Investigation of the Application of Constraint Logic Programming to Protein Structure Prediction

Darren L. Widenmaier

Supervisor: Anthony J. Kusalik

Abstract

Prediction of a protein's three-dimensional structure from its amino acid sequence is a 'holy grail' of molecular biology. Current methods have had limited success, and with the Human Genome Project expected to uncover 10^5 protein sequences, the need to quickly determine protein structures will only increase. This review presents the background of the biological application area to a computer science audience, and examines the current computational methods in this area. Finally, Constraint Logic Programming is examined and evaluated for its suitability to this application area, and is found to have many advantages that would warrant the implementation of a protein structure prediction system in this paradigm.

1 Introduction

Proteins are important molecules due to the crucial roles they play throughout biology. Proteins provide our bodies with structure and motion, act as biological messengers, provide immune protection, and control the biochemical reactions that give us life. The significance of DNA to biology is well known, but proteins (via the information encoded in their structures) are now considered the 'second half' of the genetic code, an observation which emphasizes their status, and places the quest to understand protein structure on the same level as the historic quest to understand the structure of DNA.

Understanding a protein's structure is the key to understanding its function; unfortunately, determining the three-dimensional (3D) structure is very difficult. Current biochemical experimental methods are tedious and do not work for all proteins, and as a result only about 1,000 3D structures are known. This slow progress, combined with the fact that there are about 26,000 known protein sequences (with the Human Genome Project expected to uncover the genetic information to code for 100,000 proteins), has lead to a great deal of interest in using computational methods to predict structure from sequence and other data that are available. However, there has been only moderate success in this approach, and the problem is currently computationally intractable.

This review examines the computational methods now in use, and explores the possibility of using a relatively unknown paradigm, that of Constraint Logic Programming (CLP), to attack the problem of predicting protein structure.

2 Application Area Background

Proteins are composed of structural units called amino acids. Amino acids are small organic molecules which share a common structure and are differentiated by a '**side chain**' or '**functional group**.'

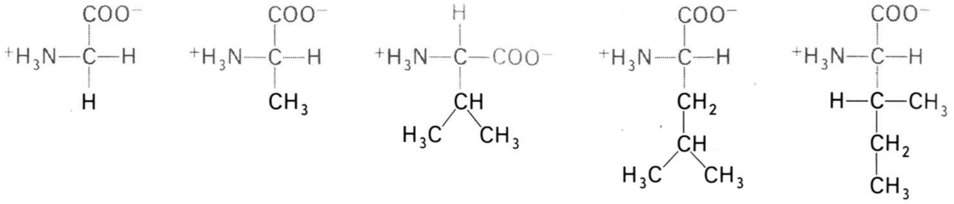

Figure 1: Structural formulae and space-filling models for five amino acids

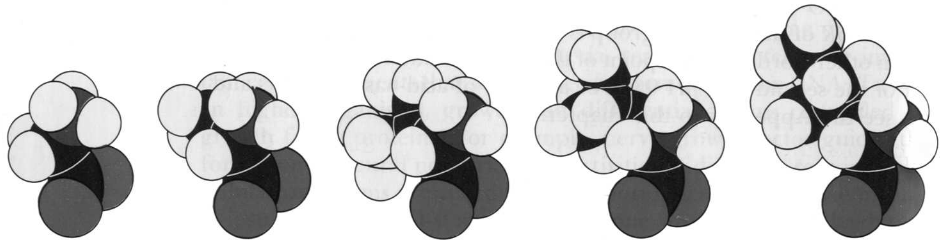

Figure 2: Fully extended protein chain

Figure 1 shows the structural formulae and space-filling models of five of the 20 amino acids that occur in proteins; the darkened portions in the structural formulae are the functional groups. Amino acids are linked to each other in linear chains via peptide bonds (leaving amino acid residues, or simply **residues**). In theory, any number of amino acids may be linked together, in any order, to form a protein. This is the source of the great variety of proteins that may exist in nature or be created in the laboratory.

The order of amino acid residues in a protein is called its **sequence** or **primary structure**. This may be visualized as a long, string-like molecule, with the common portions of the amino acids forming the backbone, and the side chains in this string distinguishing one protein from another [Figure 2]. This primary structure is relatively easily determined by simple chemical analysis of the protein, or, more commonly, by DNA sequencing (a DNA sequence that is known to code for a protein can easily be translated into a protein sequence). However, proteins rarely exist in this linear form under physiological conditions.

