. Hence, the rules are treated independently.

Locations predicted by knowledge sources of methods 'inductive' and 'stereochemistry' are posted to the level *second_certainty* while locations predicted by the method 'statistic' are posted to level *island*. Since tasks at level *island* involve outcome from method 'statistic' at the level *primary* only, hence, knowledge sources at the level *island* are the same as before.

6.3.2.2 Level *second certainty*

An additional knowledge source 'combine_method', is needed to combine the reliability of various groups. This happens when locations predicted by several predictive methods, are overlapping. Propagation is done in two ways:

a) both predict a similar structure;
b) predictions are in conflict.

In case (a), the new reliability for the structure is calculated as:

$$\text{New\_structure}(\alpha) = 1-(1-\text{Reliability1})(1-\text{Reliabilitiy2})$$
$$= 1-(1-p1)(1-p2).$$

For case (b):

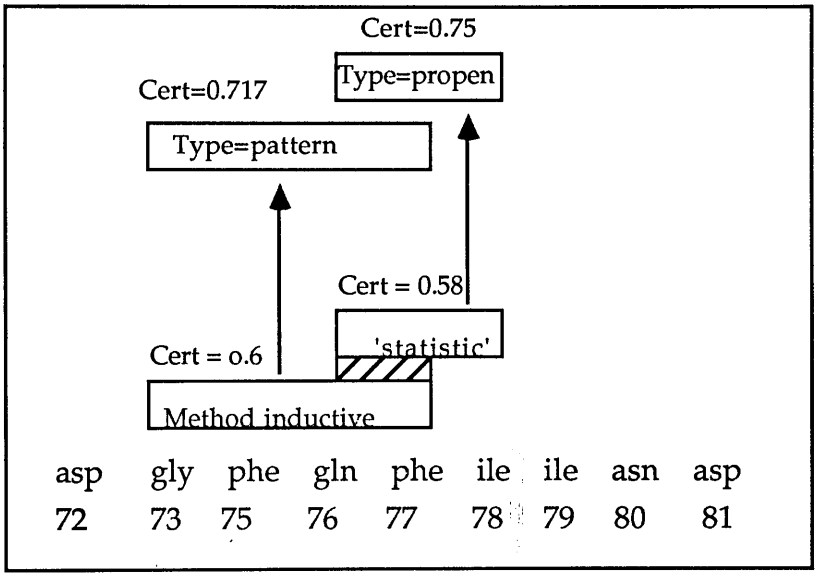
$$\text{New\_structure}(\alpha) = p1*(1-p2)/(1-p1*p2)$$

Under the current implementation, calculation of the propagation can be based on two types:

a) propagation of the reliability value using the D-S method is shown by equations for case (a) and case (b). However, under the current implementation, case(a) is the only appropriate. Reliability for each methods is provided and it is based on the result as discussed in Chapter Five;

b) reliability values for overlapping locations between two and three predictive methods are generated during the prediction. These values are based on the reliability at overlapping locations by several predictive methods as discussed in Chapter Five.

For both situations, the new reliability at overlapping locations are spread equally to the locations. This is done by calculating the average of the reliability at that locations. An example is shown as in Diag.6.3.3.2 as below:



Diag 6.3.3.2: Propagation for different type of sources

The reliability at overlap locations is:

Reliability(overlap) = 1 - (1-0.6)*(1-0.58) = 0.832

The formula for an average of reliability is:

average_reliability(Method) = (1-Ratio)*Old_value + Ratio*New_value

So, for each location:

reliability('inductive') = (1-0.5)*0.6 + (0.5)*0.832 = 0.717

reliability('statistic') = (1-0.67)*0.58 + (0.67)*0.832 = 0.75

The average of reliability is calculated for all overlapping locations. Thus, many averages are generated for each locations as a result of different overlaps for several methods. Under the current implementation, the highest value of the average reliability is chosen to represent the location to be used at level secondary.

6.3.2.3 Level *secondary*

Two additional knowledge sources are created for this level. The purpose of these knowledge sources are to extend the boundary of their secondary structures. The extension can be done in two directions: an extension on the left is done by the knowledge source 'extend_left'; while an extension on the right is done by the knowledge source 'extend_right'. This is as shown in the Diag 6.3.2.3 as below.

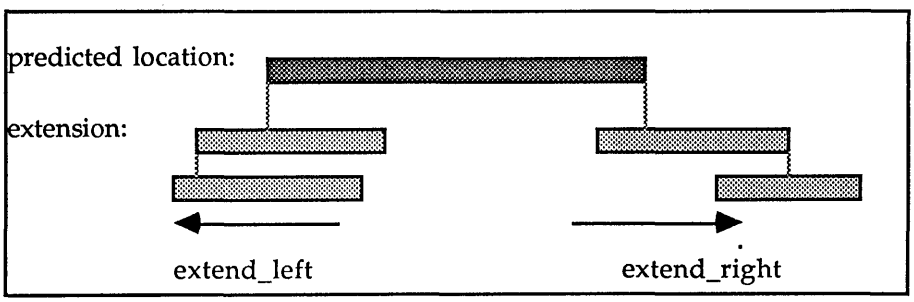


Fig:6.3.2.3: Extending boundary for predicted secondary structures.

6.3.3 Changes for knowledge sources for control problems

Several changes are made to existing knowledge sources for control problems. Furthermore, several new policies are added to provide the policy for the competition at level *secondary*.

6.3.3.1 knowledge source at level *plan*

As shown in Diag.6.3, there are three instead of two strategies to be implemented for this problem. Hence, in the knowledge source

'plan_secondary', the predicate 'strategy_planned' and criteria for the completion of the problem 'predict_secondary' have to be changed.

6.3.3.2 knowledge sources at level *strategy*

There are several changes in the knowledge source 'strategy_second1' besides a knowledge source 'strategy_second2' for an additional strategy in this prediction. In 'strategy_second1', the focus plan for methods 'stereochemistry' and 'inductive' have to be included. The focus decisions to be dealt by predictive methods in strategy 'strategy_second1' are as follows:

```
**********************************************
    Method 'statistic':
        1. primary;
        2. island;
        3. second_certainty.
    Method 'lim' :
        1. primary.
    Method 'king' :
        1. primary.
**********************************************
```

Fig.6.3.3.2: Methods in 'strategy_second1'

The knowledge source 'strategy_second1' is activated after the completion of the strategy 'strategy_second1' as planned by the knowledge source 'plan_secondary'. It implements a sequence of focus decisions to solve the problem at domain levels *second_certainty* and *secondary*.

6.3.3.3 knowledge sources at level *policy*

There are several knowledge sources involved. However, an important new policy is implemented for the strategy 'strategy_second2' at domain level *secondary* . The policy is to provide the weight between uncertainty and reliability for a location.

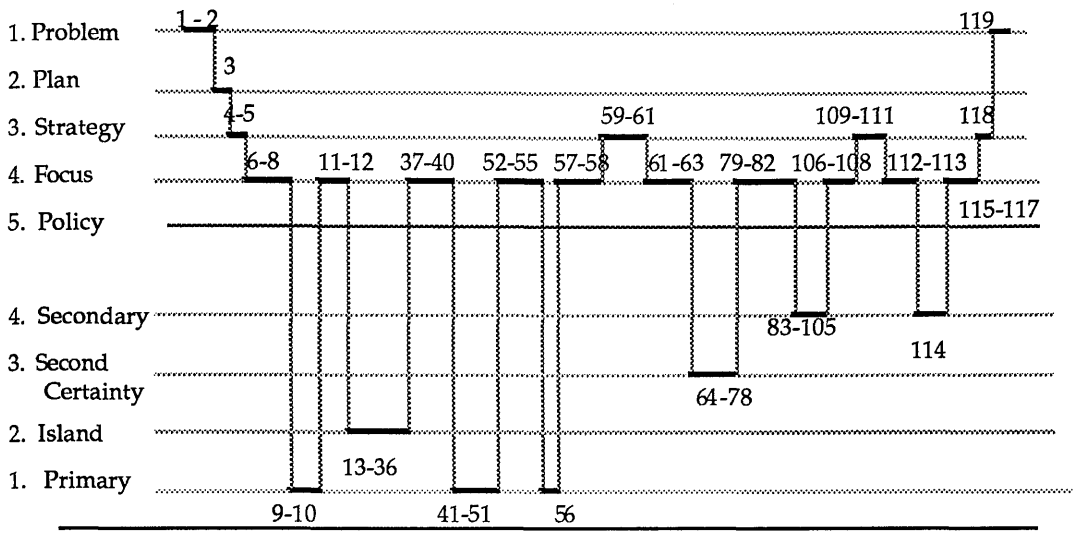
The average value is used as part of the basis to decide on locations to choose when they are overlapping. The action part of the knowledge source is as follows :

```
**********************************************************
    Action:
        schedule_policy :  highest certainty;
                        : highest reliability;
        weight_policy   : certainty is 0.65;
                        : reliability is 0.35.
        criteria        : complete level secondary for
                          strategy 'strategy_second2'.
**********************************************************
```

Fig:6.3.3.3: policy to deal with overlapping locations

6.3.4 Problem-solving process for 'combined methods'

Fig 6.3.4 shows actions that take place at various levels of the domain and control blackboard during secondary structure prediction for protein 'b_act'. In order to provide a smoother appearance of decisions at level *policy*, all policy knowledge sources are modified so that they are executed immediately without being scheduled. This is done simply by the action component of the knowledge sources put as part of the trigger conditions. They are executed once their trigger conditions are satisfied. Without the modification, the policy actions appear immediately after every higher level of control decision. The dotted line at the control problem level indicates the time interval for the decision to be implemented. In shows the time interval for three strategies used and their respective focus decisions. It also shows the time interval of the actions at the domain blackboard which corresponds to the focus decisions at the control blackboard.



