PROTEAN: DERIVING PROTEIN STRUCTURE FROM CONSTRAINTS

Barbara Hayes-Roth*, Bruce Buchanan*, Olivier Lichtarge+,
Mike Hewett*, Russ Altman*, James Brinkley*,
Craig Cornelius*, Bruce Duncan*, and Oleg Jardetzky+[1]
KNOWLEDGE SYSTEMS LABORATORY STANFORD MAGNETIC RESONANCE LAB.
COMPUTER SCIENCE DEPARTMENT STANFORD UNIVERSITY MEDICAL CTR.
STANFORD UNIVERSITY
STANFORD, CA 94025

Abstract

PROTEAN is an evolving knowledge-based system that is intended to identify the three-dimensional conformations of proteins in solution. Using a variety of empirically derived constraints, PROTEAN must identify legal positions for each of a protein's constituent structures (e.g., atoms, amino acids, helices) in three-dimensional space. In fact, because protein-structure analysis is an underconstrained problem, PROTEAN must identify the entire *family* of conformations allowed by available constraints. In this paper, we discuss PROTEAN's approach to the protein-structure analysis problem and its current implementation within the BB1 blackboard architecture.

1. Introduction

PROTEAN [3, 7, 9] is an evolving knowledge-based system, framed within the blackboard architecture, that is intended to derive the three-dimensional conformations of proteins in solution from empirical constraints. 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. Accordingly, in designing PROTEAN, we are developing knowledge and methods that apply to arrangement problems generally. We describe the PROTEAN system, as implemented in the BB1 blackboard architecture [5], and present a trace of PROTEAN's efforts to solve a small protein fragment, the lac-repressor headpiece. Finally, we discuss PROTEAN's current status.

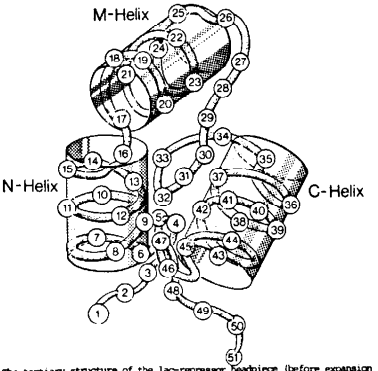
2. Protein Structure Elucidation

Determining the structures of individual proteins is a fundamental problem in biochemistry. It is the first step toward understanding the physical basic underlying protein functions and, possibly, designing new proteins for medical or industrial use.

Biochemists distinguish the primary, secondary, and tertiary structure of a protein. A protein's primary structure is its defining, linear sequence of amino acids. A protein's secondary structure is the sequence of architectural subunits (alpha helices, beta sheets, and random coils) superimposed on successive subsequences of its primary structure. A protein's tertiary structure is the folding of the primary and secondary structures in three-dimensional space. Figure 1 shows the structure of a protein called the lac repressor headpiece.

[1]This work was funded in part by:NIH Grant #RR-00785; NIH Grant #RR-00711; NSF Grant #DMB84-2038; NASA/Ames Grant #NCC 2-274; Boeing Grant #W266875; DARPA Contract #N00039-83-C-0136. We thank Jeff Harvey, Vaughan Johnson, and Alan Garvey for their work on BB1.

3-D Structure of the Lac Repressor Headpiece
Defined by NMR



1. The tertiary structure of the lac-repressor headpiece (before expansion of amino acids into their constituent atoms). The primary structure is the linear string of 51 amino acids. The secondary structures are the three alpha helices, shaded as cylinders, and the four random coils flanking the helices. The tertiary structure is the folding of the primary and secondary structures in three-dimensional space.

Biochemists have developed reliable methods exist for determining a protein's primary structure and its secondary structure. In addition, they know the atomic structure of each of the twenty different amino acids that can appear in the primary structure and the radius of each different atom (its *van der Waals' radius*). They know the architectural characteristics of alpha helices, beta sheets, and random coils. They can determine the overall size, shape, and density of the protein molecule with hydrodynamic and light-scattering methods.