A protein in this linear form is **denatured**, and loses its biological effect; proteins must fold up into a specific 3D shape, or **conformation**, to function. This folding is driven by energy minimization: chemical and physical interactions between the protein and itself and its environment force the protein into a conformation that minimizes its total energy (minimizes unfavorable interactions) and is biologically stable. These interactions are dependent on the type and occurrence of the constituent amino acids and thus the information required to fully specify a proteins conformation is contained entirely within its primary structure (sequence), given a particular environment [2].

There are some key constraints to the folding of proteins to minimize energy. The peptide bond which connects adjacent amino acids is rigid and planar, and rotation about this bond does not occur [Figure 3]. There are two bonds (on either side of the main carbon atom) for each amino acid about which rotation *can* occur, thus allowing the protein to twist into its final conformation. However,

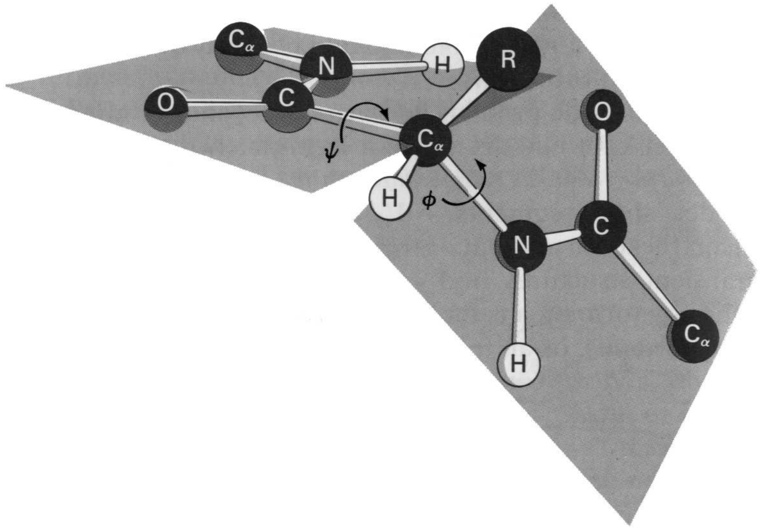

Figure 3: Amino acid residue with two planar peptide bonds

rotation about these two bonds (designated **phi** and **psi** (ϕ and ψ)) is not entirely free. Each amino acid can only access certain ranges of ϕ and ψ due to interactions of the side chains with the backbone. The accessible values for each amino acid are summarized in **Ramachandran plots** ($\phi - \psi$plots), named after the researcher who did the founding work in this area [Figure 4] [12]. Specifying ϕ and ψ for each amino acid in the protein will describe the position of the backbone in space, and thus specify fairly precisely the conformation of the protein [14] (this will not *completely* specify the structure, due to the rotation of each side chain about the bond that connects it to the backbone).

The position of the backbone defines the conformation (**tertiary structure**) of a protein, but it also defines an intermediate level of structure: **secondary structure**. Regions of proteins often fold into regular secondary structures, such as α-helices, β-sheets, and random coils, among others; these secondary structure then interact to define the tertiary structure. Figure 5 shows an α-helix, with some of the interactions between the residues indicated by dashed lines.

The ease of determining the sequence of a protein, combined with the fact that the primary structure (in conjunction with the normal physiological environment) completely specifies the tertiary (3D) structure, presents an exciting possibility–predicting the final conformation from sequence data alone. Current biochemical methods for determining conformation, X-ray crystallography and Nuclear Magnetic Resonance Imaging, are laborious and time-consuming, typically taking at least a person-year to determine a single structure. Additionally, these methods do not work with all proteins. These difficulties, and the ease of finding protein sequences, provide the motivation for the computational work examined in following section.

3 Current Computational Methods

There are two broad classes of structure prediction [7]. One class takes an *ab inito* (from 'first-principles') approach, modeling the quantum and molecular mechanical interactions of individual atoms or molecules to determine the final structure. The other class encompasses the variety of approaches that use heuristics or empirical methods. Several methods also use aspects of both of these approaches.