Diag.6.3.3.4: Problem-solving process for secondary structure prediction for 'combined methods'

6.4 Tertiary Structure Prediction

6.4.1 Introduction

In Chapter One section (1.3.3), the difficulty of implementing tertiary structure prediction in this work was discussed. This is due to the difficulties in providing genuine rules to be used in the prediction. As an alternative, knowledge sources for tertiary structure prediction are generated, as discussed in Chapter Four. In the implementation, tertiary structure prediction is activated once the secondary structure prediction is completed. Hence, the predictions are implemented in sequence. This task is carried out by two control knowledge sources as follows:

```
**********************************************
knowledge source: plan predict structure
Action:
        current_problem: predict structure,
        criteria:
            list of secondary structures,
            list of tertiary structure,
            complete strategy,
        strategy_plan:
```

```
                    strategy_structure.

knowledge source: strategy predict structure
Action:
                current_strategy: strategy_predict,
                criteria:
                    complete focus decisions,
                focus_plan:
                    first: predict_secondary,
                    last:predict_tertiary
*****************************************
```

It should be mentioned that the difficulty in formulating a theory for uncertainty using homology technique remains. In this work, the uncertainty problem is resolved by primary sequence homology discussed in Chapter Four.

6.4.2 An approach to the problem

Domain problems are divided into two levels. The first level contains the possible tertiary structures with their uncertainty value. The second contains the best possible structures for a particular location derived from the competition between structures at the first level. For control problems, the first task is to control the amount of computer memory used in the entire problem-solving. This is due to a large number of knowledge sources for domain problems of the prediction. Without proper control, the amount of memory needed is very large. The second task is to provide the policies required to schedule knowledge sources at the domain blackboard.

6.4.3 Knowledge sources for control problems

Knowledge sources for control problems of the prediction have to solve those two tasks as mentioned in section (6.4.2). In solving the first task, the problem tries to find out the number of proteins available for comparisons. If the number is less than a certain amount, it adopt

'strategy_third1' while, if the amount is greater, strategy 'break_rules' is adopted. The 'strategy-third1' has similar strategy as in secondary structure prediction while the strategy 'break_rules' is different. Strategy 'break_rules' implements its strategy by: consults domain knowledge sources of each proteins where they were generated; executes them; and then delete all of them once they are no longer useful.

6.4.3.1 knowledge sources at level *plan*

A knowledge source 'plan_tertiary' is the only knowledge source in this level. It is triggered by an event 'accept problem predict_tertiary'. In the action part, it installs 'predict_tertiary' as the current problem and sets criteria for its termination. It plans a sequence of strategies based on the number of proteins to be used. The action part of the knowledge source is shown below:

```
**********************************************************
Action:
        current problem: predict_tertiary,
        criteria problem:
                complete strategy 'strategy_third1',
        calculate number of proteins: N,
        if N > 3
                plan_strategy:
                        first: break_rules,
                        last: strategy_third1,
        if N =< 3
                load knowledge sources from domain problems,
                plan_strategy:
                        strategy_third1.
**********************************************************
```

Fig.6.4.3.1: Action part for 'plan_tertiary'

6.4.3.2 knowledge sources at level *strategy*

The most important strategy at this level is 'break rules'. The purpose of the strategy is to overcome a memory problem during the problem-solving process. This is because there are many knowledge sources involved at level 'tertiary', thus, there is not enough memory to store all the knowledge source at one time. Strategy 'break rules' consults each protein at one time, triggers the conditions and finally, those which conditions are satisfied, their actions are executed. If there is no knowledge source for domain problems in the trigger list, then all of them are deleted. The next step is to consult the knowledge sources of another protein. Once the domain problems at level tertiary are completed, then it focuses problems at level 'tertiary1'.

```
**************************************************
Trigger:
        event : problem has been planned
        number of proteins: > 3,
Action:
        current_strategy: break_rules,
        criteria_strategy:
                no more proteins to be consulted,
                focus_completed.
        focus_plan:
                first: domain level tertiary,
                last: domain level tertiary1.

**************************************************
```

Fig:6.4.3.2: Strategy 'break rules' at level *strategy*

6.4.3.3 knowledge sources at level *focus*

There are several knowledge sources at this level. However, two of them are of interest in this section. They are used to implement strategy 'break_rules' when the current focus decision is at level *tertiary*. Part of these knowledge sources are shown in Fig.6.4.3.3. In knowledge source 'check protein', its action part is to load all files containing knowledge sources for domain problems at

level *tertiary*. In knowledge source 'erase rules', it deletes all knowledge sources for domain problems at level *tertiary* when they are no longer represented in the trigger list. This is done for two reasons: all their actions have been executed; they are not being triggered at all. The implementation by these two control knowledge sources is continued until there are no more proteins for which their domain knowledge sources need to be consulted.

****

knowledge source: check protein:

Trigger:

current_strategy: break_rules,
current_focus: domain tertiary,
its knowledge source is not in trigger list,
there is a protein P.

Action:

load all files which contain domain knowledge sources for protein P,
delete predicate protein P.

knowledge source: erase rules

Trigger:

current_strategy: break_rules,
current_focus: domain tertiary
there are rules for domain knowledge sources,
there are no KSAR in trigger list representing knowledge sources at level tertiary.

Action:

delete all knowledge sources of domain tertiary from memory.

****

Fig:6.4.3.3: Implementing 'break rules

6.4.3.4 knowledge source at level *policy*

Several policies are required to implement the way the problem is solved by this approach. A brief description of policies are as follows:

KS: trigger_tertiary - the policy is implemented throughout the

strategy 'break_rules'. the policy allows domain knowledge sources to be triggered.

KS: high_certainty - the policy is implemented at level tertiary1.

6.4.4. Knowledge source for domain problems.

Domain problems are divided into two levels. Level 'tertiary' contains the results generated by knowledge sources of tertiary structures. The results are stored in predicate 'group_tertiary'. In the second level, i.e. level 'tertiary1', there are three knowledge sources, i.e. 'highest_tertiary', 'exact overlap' and 'overlap_independent', to select the most possible tertiary structures. The process of selection is similar as in secondary structure prediction.

6.4.5 An example of tertiary structure prediction

Fig.6.4.5 shows the predicted locations of tertiary structure in the protein 'b_act'. The locations are based on the similarity of secondary structure pattern of the tertiary structure in the protein data base. The location in the protein 'b_act' is indicated by the first two arguments in the predicate 'group_tertiary', which are the protein's ordered secondary structure numbers.

****

	Location		Structure	KSAR	Uncertainty	
group_tertiary(	26,	28,	independent,	217,	0.15385,	active).
group_tertiary(	24,	26,	independent,	216,	0.09375,	active).
group_tertiary(	22,	24,	independent,	215,	0.16129,	active).
group_tertiary(	9,	11,	independent,	214,	0.12,	active).
group_tertiary(	24,	28,	single,	277,	0.11538,	active).
group_tertiary(	22,	26,	single,	276,	0.10526,	active).
group_tertiary(	24,	28,	single,	275,	0.11364,	active).
group_tertiary(	22,	26,	single,	274,	0.125,	active).
group_tertiary(9,		15,	single,	307,	0.094,	active).
group_tertiary(	9,	15,	single,	306,	0.094,	active).
group_tertiary(	24,	28,	single,	330,	0.11364,	active).
group_tertiary(	22,	26,	single,	329,	0.25,	active).
group_tertiary(	24,	28,	single,	353,	0.13158,	active).

```
group_tertiary( 22,  26,  single,     352,  0.0909,   active).
group_tertiary( 24,  28,  multiple,   376,  0.14286,  active).
group_tertiary( 22,  26,  multiple,   375,  0.15094,  active).
**********************************************************************
```

Fig:6. 4.5: Possible tertiary structure and their location at level 'tertiary'.