Protein crystallography currently is the best method for determining tertiary structure and there has been some success in developing knowledge-based systems for interpreting crystallographic data [12]. However, obtaining crystallized samples of proteins is not always possible. Moreover, it is not known whether the identified crystal structures match the structures of proteins in solution. The crystal structures almost certainly deviate from the solution structures in one respect: they conceal the potential mobility of a protein's constituent structures.

High-resolution nuclear magnetic resonance (NMR) offers an alternative method of obtaining structual information about proteins in solution [11, 14]. NMR experiments yield a set of measurements called *nuclear Overhauser effects* (NOEs). Each NOE signifies that two of a molecule's constituent atoms are in close spatial proximity (within a range of 2-5 angstroms). Other measures reveal the overall size and shape of the protein and identify atoms located near the surface of the molecule. Taken together, these data substantially constrain the space of plausible tertiary structures.

Efforts to develop computer programs for deriving protein structure from NMR data have focused on distance geometry algorithms that minimize the value of some distance error function [14, 13, 1, 15]. These approaches suffer two major limitations. First, since NMR data are sparse, they do not identify a unique solution for a given protein. Existing programs do not thoroughly explore the "conformational space" of solutions that satisfy a given set of constraints. Instead, they explore a local region of solutions around a plausible starting structure. Second, existing programs treat potential mobility in a very limited fashion. They may hypothesize minor mobility of small substructures (such as amino acid sidechains), while failing to consider major mobility of larger substructures (such as helices).

3. Approach

PROTEAN is intended to surmount the limitations of existing methods for elucidating protein structure from NMR data. Thus, PROTEAN must: identify the *family* of conformations allowed by available constraints; incorporate all available constraints to restrict the family as much as possible; and characterize the mobility of protein substructures allowed by the constraints. In so doing, it must cope with the large, combinatoric search space entailed in protein-structure analysis.

PROTEAN's fundamental operation is to identify and then refine the family of positions in which a structure satisfies a designated set of constraints. Successively applied constraints successively restrict the family hypothesized for a given structure. We have identified a variety of potentially useful constraints on protein structure (see Table 1). Some of these are local constraints, such as NOE data signifying the proximity of a particular pair of atoms. Others are global constraints, such as molecular size. By combining these qualitatively different kinds of constraints, PROTEAN should be able to restrict the space of possible protein conformations.

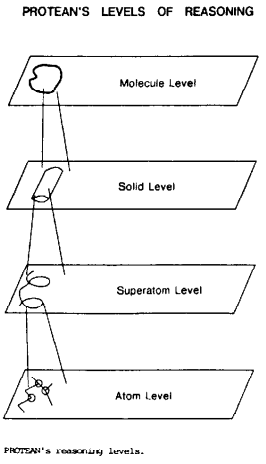
Table 1. Some of the Available Constraints on Protein Structure

Primary structure
Atomic structures of individual amino acids
van der Waals' radii of individual atoms
Peptide bond geometry
Secondary structure
Architectures of alpha-helices and beta-sheets
Molecular size
Molecular shape
Molecular density
NOE measurements
Surface data

PROTEAN must consider two factors in reasoning about structural mobility. First, it must infer mobility whenever it finds that no position for a structural subunit (such as a helix) satisfies a set of applicable constraints. Second, it must incorporate user-specified hypotheses that particular sets of constraints are or are not satisfied simultaneously in a single conformation. In both cases, PROTEAN must reason about alternative families of positions for affected structures under non-simultaneous constraint sets.

To reduce the combinatorics of search, PROTEAN adopts a "divide-and-conquer" approach. It defines partial solutions that incorporate different subsets of a protein's constituent structures and different subsets of its constraints. It focuses first on satisfying constraints within each partial solution, positioning each structure relative to a single fixed structure. After substantially restricting the positions of structures within two overlapping partial solutions, PROTEAN applies constraints between them.

Also to reduce search combinatorics, PROTEAN reasons bidirectionally across different levels of abstraction (see Figure 3). At the *molecule* level, PROTEAN reasons about the overall size, shape, and density of the molecule. At the *solid* level, it reasons about the protein's constituent alpha-helices, beta-sheets, and random coils, representing these structures as geometric cylinders, prisms, and spheres. At the *superatom* level, it reasons about the protein's constituent amino acids, in terms of peptide units (represented as prisms) and sidechains (represented as spheres). Finally, at the *atom* level, PROTEAN reasons about the protein's individual atoms. When PROTEAN reasons top-down, it uses the hypothesized position of a structure at one level to restrict its examination of positions of constituent structures at a lower level. When PROTEAN reasons bottom-up, it uses the hypothesized position of a structure at one level to restrict the position of its superordinate structure. Since most of the current implementation operates at the Solid level, we have not yet explored the full power of bidirectional reasoning.