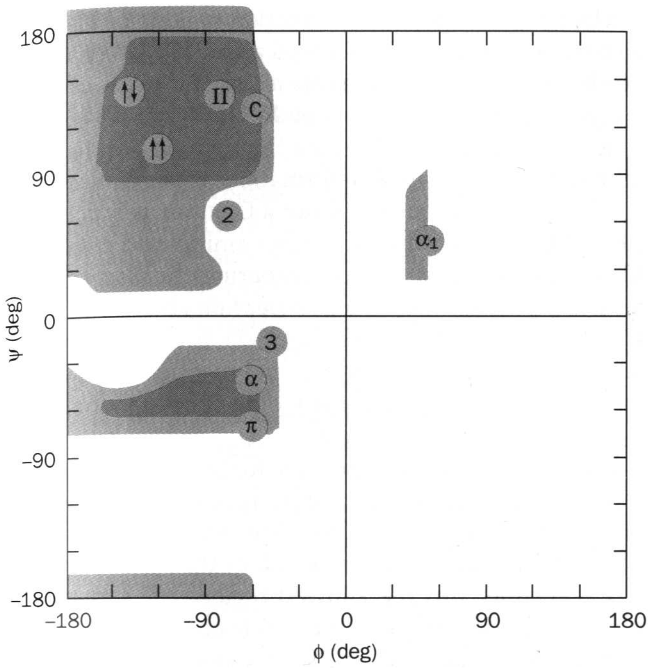

Figure 4: Ramachandran Plot of observed $\phi - \psi$ values for poly-L alanine

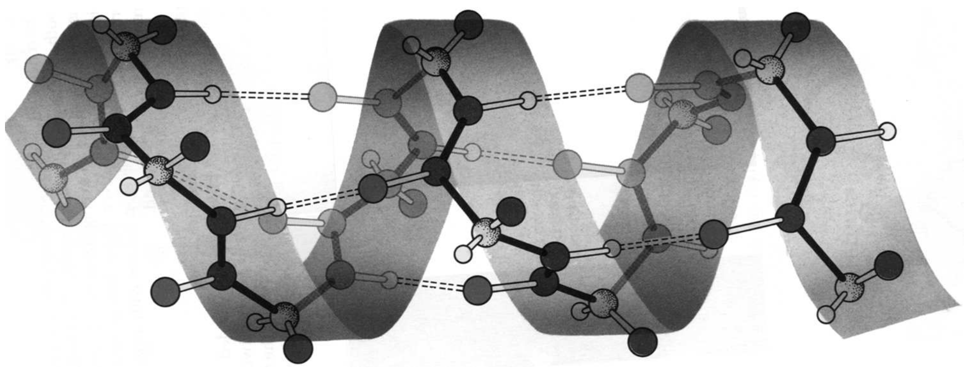

Figure 5: α-Helix, showing the interactions between (sequentially) distant residues

3.1 Heuristic and Empirical Approaches

The work upon which most of the empirical approaches are based was published in 1974 by Chou and Fasman [3]. This study was an extensive statistical analysis of 15 proteins whose sequences and conformations were known. Chou and Fasman tabulated the occurrence of each of the twenty amino acids in α-helical, β-sheet and random coil secondary structures, and thus calculated the probability that a particular residue would be found in one of those structures. This analysis showed that certain amino acids had a definite propensity for one type of secondary structure over the others. The researchers also studied the regions on the boundaries of helices, to determine residues likely to appear at such boundaries. Again, certain residues were found to appear with much greater frequency at either the start or end of helices.

These observations provided other researchers with the 'tools' to begin to analyze protein sequences. For example, proline residues were never found in inner helices, and instead were most often found in random coil regions. This observation can then be used to make predictions about secondary structure from sequence data. In this way, a probability is assigned to each residue in a new sequence that describes the likelihood that residue is found in one of the three secondary structures. If a majority of residues (for example, four residues out of six) have a propensity for helix formation, then a helix is assigned to that portion of the sequence. The helix is then assumed to continue until either a proline residue or a series of residues found at the boundaries of helices is reached. Using this method, Chou and Fasman [3] were able to predict the three basic secondary structures with 75-80% accuracy, using the sequences of their original 15 proteins and four others.

While this result may not seem surprising considering that the technique was evaluated using the proteins from which the model was derived, this was a significant result at the time. Subsequent studies have refined this basic method, developing more accurate probabilities by examining more known structures. However, this made the situation more complex, with exceptions to the empirical 'rules' becoming common as more structures were examined. Because of this, the accuracy of this approach is now considered to be roughly 65%, when tested on a wide range of proteins [13]. This emphasizes one of the problems with this method–looking at each amino acid in isolation does not provide enough information to fold a protein. Additionally, this method only addresses secondary structure, and is not able to predict how the secondary structures will interact to define the final conformation.