Considering the possible tertiary structures at several overlapping locations in Fig.6.4.4 if a policy to be used to select tertiary structure is:

a) the highest uncertainty value for overlapping locations;

then the problem-solving process is as follows:

Step 1

The group for location [22,24] and KSAR 215 is chosen. The location is hypothesized to be 'independent' with an uncertainty value 0.16.

Step 2

Groups where locations overlap with location [22,24] are no longer useful and their status is changed to 'inactive'. This is because they are not allowed to overlap with locations of tertiary structure 'independent'. The groups are those with KSARs: 216, 277, 276, 274, 329, 352, 375, 376. The actions are done in several cycles by a domain knowledge source 'overlap_independent' in level 'tertiary1'.

Step 3

In the next cycle, the group with location [9,11] and KSAR 214 is chosen. The location is hypothesized to be 'independent' with an uncertainty value 0.12.

Step 4

As a consequence of step 3, the following groups are no longer useful and their status are changed to be 'inactive':
KSARs: 306,307.

As a result of the policy at level 'tertiary1' for tertiary structure

prediction, two locations ([22,24] and [9,11]) are hypothesized for tertiary structure. The strength of the conformations is given by their uncertainty values as 0.16 and 0.12. Since there is no policy on a threshold for uncertainty value, both locations are accepted because their uncertainty value is the highest. However, the uncertainty values for both locations are too low and seem to be unlikely to be accepted if a threshold value is provided. This is reflected in the actual protein 'b_act' where those two locations do not form the structure as predicted. Instead two tertiary structures of 'multiple pattern' are the ones that occur in the protein.

6.5 Summary of the Chapter

In this Chapter, we have discussed briefly the implementation of PREDMOLL for a series of protein structure predictions. It is shown that a high percentage of the tasks in the implementation are performed by generic knowledge sources. Nonetheless, several user-defined knowledge sources are discussed and their roles are highlighted. Discussions of the performance of the implementation together with many aspects of this work are discussed in the next chapter.

CHAPTER SEVEN
DISCUSSION

7.1 Progress in the secondary structure prediction

7.1.1 Result of the method 'statistic'

The result produced from several heuristic rules (method 'statistic') developed in this work is quite encouraging for alpha helices, but not for beta strand and beta turn prediction. The overall precision of nearly 60% for alpha helices is considered to be acceptable. This is because the number of rules used are small and their principles are simple.

The uncertainty technique was used to give a formalism to determine the boundary of a secondary structure. It provides a robust technique to decide whether a sequence of amino acids adopts a particular secondary structure. It allows the creation of flexible heuristic rules. Furthermore, the threshold value can be adjusted to achieve an optimum result. The advantages of the heuristic rules can be summarized as below:

a) provides a robust technique in determining the boundary of secondary structure for a sequence of amino acid;
b) provides simple heuristic rules which are acceptable and easy to understand even for those without background in biochemistry;
c) provides the advantage of achieving a desired result by adjusting a threshold value;
d) provides the basis for future improvements. This is done by determining the class of the protein (proteins can be divided into several classes according to composition of its secondary structures

[Taylor84]) in the first stage, while in the second stage, probability values for the class is used in the prediction which is expected to produce a better result.

7.1.2 Advantages of the 'combined methods'

In Chapter Five, results based on a series of secondary structure predictions have been discussed. The results, which are based on the method 'statistic', showed further improvements when more rules were added. As suggested in Chapter One, instead of a single method, a combination is better for secondary structure prediction. Results of the prediction for the combination of method 'statistic', method 'inductive' and method 'stereochemistry' are shown in Fig.7.1.1.

During the process of prediction, whenever there is a conflict due to overlap at a particular location, the problem is resolved by deciding on the location with the highest uncertainty value. The uncertainty value is based on the 'relative probability' as discussed in Chapter Four. After secondary structure locations were decided using this technique, boundaries of the hypothesized locations are to be extended. In order to decide whether the new extended location is to be accepted or rejected, a similar technique of using the uncertainty value is employed. The new location is accepted if it satisfies the two conditions: its uncertainty value is the highest among the possible secondary structures; its uncertainty value is greater than the pre-determined threshold value.

	alpha combined	alpha thres 0.5	beta combined	beta thres.0.5
locations/actual	2277	2277	1548	1548
location/predicted	3666	2984	835	456
correct	1562	1348	365	221
precision	0.68	0.59	0.236	0.14
reliability	0.42	0.45	0.44	0.48

Fig:7.1.1(a): comparison for precision & reliability between a single method with threshold 0.5 and the combined methods

The result in Fig.7.1.1(a) is presented as the comparison between the combined methods and method 'statistic' with threshold value 0.5. The precisions for both alpha helix and beta strand increased by approximately 0.9 from 0.59 and 0.14 consecutively. Nevertheless, the reliability for both secondary structures is decreasing (alpha helix from 0.45 to 0.42 and beta strand from 0.48 to 0.44). However, the new reliability values are considered to be acceptable and perhaps can be further improved.

Another way of comparing the performance for the prediction is by comparing 'combined precision' value of individual methods. The value relates to the precision for predicting both alpha helix and beta strand together. Its calculation is derived from the previous result for alpha helix and beta strand as follows:

location/actual (alpha helix) = $T\alpha$
location/actual(beta strand) = $T\beta$
correct(alpha helix) = $C\alpha$
correct(beta strand) = $C\beta$

$$\text{combined precision(alpha/beta)} = \frac{T\alpha + T\beta}{C\alpha + C\beta} \qquad (1)$$

Using the calculation as in equation(1), values of 'combined precision' for several methods are obtained as follows:

For (α+β)	A:thresh 0.5 (5.2.1.2a)	B:add rules (5.2.1.2c)	lim[1] (5.2.2)	C:combined method
locations/actual	3825	3825	3825	3825
locations/predicted	3440	3723	4123	4501
correct	1569	1646	1391	1927
combined precision	0.41	0.43	0.36	0.503
combined reliability	0.456	0.44	0.33	0.43

Fig.7.1.1(b): The values of combined precision for several methods

Values of 'combined precision' and 'combined reliability' for several methods are shown in Fig.7.1.1(b). The first two methods are based on the method 'statistic' and their reference sections are shown in brackets. The third method is based on the method 'stereochemistry'. The fourth method is based on the combined methods as discussed earlier. Comparisons are made between methods A, B and C. All predictive methods are based on the method 'statistic' but with additional rules and conditions. For an example, method B is based on method A with additional rules for beta strands and beta turns. Method C is based on method B with additional rules of method 'stereochemistry' and method 'inductive'.

The result indicates that the 'combined precision' is increased when additional rules are added to the domain problems. This is shown by improvements of the 'combined precision' from method A to B (from 0.41 to 0.43) and from method B to C (from 0.43 to 0.50).

[1]The rules are not fully implemented

Thus, the improvement by combining methods is 7%. The results confirm the hypothesis stated in Chapter One that a combination of predictive methods improves the precision of secondary structure prediction.