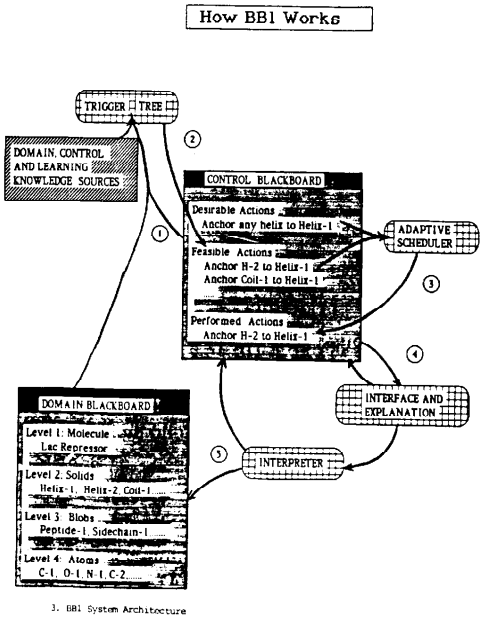
2. PROTEAN's reasoning levels.

We envision a basic successive refinement strategy, with some opportunistic deviation. Thus, PROTEAN should reason reason top-down through the levels of abstraction, with some bottom-up adjustment of results. It should apply this strategy simultaneously to several overlapping partial solutions, integrating them only after it has applied all or most of their internal constraints. Within this general strategy, PROTEAN still faces an extensive solution space and must reason more specifically about the most efficient order in which to apply individual constraints to individual structures in particular partial solutions. We have implemented a strategy that combines domain-independent computational principles (e.g., choosing partial-solution "anchors" that have many constraints to many other structures; focusing on structures that have been restricted to small families; and preferring strong constraints) with biochemistry knowledge (e.g., defining the space of potentially useful constraints; and characterizing the constraining power of different constraints). Since intelligent control is a critical component of effective problem-solving in PROTEAN, we plan to experiment with these and other control strategies.

4. Current Implementation

We are developing PROTEAN within the BB1 blackboard architecture [5], which defines: (a) functionally independent *problem-solving knowledge sources* to generate and refine solution elements; (b) a multi-level *solution blackboard* on which these knowledge sources record evolving solutions; (c) *control knowledge sources* to reason about problem-solving strategy; (d) the BB1 *control blackboard* on which control knowledge sources record the evolving control plan; and (e) an adaptive *scheduler* that

uses the current control plan to determine which knowledge source should execute its action on each problem-solving cycle.



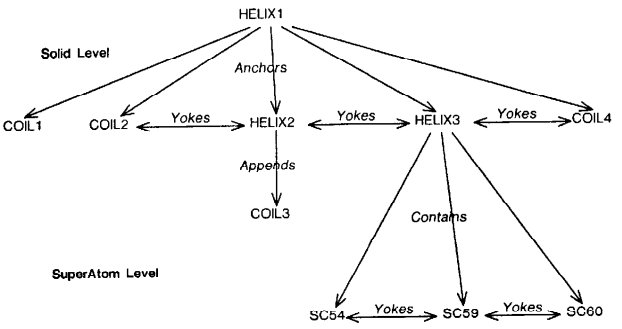
3. BB1 System Architecture

A BB1 system (see Figure 3) iterates the following steps:

1. An action (called a KSAR or knowledge source activation record) is executed, causing changes to elements on the solution or control blackboard.
2. These blackboard changes trigger one or more problem-solving or control knowledge sources, placing new KSARs on the agenda. Each KSAR instantiates an action definition from a knowledge source in the context of the current problem-solving state.
3. The scheduler chooses the pending KSAR that best satisfies the current control plan.

Thus, BB1 provides a uniform, integrated blackboard mechanism for reasoning about the solution to the problem at hand as well as about the problem-solving process *per se*.