A more advanced empirical method based on pattern matching is **homology modeling** [5]. Homology modeling, or sequence similarity searching, makes extensive use of protein structure databases, and the fact that many structural motifs are shared between proteins. When a new sequence is to be analyzed, it is compared to the sequences of proteins of known structure to find any similar regions. If a reasonable match is found, then the structure derived from the new sequence is assumed to be the same as the structure of the protein with which there is a sequence match.

The advantage of this method over Chou and Fasman's is that several residues are examined at once; matches are usually found between strings of roughly five or ten residues. Therefore, this method carries at least some information about the local environment of a particular residue, but still fails to take into account the effects of residues that may be a great distance away in the linear primary structure, but may in fact interact in the final tertiary structure. Of course, in certain cases, there will be a great deal of similarity between two entire protein sequences; in this case, the proteins are said to have a high degree of homology (to each other), and are likely related. In these cases, the overall accuracy of the tertiary structure prediction can be very high, since the proteins can be considered 'variations on a theme,' and thus have similar conformations. Unfortunately, the majority (five out of six) of new sequences show no detectable homology to sequences in current databases [13].

A novel empirical method is the knowledge-based approach of Clark *et al.* [4]. This work attempts to model protein topology–the details of the association of secondary structures (for example, whether a pair of helices is parallel or antiparallel). This requires that the secondary structure is known (or predicted), but complete accuracy is not essential for use with topology prediction. This an important feature considering that the methods already discussed have only modest accuracy. Clark *et al.* collected a set of 11 high-level constraints, or folding rules, that were devised by molecular biologists through empirical observation. This is one of the few approaches to use such high-level 'expert-systems knowledge.

Another unique feature of this work is that the constraints were implemented in a CLP language. The use of logic programming (LP) allowed the expert knowledge of biochemists to be expressed in only five or ten lines of code per constraint, and the use of constraints lead to massive efficiency gains over a plain LP language. In one test case, the CLP implementation calculated an answer in approximately two minutes, whereas the standard LP implementation was projected to take 15 years! It would be valuable to compare this result to an imperative approach, but there does not appear to be an imperative implementation of topology prediction. This may be due to the difficulty of encoding the expert knowledge constraints, or because the method was not considered useful. The second possibility is difficult to assess, as the authors tested their system on only seven proteins, and even that testing was primarily used to 'tune' the constraints so that system predicted the known topologies of the test proteins. Thus, this approach is still in a developmental stage, and an evaluation will have to wait for a more mature system.

3.2 *Ab Inito* Approaches

The founding work for this approach is summarized in Anfinsen's Nobel prize acceptance speech [2]. Anfinsen identified two important concepts: the primary structure contains all of the information to specify the complete conformation, and proteins fold to minimize their global free energy. The first observation provides the justification for predicting protein structure from sequence data alone. The second, also called the 'thermodynamic hypothesis,' provides the basis for ab inito approaches to structure prediction. The thermodynamic hypothesis states that a normal, active protein in its natural physiological environment will fold in such a way that the total energy of the system is at its lowest. In other words, the protein's conformation will be such that, overall, unfavorable interactions between the protein and itself and its environment will be minimized.

These interactions are calculated via a set of energy terms. Each term models various quantum or molecular mechanical effects such as electrostatic energy, van der Waals interactions, and hydrophobicity [5]. Thus, the folding problem amounts to maintaining this set of terms for each amino acid in the protein, and then finding a conformation that yields the global minimum for these terms. Unfortunately, the search space for this problem is extremely large.

Proteins range in size from a few amino acid residues to over 4500, and each residue has, on average, about 13 atoms. A worst-case problem, which would specify the position of each atom (roughly 60,000 for the above protein) at a resolution of $2\mathring{A}$ (the typical resolution of X-ray crystallographic methods), would define a search space of $10^{46,000}$ possible structures! Even with a simplified model, using only 3 atoms per residue and a $3\mathring{A}$ resolution, on a protein of moderate size (450 residues) yields $10^{1,700}$ possible structures [8]. Fortunately, in practice, the problem can be much simpler. Much of the search space can be eliminated *a priori*; for example, two atoms cannot occupy the same space. So, for the example above, the search space will be considerably smaller than $10^{1,700}$. Still, this leaves a great number of possibilities to be evaluated, and this evaluation involves calculating several energy terms (most of which are computationally intensive) for each amino acid. Obviously, it is crucial to reduce the search space, and important to optimize the evaluation of the energy terms.