7.1.3.Presentation of results

Results of several predictive methods are displayed by a program called DISPLAY. The program displays results of secondary structure prediction where the performance of different methods is observed. An example of the display is based on the prediction on protein 'b_act' as shown in Fig.7.1.3. It is evident that both the location, and even the secondary structure may vary with different predictive methods. Consider the location 25 to 42. This location is assigned as alpha helix in the protein data base. Both methods A and B predict alpha helices at location 25 to 35 only. This is because at location 36 there is the amino acid 'gly' which is considered anti-helical. The method 'statistic' avoids putting the boundary beyond this location which is an island 'breaker'. However, with additional rules from method 'inductive', the boundary is extended to location 39 because its 'relative probability' is the highest compared to other secondary structures. So the evidence that additional rules extend the boundary and improve the performance compared to a single predictive method is easily seen by the display facility.

```
**********************************************************************
Scale          :12345678901234567890123456789012345678901234567890123456789012345678901234567890

Residue    :LPSYVDWRSAGAVVDI KSQGECGGCWAFSAI BTVEGI NKI TSGSLI SLSEQELI DCGRTQ
Real       :        AAAA                      AAAAAAAAAAAAAAAAAA        AAAAAA TT
Thres 0.5  :                                  AAAAAAAAAAA                AAAAAAA
Thres 0.0  :                                  AAAAAAAAAAA                AAAAAAA
Lim        :AAAAA          BBBB  AAAAAAAA                        BBBB    BBBBBB
Combine    :                BBBB          AAAAAAAAAAAAAAA     BBBB  AAAAAAA

Residue    :NTRGCDGGYI TD GF QF I I N DGGI NTEENYPYTAQDGDCDVALQDQKYVTI DTYENVPYNN
Real       :TTTTTTT  AAAAAAAAAAA    BBB                  AAAA          BBBB
Thres 0.5  :                BBBBBB                      AAAAAAA  BBBBBB
Thres 0    :          BBB     AAAAAAA                    AAAAAAA     BBBB
Lim        ::   AAAAA          BBBB                   AAAAAAAAA   BBBBBB        AAA
Combine    ::         BBB        BBB                     AAAAAAA     BBBB        AAA

Residue    :E WALQTA VTYQPVSVALDAAG DAF KQYAS GI FTGPCGTAVDHAIVI VGYGTEGGVDYWI V
Real       :AAAAAAAA      BBBBB       AAAA                      BBBBBBBBBBTTTTBBBBB
Thres 0.5  :AAAAA     BBBB    AAAAAA AAAAAAAA  BBBB          BBBBBB            BBBB
Thres 0    :AAAAA     BBB     AAAAAA AAAAAAAA  BBB             BBBB            BBBB
Lim        :AA    AAAAA    AAAAAAA                AAAAA A  AAAAAAAAAAAAAA*AAAAAAAA
Combine    :AAAAAAA *BBAAAAAAAAAAAAAAAAAA*BB              BBBBB            BBBB

Residue    :KNSWDTTWGEEGYMRI.L RN.VGGAGTCGI ATMPSYPVKY
Real       :B             TTT*BBBB      TTTTTTTTTT    BBBB
Thres 0.5  :B       BBB       BBBBB.B
Thres 0    :        BBB       BBBBB.B    TTTT
Lim:       :AAAAAAAAAAAAAAAAAAAA                  AAAAA
Combine    :         BBB AAAAAAAAAAATTTT

Legend:
a) Single letter code for residue
        gly ala ser cys leu val ile met trp phe tyr thr his lys arg gln glu asn asp pro
         G   A   S   C   L   V   I   M   W   F   Y   T   H   K   R   Q   E   N   D   P
b) Single letter code for secondary structure
        alpha helix = A
        beta strand = B
        beta turn = T
**********************************************************************
```

Fig.7.1.3: Results Presentation

7.2 Problems in secondary structure prediction

7.2.1 Choice of the additional methods

The strategy employed in the present work to achieve better prediction is by providing additional rules for method 'statistic'. This is done by creating extra rules from this method and/or adding rules from other predictive methods. Although results presented so far showed an improvement in the results with additional rules by

combining methods, nevertheless they are lower than expected. One of the possible reasons is the choice of the additional methods. It was expected that the additional methods would provide additional locations and improve the reliability, insteads, their results are lower than expected. The method 'stereochemistry', which is based on a large number of rules and parameters, does not produce an expected result of precision between 0.50-0.55 [Taylor87] while the reliability for all rules of the method 'inductive' are less than 0.6[2]. Since the performance of these methods are not as claimed, the result of the combined methods is not as expected.

The poor result of the method 'stereochemistry' is possibly because the rules are not completely implemented in the prediction. However, the method 'inductive' is fully implemented in this work. Despite the claim that the reliability of the rules is greater than 0.6, similar rules for prediction but with a greater number of proteins in this work do not produce the stated result. The reliability for six of the rules (king_alpha_1, king_alpha_4, king_alpha_7, king_beta_1, king_beta_2 and king_beta_5) are around 0.5 only, while five others (king_alpha1, king_alpha_3, king_alpha_5, king_beta_3 and king_beta_5) are less than 0.45. Furthermore, two of the rules (king_alpha_6 and king_apha_8) do not produce any of the predicted locations in the fifty four proteins. Since the rules are simple and easy to implement, the validity of the prediction by this method in this work is satisfactory.

In evaluating the precision for the method 'inductive', rules are treated separately which will give the method an artificially higher reliability and precision. Even though there is an advantage as given in calculating the overall precision, the 'combined precision' for alpha helix and beta strand is 30%, which is low. The result is as follows:

[2]This is based on the result by King[King87].

	Alpha	Beta Strand
Total in protein	2277	1548
Predicted	1252	1568
Correct	575	586
Prob	0.25	0.36

Fig 7.2.19(a): Overall results using King's method

The figures in Fig 7.2.1(a) indicate that only 25% of alpha helices and 36% of beta strands are correctly located. These values would be lower if overlaps between locations were considered. An example is shown in secondary structure prediction for the protein 'b_act':

```
**********************************************************
Rules              Location          Secondary Structure

king_alpha_1    : [192-196]          alpha  helix
king_beta_3     : [195-198]          beta strand
king_alpha_7    : [191-195]          alpha  helix
king_alpha_4    : [195-200]          alpha  helix
**********************************************************
```

Fig.7.2.1: Example for overlapping for method 'inductive'

The example in Fig. 7.2.1(b) reveals two problems. The first problem is about overlaps between two groups with similar structures. This is shown by the locations found from the rules 'king_alpha_1' and 'king_alpha_7'. In this evaluation, the correct locations are calculated twice because the rules are treated separately, hence, it does not subtract the locations which overlap. The second problem is the selection of secondary structure as a result of a competition at overlaps. Again, this method does not provide a technique for solving this problem. If techniques for both problems are available in the method, the precision for the method is actually lower than the precision in

Fig.7.2.1(a).

7.2.2 Difficulty for beta strand prediction

Several workers [Robson79, Lim74] found that alpha helix prediction is relatively easy. This is also true in this work where the result for alpha helix prediction is consistently high. However, a similar performance is not found for beta strand prediction. Perhaps, this is because alpha helices are usually in longer sequences which makes them more consistently predictable even by heuristic rules, while beta strands are usually in short sequences. Furthermore, each beta strand requires another beta strand to form a beta sheet. Thus, beta strand prediction conceptually involves predicting a beta sheet, which results in the prediction being more difficult than expected.

Most workers employ a strategy for predicting beta strands at any location not predicted earlier to be alpha helices [Lim74, Robson79]. This strategy avoids the difficulty of developing a formula to deal with competition between secondary structures. This approach is not adopted in PREDMOLL, as the formula for uncertainty and other criteria are used in deciding the most possible secondary structure. The performance of beta strand prediction is low, where the highest precision achieved so far by the method 'statistic' with a modest number of rules is 0.21. However, the average reliability of prediction is around 0.45 which is considered acceptable under the circumstances. Additional locations predicted by additional methods increase the precision to 0.23 which is better than before but still not sufficient. As a result, the poor precision of beta strand prediction affects the outcome of 'combined precision'. Hence, additional rules for beta strands are required and perhaps a new approach should be initiated to improve precision.

7.2.3 Algorithm for assigning secondary structure

As mentioned in Chapter Four, a variety of algorithms is used in assigning secondary structure for the protein data base. As a consequence, variations in reliability for different method used in prediction are expected. When the assignment of the secondary structures in the protein data-base varies, the probability values or patterns derived from it are also expected to be different. Thus, two areas where secondary structure assignment affects the present work are as follows:

a) values for the propagation factor to be used in calculating the 'relative probability' for a sequence of amino acids to adopt a particular secondary structure;
b) patterns derived by the method 'stereochemistry' and method 'inductive'. In these methods, they are derived from the secondary structure definitions in the protein database. The patterns are generated based on these assignments. If the definitions are not similar, different results are expected.

7.2.4 Threshold Value

Several workers try to avoid the use of threshold values in their prediction systems [CohenFE86, Lim74,King87]. The reason for avoiding the value is because of difficulty in developing the uncertainty technique. Furthermore, the decision on the threshold value is itself a matter of investigation. As a consequence, rules generated from a method without uncertainty values are treated as certainties. However, in reality, the results are far from perfect. The errors of predicting secondary structure reflects the uncertainty in prediction. It can be concluded that as long as there is no credible theory that can be sufficiently predict the secondary structure, the threshold value, perhaps, could be used to provide the optimum result. This is shown when the screening method is applied to predicted locations as

discussed in Chapter Five. The result indicates improvements to its reliability.

7.3 Tertiary Structure Prediction

The tertiary structure prediction in this work is still at early stage of implementation. The work is destined for a long term project as many aspects of the requirement have to be developed. Nevertheless, the first stage of this work has started by providing a procedure for predicting tertiary structure. This includes the way the uncertainty problem in the prediction is tackled, the selection of the location for primary structure homology, and finally the use of secondary structure as a means for predicting tertiary structure.