One way to reduce the search space is provide the system with a proposed structure. This is a hybrid approach, using empirical methods to obtain a rough structure, and then using *ab inito* methods to refine that structure. This technique has actually been quite successful, and several packages are available to perform energy minimization starting with a proposed structure, a process called **energy refinement**. However, these packages are only useful for refining a structure, performing local minimizations and adjusting local structure to remove unfavorable interactions; they cannot make large, general structural changes to search for a better conformation.

Energy refinement algorithms fail to make significant structural changes due to the 'local minimum problem.' These algorithms perturb an initial structure, attempting to minimize energy. However, they will not make a change in the structure that will increase the energy, and this can prevent them from 'climbing' out of 'energy wells.' At the point labeled MG in Figure 6, there exists a structure

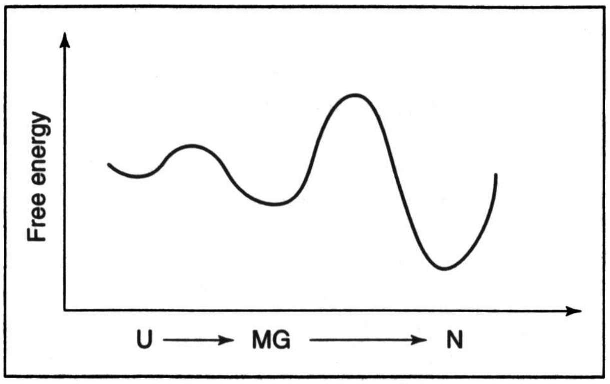

Figure 6: Relative energy of conformations from unfolded (U) to native (N)

of lower energy than the initial, unfolded (U) structure, and this relative minimum can be found by energy refinement. However, the final native state N is the true minimum, and to reach this structure from MG, the algorithm would first have to generate a structure of greater energy than it started with.

Conformational search is perhaps the most comprehensive of the ab inito methods, and one that avoids the problems of local minima. Starting with only raw sequence data, these algorithms generate (randomly or by rules) structures which are then tested or further optimized by the energy minimization models described above. This avoids the problem of local minima and incorrect initial models, as all possible structures can, in theory, be generated and evaluated.

These methods are usually based on$\phi - \psi$maps, with the algorithm generating new $\phi - \psi$ values to specify the position of the protein backbone, and thus generate a new structure to evaluate. The methods differ in how they evaluate the structures and how they choose new $\phi - \psi$ values. Kang et al. [10] derived their own Ramachandran plot using values found in 55 globular proteins, averaged over all residues. They included a third dimension, the frequency with which each $\phi - \psi$ value was observed in the sample. This frequency was used to assign a probability to each $\phi - \psi$pair (using a 10 degree interval), and thus, the values that appear more often in nature were more likely to be chosen when generating new structures. This method was only moderately accurate in predicting the structure of several small (15 residues) proteins, possibly due to insufficient 'training' data (the 55 proteins from which the Ramachandran plots were constructed) and from averaging the values from all residues.

A more recent method, Biased Probability Monte Carlo (BPMC) conformational searching, by Abagyan and Totrov corrects these shortcomings [1]. They used 191 proteins to calculate the Ramachandran plots, and maintained separate plots for each amino acid residue. They also used a more extensive energy minimization routine to check the generated structures. The results of these improvements are encouraging: the conformations of two small proteins (12 and 16 residues) were predicted with very good accuracy. It will be interesting to see the results of more rigorous testing of this method, as it appears to be the most successful protein structure prediction system to date.

4 CLP Background

All of the computational methods described in the previous section, with the exception of the Clark paper [4], use imperative languages to attack the problem area. Most people with a computing background are familiar with imperative languages, such as C and Pascal. Another, less well-known, class of languages is the declarative languages, such as Prolog and Lisp. However, these languages are gaining commercial acceptance and are increasingly being used to tackle problems that have remained difficult to solve using imperative languages. This section examines Constraint Logic Programming, a relatively new field in the declarative paradigm that holds particular promise for the problem area of protein structure prediction.

Constraint Logic Programming (CLP) is composed of two other declarative paradigms, logic programming (LP), and constraint solving [9]. One of the most important characteristics of LP is the declarative nature of the programs. This makes it easy (relative to imperative languages) to encode the high-level logic of the computation, rather than be bogged down in the details of the implementation. LP languages are often called Very High Level languages for this reason.

LP implementations such as Prolog usually have built-in facilities for backtracking search. Indeed, the basic computation model for Prolog is to follow a branch of a resolution tree, and, upon failure, backtrack to a previous node to try another branch [11]. This makes Prolog particularly well-suited to expressing search problems, 'liberating the user from tree-search programming' [15].

Another advantage of LP languages for large search problems is that the basic derivation step, or 'execution unit,' is nondeterministic (this is not the case for plain Prolog, however); that is to say, the program in general says nothing about the order of execution of its statements. This is in contrast to imperative programs, which are inherently deterministic and sequential. Nondeterminism helps to avoid the von Neumann bottleneck found in imperative languages, and facilitates parallel implementations.

A key problem in computer science today is parallel execution, and converting traditional imperative programs to run in parallel has been difficult. Additionally, a problem such as protein structure prediction that is extremely computationally intensive must be attacked with parallel implementation in mind. Fortunately, the nondeterminism of declarative languages makes writing parallel compilers and programs easier than in the imperative paradigm.

However, LP is not without problems. The most important limitation with regard to scientific computing is performing numeric calculations. For example, Prolog only deals with the unification of symbols; a term such as '2 + 3' has no special meaning to a Prolog system (in particular, it does not 'mean' 5, but rather is only the functional term built from the symbols '2', '+' and '3'). A more important limitation is Prolog's 'generate and test' method for solving problems. Although Prolog is very proficient at stating search problems, it is inefficient at solving them [15]. The basic method is to follow a search tree branch to a leaf, and upon failing, backtrack to another branch. A more efficient method is to prune off branches of the search tree *a priori*, before expending resources (futilely) examining them.

Fortunately, these shortcomings of Prolog are addressed by incorporating constraints. Constraint solvers and Constraint Satisfaction Problems have been studied in Operations Research and Artificial Intelligence to deal with combinatorial problems such as cutting stock, scheduling and VLSI layout. These problems require finding a point in a search space satisfying a group of constraints [15].

Constraint solvers function over a particular domain (for example, real numbers) and this domain defines the class of CLP language. The incorporation of a constraint solver over the domain of real numbers into Prolog yields CLP($\Re$) (which denotes the class of CLPs which function with real number constraints). CLP($\Re$) removes Prologs difficulty of dealing with arithmetic by making numbers and standard mathematical operators part of the language. However, CLP is more than just 'math in Prolog'–it is fundamentally different from a Horn-clause LP language.

The key feature of a CLP language is the maintenance of a set of constraints along with each LP goal, and system for ensuring the **consistency** of those constraints. These constraints are checked for consistency at each LP resolution step, which allows the search space to be pruned *a priori*. This active use of constraints to eliminate branches of a search tree aid the implementation of search techniques such as forward checking or generalized propagation. Thus, CLP combines the power of LP to express high-level knowledge and combinatorial search problems, with a system for the efficient solution of such NP- complete problems.

A canonical example of the use of CLP is in the 'n-queens' problem, in which n queens are placed on an $n \times n$ chessboard such that they cannot attack each other. If the first queen is placed in position (1,1), then the constraint that no queen can attack any other means that the entire first row, first column and main diagonal can no longer be used for another queen [Figure 7 (A)]. When the system attempts to place the second queen, the search space has been constrained, and only the non-shaded squares need be considered. In this case, the queen is placed in position (2,3); under a standard