In this work, several facilities have been provided for use in prediction. These include a program to perform inexact homology and the transformation of a secondary structure sequence into its position in the tertiary structure. Even though many facilities have been provided, difficulties remain in three aspects:

a) lack of rules for tertiary structure;
b) lack of techniques for handling uncertainty;
c) the reliability and precision of secondary structure predictions are poor.

The problem in (a) is artificially resolved in this work by generating rules based on a regular pattern of OGSS. Classifications of the structures, however, are used as a temporary measure for the purpose of implementing PREDMOLL. In reality, the genuine rules are more difficult to provide. Although tertiary structure is observed in the form of a 'Greek key'[Rawlings86], the likelihood that a secondary structure will form a particular tertiary structure is not known.

The problem in (b) is more serious. Criteria for uncertainty values to be used in predictions are difficult to generate (ARIADNE for example). In this work, a similarity value based on inexact primary structure homology is used as the basis for calculating uncertainty in this work. Nevertheless, difficulty remains in deciding on an uncertainty value as the threshold value for the competition. The difficulty in (c) is due to inherent weaknesses of the overall performance. A poor secondary structure prediction result produces poorer result for tertiary structure prediction.

An advantage of this work is due to the application of rule based systems in the problem. The system efficiently consults a large number of rules and at the same time manages the memory used.

7.4 Control Strategy in PREDMOLL

Developing a facility to present the problem in terms of a rule-based system is perhaps, the most significant contribution in this work. One of the objectives in this work was to provide such a system, as discussed in Chapter One. The features of the control strategy in PREDMOLL and its implementation for a series of predictions were discussed. In order to evaluate the performance of PREDMOLL, discussions are made on aspect of its implementation.

7.4.1 Overview for the implementation

a) Representing knowledge source

The knowledge is represented in the system based on a fixed format as discussed in Chapter Two. It contains the names of the rules, the problem, the location in the blackboard, trigger and feasible conditions, other scheduling variables and its actions appropriate to the current problem-solving status. A knowledge source for both domain and control problems has similar format which allows the users to represent it in a similar way: it explicitly specifies the conditions for it

to be scheduled and the action to be taken.

b) Planning control strategy

PREDMOLL solves problems using a standard control planning. So sub-problems and main problems are solved by uniform control planning. During an implementation of PREDMOLL, a sub-problem is transformed easily as the problem to be solved and the main problem easily coordinates the sub-problem 'predict secondary' and 'predict tertiary' in order to solve the problem 'predict structure'.

c) A small number of control knowledge sources are required

PREDMOLL is equipped with generic knowledge sources which are triggered once their conditions are satisfied. The generic knowledge sources were created to perform general control decisions for PREDMOLL. These include knowledge sources such as 'end policy', 'end focus', 'accept problem', 'strategy focus' and etc. A general way for PREDMOLL to solve its problem is when the control problems are planned as below:

- several sub-problems;
- a sequence of strategies;
- a sequence of focus decisions;
- hierarchical focus decisions.

Since PREDMOLL provides generic knowledge sources to control many of these complex tasks automatically, the number of user-defined knowledge sources required is small. This is shown in several predictions in Chapter Six where most of the required control knowledge sources are for planning the problem and strategy. For the focus decision, except for tertiary structure prediction where the knowledge sources are user-defined, all the focus decisions are planned in sequence, thus, they can be performed by generic knowledge sources.

d) Modular knowledge sources

Knowledge sources for both control and domain problems are made in such a way that they are independent from each other.

7.4.2 Intelligent Behaviour

In the previous sections, discussions focus on the features of PREDMOLL and its implementation. Beside that, it was mentioned that PREDMOLL should behave intelligently during a problem solving process. It is difficult to define what constitutes intelligent behaviour. However, a set of behaviour has been defined as characteristic of an intelligent system[HayesRothB85]. A brief discussions is given in relation to PREDMOLL:

1. Make explicit control decision

All control decisions in PREDMOLL are explicitly represented. In their knowledge sources, it is stated what actions have to be performed and when.

2. Decide what action to perform by reconciling independent decisions

In PREDMOLL, trigger conditions and feasible conditions reflect the desirability and the feasibility of actions to be performed. When trigger conditions are satisfied, it indicates that the action of the knowledge source is desirable. However, if the feasible actions are not satisfied, then it cannot be performed (desirable but not feasible). In one of the examples, strategies for problem 'predict secondary' are triggered by an event 'problem has been planned'. The control decision prefers a knowledge source from level *strategy* to be performed. A strategy 'strategy1' is feasible to be scheduled but 'strategy2' is only feasible once the 'strategy1' has been performed. So, strategy 'strategy2' is desirable but not feasible at that time.

3. Adopt variable grain-size control heuristic

The control decisions at level *policy* in PREDMOLL play important roles in adapting policies according to the current decisions for problem-solving. This includes adapting different weights for attributes at different states of problem-solving, and also the way the rating is computed for scheduling.

4. Adopt control heuristic that focus on useful attributes

Control heuristics in PREDMOLL always focus on useful attributes in the current problem solving. As an example, if the control decision is interested to perform the method 'statistic' at level *primary*, it first adopts a trigger policy to trigger only knowledge sources belonging to the method 'statistic'. Then it focus on the domain problems at level *primary*. Policy decisions install all relevant policies required to implement the control decisions of the problem-solving.

5.0 Adopt, retain and discard individual control heuristic

In the example in (4), once the criteria for the focus decision at domain level *primary* are satisfied, the control decision is discarded. Then, the system decides to change it to a new focus decision at domain level *island*. So, the focus decision for the method 'statistic' is retained along with a new adopted focus decision at the domain level *island*.

6.0 Plan a sequence of strategy action

PREDMOLL is equipped with a sequence of focus decisions at level *strategy* and at level *focus*. These control decisions provide a sequence of actions under appropriate circumstances.

The above six examples of behaviour cover the overall behaviours as required for an intelligent system as defined specifically for OPM[HayesRothB85]. It discussed how PREDMOLL exhibits such behaviour and thereby establishes itself as an intelligent system. Nevertheless, PREDMOLL exceeds this definition and surpasses the

behaviour of OPM. The comparison between PREDMOLL and OPM was discussed in Chapter Two.

7.5 Implementing PROTEAN by PREDMOLL control architecture

PROTEAN is an expert system used to predict protein 3-dimensional structure directly using NMR. The method is briefly described in Chapter One. As OPM, the system is implemented within BB1 blackboard architecture. In this section, we compare the behaviour of two systems in solving the same control problems, one is within BB1 architecture and another one uses control facilities provided in PREDMOLL. In this comparison no major changes are made to control knowledge sources specifically developed for PROTEAN except as mentioned in the discussion.

PROTEAN uses sixteen control knowledge sources to solve control problems. Twelve of them are domain specific while the other four are generic BB1 architecture control knowledge sources. The generic knowledge sources for BB1 and its equivalent in PREDMOLL are shown in Fig.7.5 as below:

BB1	PREDMOLL
initialize focus	strategy_focus
update_focus	change focus
terminate_focus	end_focus
terminate_strategy	end_strategy

Fig7.5: Generic knowledge sources in BB1 and its equivalent in PREDMOLL

The domain specific control knowledge sources are classified into: one at level 'strategy', two at level 'focus' and eight at level 'heuristic'. In this system, the levels 'strategy' and 'focus' have the same function as the levels with similar names in PREDMOLL while the knowledge

sources at level 'heuristic' have similar functions as in level 'policy' in PREDMOLL. The control plan for PROTEAN for its problem-solving process is shown in Fig.7.5(a) as below:

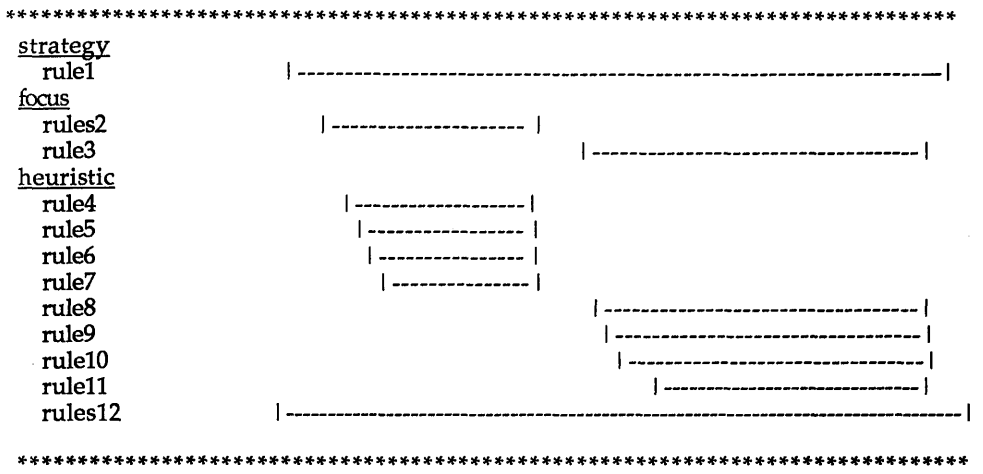


Fig7.5(a): Control Plan for PROTEAN

7.5.1 PROTEAN within BB1 architecture

Control problems are generally initiated by 'the problem' to be solved being recorded in the blackboard and a heuristic is triggered to be used throughout the problem-solving. At strategy level, PROTEAN records a decision to use strategy along with the information it needs to generate the prescribed sequence of steps. This step is recorded as individual decisions at the focus level, encompassing sequential problem-solving time-intervals. Each focus decision records information in which decisions are at the heuristic level.