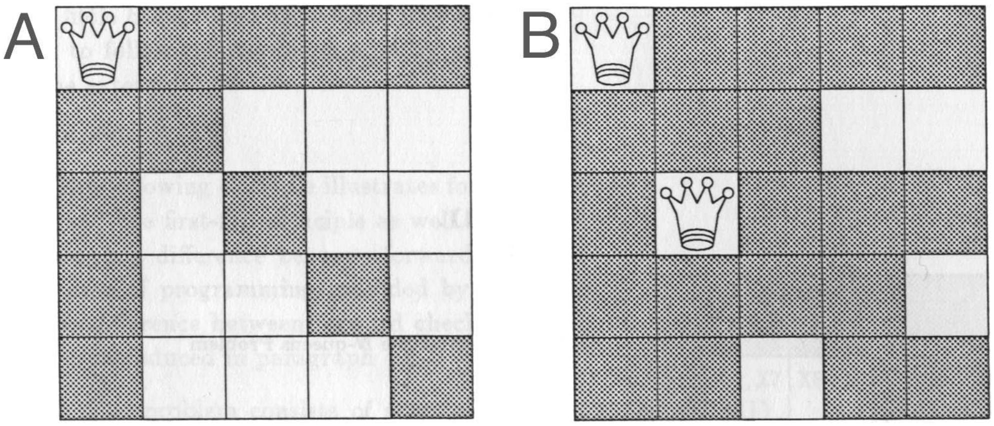

Figure 7: Safe and threatened positions after one (A) and two (B) moves

```
queens([]).
queens([X|Y]) ←
    indomain(X),
    noattack(X,Y),
    queens(Y).

noattack(X,Xs) ←
    noattack(X,Xs,1).

noattack(X,[],Nb).
noattack(X,[Y|Ys],Nb) ←
    X ≠ Y,
    X ≠ Y - Nb,
    X ≠ Y + Nb,
    Nb1 is Nb + 1,
    noattack(X,Ys,Nb1).
```

Figure 8: Prolog code for the n-queens problem

backtracking Prolog system, positions (2,1) and (2,2) would have been tested and rejected, before finding position (2,3) [Figure 7 (B)]. The system at this point is even more constrained, with the search space for the remaining three queens reduced to only four possible choices, from the original 25. This graphically illustrates the use of a set of constraints to sharply reduce a search space, which is crucial for dealing with non-trivial NP-complete problems. Additionally, this facility does not come at the expense of expressivity or readability, as evidenced by the code for the n-queens program [Figure 8].

5 Application of CLP to Protein Structure Prediction

The features of CLP described in the previous section are a good match for the problems of protein structure prediction. The main problem is the computational intractability of structure prediction. In the general case, this problem amounts to an NP-complete combinatorial search, and for this class of problems, CLP compares with or even exceeds the efficiency of special-purpose imperative solutions [9, 15]. Of course, even these traditional imperative approaches have failed to address the problem, so how can CLP be expected to do significantly better? Imperative approaches have failed because of flaws inherent in that programming paradigm, flaws which have made it difficult to encode the knowledge to efficiently solve the problem.

One set of imperative approaches deals with intractability by simplifying the problem– simplifying the representation of the structure, the equations used to minimize energy, and the search (by starting with a nearly complete structure, as in energy refinement). However, this necessarily removes information, and causes a serious problem. As Anfinsen noted, there is a 'very fine balance between native...structure and...biologically meaningless [structure]' [2]. Simplifying the problem may remove information crucial to achieving a proper conformation. This is supported by Abagyans analysis of earlier Monte Carlo conformational searches which failed to converge on a minimum energy structure [1]. It would appear that the trend should be toward representing more information. Fortunately, for a CLP solution, this should be an advantage. For non-trivial inputs (more than a dozen residues), the growth in the search space becomes the limiting factor, rather than the complexity of the minimization equations or the structure model. Thus, the cost of retaining and managing a greater amount of information encoded in a detailed model is more than offset by using that information to constrain the search space.

Another method to deal with the intractability of the problem is to avoid a conformational search altogether, as in the empirical approaches in section 3.1. However, the fundamental flaw in most of those methods is that they look at each residue (or a small string of residues) in isolation. This is problematic because a string of even five residues has been found to exist in an α-helical formation in one protein, and in a β-sheet in another [14]. Thus, even the extended local environment does not contain enough information to determine its own structure–the entire protein must be examined at once. This is not to say that this approach is without merit. In keeping with the idea of maintaining all available information, the empirical methods such as Chou and Fasman's [3] and Clark et al. [4] can make a contribution to any prediction system; the problem is to properly integrate this method with conformational search.

The integration of all the available techniques and data is another area where a CLP system should have a significant advantage over present systems. LP is known for its extensibility, and the ability to represent high-level knowledge, such as the empirical constraints used by Clark et al. [4]. CLP can additionally handle the mathematical calculations and constraints, as in the energy minimization methods of section 3.2. Finally, the LP system, with its backtracking facilities, and a constraint solver, with the ability to constrain search spaces, is well-suited to the conformational search methods. A CLP system can also easily express the nondeterminism of a search procedure, thus facilitating the parallel implementations that will be required for the demanding problems of Computational Molecular Biology.