In the above control plan, the heuristic rule12 is being operated for the entire problem-solving process. Focus decision 'rule2' is executed after strategy 'rule1' is installed. The action of focus decision triggers four heuristics which are associated with it: 'rule4', 'rule5', 'rule6' and 'rule7' . They are executed one after another as required in BB1 architecture. The creation of independent knowledge sources for heuristics in PROTEAN is an improvement to OPM (in OPM, the policies to be adopted by the scheduler are simply written in their

respective focus decisions). Since the principle in BB1 architecture requires each heuristic encompassed to be exactly the same time interval as its superordinate focus decision, thus, those heuristics are terminated at the same time as the termination of 'rule2'. Once 'rule2' and its associated heuristic knowledge sources are terminated, the generic knowledge source 'update_focus' will change the current focus to 'rule3' and the process continue as before until the strategy and the problem completed.

7.5.2 PROTEAN within PREDMOLL

The control problems in PROTEAN are solved in a similar way by PREDMOLL. Although the problem is considered to be easier for PREDMOLL compared to other control problems that have been implemented in this work, there are differences in its principle and implementation. The differences can be seen from Diag 7.5(a) and Diag 7.5(b). These differences are the result of different architecture that being adopted by PREDMOLL and can be described as follows:

a) criteria at level 'policy'

In contrast to architecture in BB1, PREDMOLL requires each knowledge source at 'policy' level ('heuristic' in PROTEAN) to specify its criteria for completion. By default, it will be terminated when the decision which created it is completed. In general, a policy decision is terminated one cycle after its superordinate decision. So, in the PROTEAN case, they are not terminated by generic 'end_focus' (i.e. together with their associated focus decision), but rather by their own generic knowledge source 'end_policy' which is provided by PREDMOLL. The generic 'end_policy' is triggered when a policy is installed but is not feasible until its criteria are satisfied. By having its own criteria, it can be terminated earlier or beyond the time-interval of its creator, which in this case is a focus decision. Furthermore, it is not only can be triggered by events in control decisions but also by any solution states during the problem-solving process.

As it is shown in PROTEAN, it is obvious that the capability of heuristic decisions is restricted within BB1 architecture compared with policy decisions in PREDMOLL. Although they provide different policies to be adopted by a scheduler at a different focus decision as in PREDMOLL, they have drawbacks in their implementation. This is because while the policy decisions in PREDMOLL have many important properties (such as flexibility in its conditions to be triggered and knowledge when to be terminated), heuristics within BB1 are triggered by their control decisions and do not have the criteria for termination.

b) Two stage scheduling

PREDMOLL provides two stage scheduling which has a big effect on the way it solves the control problems. In implementing PROTEAN in PREDMOLL, all heuristics are classified under 'independent', which allows them to be executed along with the chosen action at that cycle. Since decisions at level 'policy' have normally the highest priority, they are executed once their feasible conditions are satisfied. This is shown in Fig 7.5(b) where all heuristics associated with each focus decision are executed at the same time. This is in contrast to BB1 architecture where they are executed one after another and take four cycles to complete. PREDMOLL views the extra cycles as unnecessary because the interpreter is wasting time to consult rules which it knows that will not be executed prior to those heuristics.

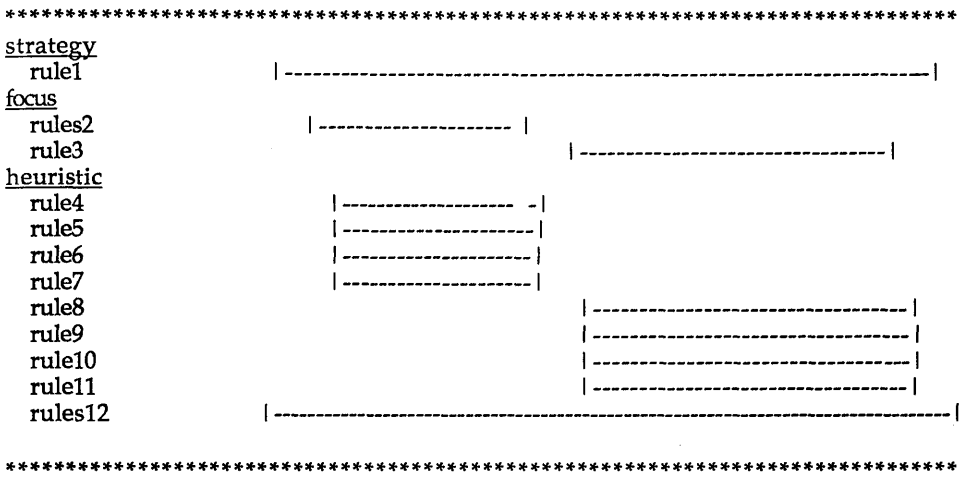


Fig7.5(b): Control Plan for PROTEAN within PREDMOLL

c) Others differences

Many other improvements are possible for PROTEAN if it were to be implemented within the PREDMOLL architecture. These involve the way it integrates rating for its scheduling mechanisms. As BB1 does not have the facility to deal with 'class' comparisons (without number: such as alpha helix has higher priority than beta strand) as in PREDMOLL, where values are required. In PREDMOLL, these facilities are provided under the second stage of scheduling mechanism. Further details of the comparison between the architectures are given in Chapter Two.

7.6 Uncertainty

Uncertainty technique, especially the one based on our method derived from the modified Bayes theorem played a distinguished role in protein structure prediction. None of the existing methods provides any clear idea about using uncertainty or provides any theory for it to be used in prediction. Two uncertainty methods are used for secondary structure prediction while a similarity value for primary sequence homology is used for tertiary uncertainty.

For the method based on the Bayes theorem, the conditional independence assumption is used. However, the effect is reduced to its minimum by normalization for every evidence added to it. The decision making is easily related to the consistency provided in this method. As a result, this method contributes significantly to the problem of maintaining consistency and reducing the effect of conditional independence assumption.

Although a significant progress is shown for reducing the effect on the assumption in the above uncertainty method, the uncertainty value by D-S technique is affected. For example, there is an overlapping location for the method 'statistic', 'inductive' and 'stereochemistry'. The values for the methods are given as follows:

p('statistic' | alpha) = 0.45

p(stereochemistry | alpha) = 0.52

p(king_alpha_1 | alpha) = 0.41

Σp(Method | alpha) = 0.45+0.52+0.41 = 1.38

Since the value of the total is greater than 1, it indicates that there is overlapping between the methods. This is shown by the calculation as below:

p(alpha | statistic,stereochemistry) = 1- (0.55)(0.48)
= 0.736
p(alpha | statistic,stereochemistry,king_alpha_1) = 1- (1-0.736)(0.59)
= 0.84.

However, the real value for that posterior probability is 0.5.

Fig. 7.7 shows a range of results for the rules in the method 'inductive'. It indicates that all the values are higher than the actual values. This indicates that the independent assumption is false. At the moment, there is no technique as yet to solve this problem for D-S method.

	α1	α2	α3	α4	α5	α7
propagation	0.84	0.86	0.82	0.87	0.84	0.86
p(H | M1,M2,M3)	0.5	0.61	0.42	0.67	0.67	0.65

Fig.7.7: Comparing the propagation values with the actual.

CHAPTER EIGHT

Conclusion and future research

8.1 Conclusions

In Chapter One, I stated my hypothesis that one of the ways to improve the precision of secondary structure prediction is by combining predictive methods. In order to test the hypothesis, three predictive methods were proposed: the first method('statistic') used heuristic rules created in this work (where there is not much knowledge about the subject); the second method ('stereochemistry') is based on a stereochemical approach where experts are needed to generate the rules; the third method('inductive') is based on an inductive learning approach where knowledge about chemical properties is needed.

Although the scheme appears satisfactory, the outcome of individual methods is less than expected. The 'combined precisions' (in percentages) is 43% for the method 'statistic', 36% for the method 'stereochemistry' and 30% for the method 'inductive'. Since, as stated elsewhere, rules for the method 'stereochemistry' are not completely implemented, the method does not perform as well as expected. Results for the method 'inductive' are disappointing because, not only is the precision poor, but reliability for its rules is lower than claimed(60%). The method 'statistic' shows resonable performance even with a small number of rules. The precision (43%) is almost comparable with that of the method of Chou and Fasman(45% [Taylor87]), although I am convinced that its combined precision can improve to a value of around 60%, if more rules can be created.

Despite the setback, a 'combined precision' for the combined methods yields 7% improvement (43% to 50%). Precision for alpha

helix prediction is 68% (from 57%) while that for beta strand is 23% (from 21%). The improvement confirmed my hypothesis, but with less precision that expected. However, without the disappointing result for the additional methods ('stereochemistry' and 'inductive'), more significant improvement is expected.

Precision of prediction is expected to increase further if more rules are created to predict individual secondary structures. Poor precision of beta strands affects the 'combined precision'. Additional rules for the secondary structure discussed in the section 5.2.1.2(c) comfirm this hypothesis. I conclude that there is room for further improvement for the 'combined methods'. The improvement can be achieved by adding more relevant rules from various sources to be used. Competition among secondary structures at overlapping locations can be dealt with by the uncertainty technique facility presented.

In tertiary structure prediction method, I criticize the basis for the uncertainty value used in the rules. Instead of an arbitrary uncertainty value in the tertiary structure rules, a similarity value for primary sequence homology is used. Locations where the homology has to be performed is similar as in ARIADNE, that is the location where the pattern of OGSS found. The technique of tertiary structure prediction in this work is still limited and a threshold value to be used as a technique for a decision-making criteria is not yet generated. The value is generated by investigating the variation of uncertainty values in the results of the tertiary structure prediction for all proteins. From this an optimum uncertainty value is generated to be used as the threshold.

The difficulties in predicting 3-dimensional structure from primary sequence has to be acknowledged. My ambition, stated in Chapter One, has had to be scaled down because experience reveals

that many aspects of the prediction need to be investigated and developed. I have listed the difficulties encountered during this work and some measures to overcome them. I conclude that the results of tertiary structure prediction are limited. Despite that, the groundwork for future work has been laid in the form of programs for inexact homology and calculation of the similarity value, and user-defined knowledge sources that provide a facility for coordinating comparisons with tertiary structures found in the protein data-base. As problems have been identified and facilities developed, further progress in the future is anticipated.

The progress of secondary structure prediction in this work contributes to the robustness of the uncertainty technique based on the Bayesian method. The uncertainty technique based on the Dempster-Shafer(D-S) evidence theory is not sufficiently encouraging to supplement the Bayesian method. The technique that is used for the singleton hypothesis was implemented as the experimental basis. The prediction result based on the Bayesian method alone is found to be better than in combination with the D-S theory, thus, I conclude that this combination is less useful for future work.

The 'relative probability' value of the modified Bayesian method provides a good decision-making facility to determine a secondary structure boundary for a sequence of amino acids of arbitrary length. The parameter is a good estimator for the actual conditional probability after it has been normalised with the total values for all existing hypotheses. The problem of conditional assumption is reduced to the minimum. The consistency for the propagation values is maintained and the decision making is done efficiently by this method. Hence, the technique is useful and should be used in future.

In Chapter One, I stated my intention to develop an intelligent system to facilitate protein structure prediction. Hence, PREDMOLL is

developed. PREDMOLL is compared to a range of systems such as HEARSAY-II, OPM and PROTEAN and it is concluded that the concept that I introduced in PREDMOLL is superior. PREDMOLL not only conforms to the intelligent behaviour as defined for OPM [HayesRothB85], but also provides more features. In fact PREDMOLL can be thought of as an extension to the existing features of an intelligent system as proposed by [HayesRothB85].

One of the weaknesses of the blackboard architecture is that it requires a large amount of computer memory to hold data, rules and new information generated during a problem-solving process. However, PREDMOLL showed how the control strategies are used to manage the amount of memory during the implementation of tertiary structure prediction. This is done by a user-defined control knowledge sources to load only relevant domain knowledge sources into memory at a particular focus decision. The domain knowledge sources are consulted and when they are no longer useful, a control knowledge source removes it from the memory.

I conclude that PREDMOLL achieves my intention of providing tools for an expert system environment for protein structure prediction. It has all the features stated in Chapter One, and some others. I point out that the present architecture with some refinements can be used beyond this application problem.

It can be concluded that this project has produced several tools and techniques for future progress in protein structure prediction. The future performance of prediction will depend on the rules for the domain problems as well as on the facilities provided for prediction. Success will demand the right combination of the two features.

8.2 Further Research

8.2.1 Combining with NMR data.

In the present work, the principle behind the technique is to combine predictive methods to achieve better precision. However, the methods are based on patterns of amino acid sequence. In the overview of the motivation for protein structure prediction, it was stated that a major motivation is to understand structure at the atomic level. In spite of the time elapsed since it first seemed that a formula might exist that would transform amino acid sequences to 3-dimensional atomic configurations, the results of secondary structure predictions are still low, let alone those for tertiary structure. Furthermore, even if a tertiary structure is predicted, its atomic configuration in 3-dimensional space still needs to be refined by a simulation technique.

It is my opinion that the present approach based on amino acid sequence as the only input, must have a limitation on the possible futher progress in determining 3-dimensional structure. Therefore, I propose that the present method be combined with a predictive method using data from NMR in future work. The advantage of predictive method using NMR data is that the 3-dimensional structure is achieved directly. PROTEAN is an example of a predictive method based on this technique but its implementation is limited to alpha helices only.

The main advantage when the two technique are combined is that they can be used to check one another. The first method is used to predict structure based on the pattern of amino acid sequence. This information would be used to support findings in the latter method, using an NMR approach. Once both methods confirm a structure for a location, the structure can be accepted. When the result consists of the atomic configuration of the protein in 3-dimensional space, it can be

shown by a graphic display.

9.2.2 Control Strategy In PREDMOLL

In the existing organisation of PREDMOLL, the full output showing how the result is achieved is not provided. However, during the execution of the problem-solving, competing knowledge sources are displayed with their rating at each cycle. The chosen action is shown with its location, event, structure, and others features depending whether the knowledge source belongs to control or domain problems. As the flow of problem solving can be traced easily, the need for a reasoning process does not arise. Furthermore, the problem at hand does not require such a facility.

The long term objective of PREDMOLL is to provide facilities for rule based systems beyond protein structure prediction. Thus, the facility for consultative and interactive systems are needed. Such extended features could be implemented based on the existing system. This is because each decision at each cycle is fully documented. The reasoning process is easily carried out by tracing back decisions taken at each cycle in the problem-solving process.

REFERENCES

[Appiah84] Appiah,A.,*Generalising the probabilistic semantics of conditionals,* Journal of Philosophical Logic 13 (1984).

[BaldwinI] Baldwin,J.F., *Fuzzy Sets and Expert System.*

[Baldwin67] Baldwin,J.F.,*support Logic Programming for Expert System,* ITRC 67.

[BaldwinII] Baldwin, J.F. and Pilsworth B.W. ,*And Inferential Fuzzy Logic Knowledge Base,* EM/FS31.

[Barr81I] Barr,A.,Feigenbaum,E., *The Handbook of Artificial Intelligence,* Volume I,(1981).

[Barr81II] Barr,A., Feigenbaum, E., *The Handbook of Artificial Intelligence,* Volume II,(1981).

[Blundell87] Blundell T.L., Sibanda B.L., Sternberg M.J.E., Thornton J.M., *Knowledge-based prediction of protein structures and the design of novel molecules,* Nature 1987.

[Bramer85] Bramer M.A., *A survey and critical review of expert system research,* 1985.

[Buchanan] Buchanan,B.G. and Duda R.O. *Principle in Ruled-Based Expert Systems,*

[Bundy84] Bundy,A, *Incidence Calculus: A Mechanism for Probabilistic Reasoning,* D.A.I.res.Paper no.216, Uni.of Edinburgh.

[Charniak85] Charniak E.,McDermott D.,*Introduction to Artificial Intelligence,* Addision Wesley, 1985.

[Chubb84] Chubb,D.W.J., *Knowledge Engineering Problems During Expert System Development,* SIMULETTER, July 1984, Vol.15, No.3

[Chou74] Chou,P.Y., Fasman,G.D. (1974) Biochemistry 13,222-245.