No single current method has shown success in the general case of protein structure prediction. Together, though, they represent the significant amount of information that is known about protein structure. A declarative, specification approach based on CLP could succeed in encoding this range of knowledge and using it simultaneously to produce the highly constrained system needed to make structure prediction computationally tractable.

Finally, LP languages are known for their flexibility and modifiability, and are often used for prototyping complex systems before they are converted to a more traditional, commercial language. The field of protein structure prediction is young and rapidly evolving, with many methods requiring evaluation. A flexible and expressive 'research language such as a CLP language should be a valuable tool in the development of programs for a field in such flux as that of structure prediction.

6 Conclusion

The methods currently in use have had only limited success at predicting protein structure. This currently intractable problem requires combinatorial search facilities, the evaluation of numerous constraints, representation of expert knowledge and parallel implementation. These requirements are met by CLP, and thus further research into implementing structure prediction in a CLP language is warranted. In light of this, the authors of the most promising systems, Abagyan [1] and Clark [4], have been contacted for further discussion and to obtain their implementations for evaluation.

The next step in this research will be to implement portions of a conformational search system in a CLP language. This will first require designing a CLP representation of the constraints given by Ramachandran plots; a good candidate language is Eclipse, which has a Prolog syntax, user-definable constraints, and OR parallelism. If this is successful, initial performance tests can be done to give a more concrete evaluation of the feasibility of this approach to protein structure prediction.

References

[1] Abagyan, R., and Totrov, M. Biased probability Monte Carlo conformational searches and electrostatic calculations for peptides and proteins. *J. Mol. Biol.*, 235(3) (1994), 983-1002.

[2] Anfinsen, C. B. Principles that govern the folding of protein chains. *Science*, 181 (1973), 223-230.

[3] Chou, P. Y., and Fasman, G. D. Conformational parameters for amino acids in helical, b-sheet, and random coil regions calculated from proteins. *Biochemistry*, 13(2) (1974), 211.

[4] Clark, D. A., Shirazi, J., and Rawlings, C. J. Protein topology prediction through constraint-based search and the evaluation of topological folding rules. *Protein Engineering*, 4(7) (1991), 751-760.

[5] Dagget, V. and Kollman, P. A. Theoretical methods in protein engineering. In *Protein Engineering: A Pratical Approach*, Oxford University Press, Oxford, 1992.

[6] Hinds, D. A, and Levitt, M. Simulation of protein-folding pathways: lost in (conformational) space? *Trends in Biotechmology*, 13 (1995) 23-27.

[7] Holbrook, S. R., Muskal, S. M., and Kim, S. Predicting Protein Structural Features With Artificial Neural Networks. In *Artificial Intelligence and Molecular Biology*, AAAI Press, Menlo Park, California, 1993, pp. 161-194.

[8] Hunter, L. Planning to Learn About Protein Structure. In *Artificial Intelligence and Molecular Biology*, AAAI Press, Menlo Park, California, 1993, pp. 259-288.

[9] Jaffar, J. and Maher, M. J. Constraint Logic Programming: A Survey. *Journal of Logic Programming*, 19, 20 (1994), 503-581.

[10] Kang, H. S., Kurochkina, N. A., and Lee, B. Estimation and use of protein backbone angle probabilities. *J. Mol. Biol.*, 229(2) (1993), 448-460.

[11] Nilsson, U., and Maluszynski, J. *Logic, Programming and Prolog*, Wiley, Chichester, England, 1990.

[12] Ramachandran, G. N. Conformation of Polypeptides and Proteins. *Advances in Protein Chemistry*, 23 (1968), 284-438

[13] Rost, B., Schneider, R., and Sander, C. Progress in protein structure prediction? *Trends in Biotechnology*, 4 (1993), 120-123.

[14] Stryer, L. *Biochemsitry*, W. H. Freeman & Co., New York, 1988, pp. 22-36.

[15] Van Hentenryck, P. *Constraint Satisfaction in Logic Programming*, MIT Press, Cambridge, MA, 1989.

[16] Voet, D., and Voet, J. *Biochemistry*, Wiley, New York, 1990, pp. 144-192.