[CohenFE83] Cohen F.E, Abarbanel R.M.,Kuntz I.D., Fletterick R.J., *Secondary structure assignment for a / b proteins by a combinatorial approach,* Biochemistry 1983,22,4894-4904.

[CohenFE86] Cohen F.E., Abarbanel R.M., Kuntz I.D., Fletterick R.J., *Turn Prediction in Proteins Using a Pattern-Matching Approach,* Biochemistry 1986,25,266-275.

[CohenP85] Cohen,J.,*Describing Prolog by Its Interpretation and Compilation,* Comm. of ACM, Dec'85, Vol.28 no 12.

[CohenPR82] Cohen,P.R., Feigenbaum, E., *The Handbook of Artificial Intelligence,* Volume III (1982).

[CohenPR85] Cohen,P.R.,*Heuristic Reasoning about uncertainty: an AI approach,* Pitman Advanced Publishing Program,1985.

[Collin87] Collins, J.F., Coulson, A.F.W., Molecular sequence comparison and

alignment, Chapter 13,Synthesis and applications of DNA and RNA, Orlando, 1987.

[Colmerauer85] Colmerauer,A.,*Prolog in 10 Figures,* Communication of the ACM, December 1985, Volume 28, Number 12.

[Dietterich81] Dietterich T.G., Michalski R.S., *Inductive learning of structural descriptions,* AI 16,1981.

[Dietterich85] Dietterich T.G., Michalski R.S., *Discovering patterns in sequences of events,* AI 25, 1985.

[Duda79] Duda R.O., Hart P.E., Nillsson N.J., *Subjective bayesian methods for rule-based inference system,* Technical Note 124, AI Center, SRI, CA.

[Forsyth85] Forsyth,R., *The Architecture of Expert System,* Chapman and Hall Ltd, 1985.

[Friedland85] Friedland P.,Kedes L.,*Discovering the secrets of DNA,* ACM, November 1985.

[Frost86] Frost,R.A.,*Introduction To Knowledge Base System,* Collin Professional and Technical Books, 1986.

[Ginsberg8482] Ginsberg,M.L.*Does Probability Have A Place in Non-monotic Reasoning ?* ,Dec.1984, Stanford Heuristic Programming Project,Working Paper No.HPP-84-82.

[Ginsberg8431] Ginsberg,M.L.*Implementing Probabilistic Reasoning,* June 1984, Stanford Heuristic Programming Project, Report No HPP-84-31.

[Glymour85] Glymour C. , *Independence Assumptions and Bayesian Updating,* AI,Vol 25,1985.

[Gordon85] Gordon,J. and Shortliffe E.H, *A method for managing Evidential Reasoning in a Hierarchical Hypothesis Space,* 1985, Artificial Intelligence.

[HayesRothF84] Hayes-Roth,F. , *The Knowledge Base Expert System: A Tutorial,* IEEE, September 1984.

[HayesRothF85] Hayes-Roth F., *Rule-based system,* ACM, September 1985.

[HayesRothB85] Hayes-Roth,B., *A Blackboard Architecture for Control,* Artificial Intelligence 26 (1985).

[HayesRothB86] Hayes-Roth B.,Buchanan B.G.,Lichtarge O.,Altman R.,Cornelius C., Duncan B., Jardetsky O., *PROTEAN: Deriving Protein Structure from Constrainst,* AAAI Vol 5,1986.

[Jackson85] Jackson P., *Introduction to Expert System,* International Computer Science Series,1985.

[Jones86] Jones J.,Millington M.,Ross P., *A blackboard shell in Prolog,* ECAI,86.

[Kabsch84] Kabsch,W., Sander,C., *Dictionary of Protein Structure: Pattern Recognition of Hydrogen Bonded and Geometrical Features.* Biopolymers 22, pp 2577-2637.

[King87] King R.D., *An inductive learning approach to the problem of predicting a proteins secondary structure from its amino acid sequence,* Turing Institute, 1987.

[Lathrop87] Lathrop R. H.,Webster T.A.,Smith T.F.,*ARIADNE: pattern-directed inference and hierarchical abstraction in protein structure recognition,* ACM, November 1987.

[Lesk80] Lesk,A.M., Chotia,C.J.,*How Different Amino Acid Sequences Determine Similar Protein Structures: The Structure and Evolutionary Dynamics of the Globins,* Journal of Molecular Biology 136 (1980), 225-270.

[Lesser75] Lesser,V.R., Fennel, R.D., Erman,L.D., and Reddy,D.R.(1975), *Functionally accurate cooperative distributed systems,*IEEE Transactions on Acoustic Speech Signal Processing 23, 11-24.

[Lim74] Lim,V.I.,*Structural Principles of the Globular Organization of Protein Chains. A Stereochemical Theory of Globular Protein Secondary Structure,*Journal of Molecular Biology 88, 857-894 (1974).

[Lindsay80] Lindsay R.K.,Buchanan B.G.,Feigenbaum E.A.,Lederberg J.,*Applications of AI for Organic Chemistry: The Dendral Project,* Advanced Computer Science Series 1980.

[Liu87] Liu,X. and Gammerman,A., *On the validity and applicability of the inferno system,* Conference on Expert System 1987.

[Michalski83] Michalski R.S., *A theory and methodology of inductive learning,* AI 20,1983.

[Michie85] Michie D.,J.M.Richards,Muggleton S.A.,Reid G.,*Application of AI in Biotechnology,* Project Report,1985.

[Milner87I] Millner-White,E. James, *A recurring loop motif in proteins, with a characteristic hydrogen bond pattern,* 1987.

[Milner87II] Millner-White E.J., Poet Ron, *Loops, bulges,turns and hairpains in proteins,* TIBS 1987.

[Morffew86] Morffew,A.J., Todd, S.J.P., *The use of PROLOG as a protein qeuerying language,* Computer and Chemistry Vol 10, 1986.

[Naylor85] Naylor,C. *How to build an inferencing engine,* Chapman and Hall Ltd,1985.

[Neesham88] Neesham,C., *In search of model answers,* Computing July 88, page 20.

[Pednault81] Pednault E.P.D., Zucker S.W., Muresan L.V., *On the independent assumption underlying subjective bayesian updating,* AI,May 1981.

[Quinlan83] Quinlan,J.R., *A coutious Approach To Uncertain Inference,* Computer

Journal,Vol26,No3,1983.

[Rawlings86] Rawlings C.J., Taylor W.R.,Nyakairu J., Fox J.,Sternberg M.J.E.,*Using Prolog to represent and reason about protein structure,* Proc. 3rd. ICLP,86.

[Rawlings87] Rawlings C.J,*Analysis and prediction of protein structure using AI,* Proceeding 4th. ESCAMD, IBC Press 1987.

[Rawlings88] Rawling C.J., *Expert System in Molecular Biology,* IUSC Workshop,1988.

[Robson79] Robson,B., Osguthorpe,D.J., Journal of Molecular Biology 132 (1979), 19-51.

[Shortliffe76] Shortliffe,E.H. (1976) *MYCIN: Computer Based Medical Consultation,* New York: American Elsevier.

[Spiegelhalter84] Spiegelhalter,D.J. and Knill-Jones R.P.,*Statistical and Knowledge-Based Approaches to Clinical Decision-Support System, with an Application in Gastroenterology,* Royal Statistical Society,1984.

[Stefik81I] Stefik,M.,*Planning and with constrainsts (MOLGEN: Part I),* AI 16,1981.

[Stefik81II] Stefik,M.,*Planning and Meta-Planning (MOLGEN: Part II),* AI 16,1981.

[Sternberg83] Sternberg M.J.E., *The analysis and prediction of protein structure,* in Computing in Biological Science,Elseviour Biomedical Press,1983.

[Taylor83] Taylor W.R., Thornton J.M., *Prediction of super-secondary structure in proteins,* Nature VOL 301,February 1983.

[Taylor84] Taylor W.R., Thornton J.M., *Recognition of super-secondary structure in proteins,* Journal in Molecular Biology, 1984.

[Taylor87] Taylor,W.R., *protein structure prediction,* Chapter 12, Synthesis and applications of DNA and RNA, Orlando, 1987.

[Waterman76] Waterman,M.S., Smith,T.F., Beyer,W.A., Some Biological Sequence Metrics, Advances in Mathematics 20,(1976).

[White84] White A.P., *Inference deficiencies in rule-based expert,* Research and Development in Expert System, Cambridge Univ. Press, 1984.

[Wilmot88] Wilmot,C.M., Thorton, J.M., *Analysis and prediction of the different types of beta turns in proteins,* Dept. of Crystallography, Birkbeck College, London (1988).