PROTEAN currently uses a four-level solution blackboard, including the levels in Figure 2. Figure 4 shows a partial solution for the lac-repressor headpiece at the Solid level of the blackboard. The example also illustrates PROTEAN's language of partial solutions. All structures within a partial solution are positioned relative to a uniquely-positioned *anchor*. In Figure 4, Helix1 is the anchor. When PROTEAN applies constraints between the anchor and another structure, it *anchors* an *anchoree*. In Figure 4, Helix1 anchors five anchorees, Coil1, Coil2, Helix2, Helix3, and Coil4. When PROTEAN applies constraints between an anchoree and a structure that has no constraints with the anchor, it *appends* an *appendage*. In Figure 4, Hexli2 appends an appendage, Coil3. When PROTEAN applies constraints between two anchorees or appendages, it *yokes* them. In Figure 4, for example, Helix2 and Helix3 yoke one another.



4. Blackboard representation of a partial solution for the lac-repressor headpiece.

PROTEAN's current problem-solving knowledge sources (see Table 2) define partial solutions at the Solid level and position alpha-helices and random coils relative to one another within those partial solutions. Although the current implementation of PROTEAN has only eleven problem-solving knowledge sources, it instantiates most of them many times for a single protein. For example, the knowledge source Anchor-Helix generates different KSARs for different anchor-anchoree pairs and for different constraints between a given pair.

Table 2. PROTEAN's Eleven Problem-Solving Knowledge Sources

Knowledge Source	Behavior
Post-the-Problem	Retrieves the description of a test protein and associated constraints from a data file and posts them on the blackboard in a form that is interpretable by other PROTEAN knowledge sources.
Post-Solid-Anchors	Creates objects that represent and describe the details of all of the test-protein's secondary structures (alpha-helices and random coils). Each one is a potential anchor for a solution.
Activate-Anchor-Space	Chooses a particular solid-anchor to be the anchor of a partial solution.
Add-Anchoree-to-Anchor-Space	Chooses a particular solid-anchor (representing it as a token object that *copies* the chosen solid-anchor) to be an anchoree in a previously established anchor-space.
Express-NOE-Constraint	Identifies the family of positions in which the NOE contact site of a structure can lie while satisfying an NOE with another structures.
Express-Covalent-Constraint	Identifies the family of positions in which the site of a covalent bond connecting a structure to another structure can lie.
Express-Tether-Constraint	Identifies the family of positions in which the Site of a covalent bond connecting a structure to another structure via a short random coil can lie.
Anchor-Helix	Identifies the family of positions in which a helix can lie while satisfying one or more constraints with an anchor, along with all constraints previously applied to it.
Anchor-Coil	Identifies the family of positions in which a coil anchoree can lie while satisfying one or more tether constraints with an anchor, along with all constraints previously applied to it.
Append-Helix	Identifies the family of positions in which a helix appendage can lie while satisfying one or more constraints with an anchoree, along with all constraints previously applied to it.
Yoke-Helices	Restricts the established families of positions for two helix anchorees to satisfy one or more constraints between them.

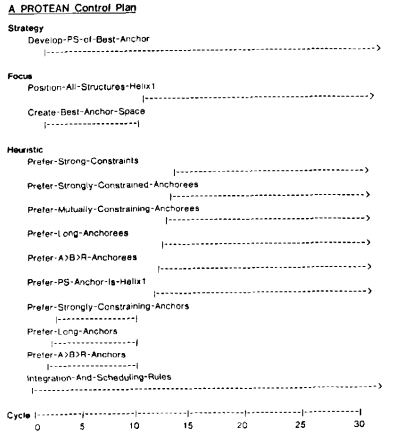
Three knowledge sources, Anchor-Helix, Anchor-Coil, and Yoke-Structures, rely upon a set of numerical functions called the *geometry system* or GS [2]. The GS represents the position of each structure as a set of six parameters. Three parameters place the structure at a particular location in the three-dimensional coordinate space and three parameters orient the strucuture about its own axes. The GS explores all values of the parameters at

some level of resolution, determining whether a designated structure positioned with those values can satisfy the designated constraints.

PROTEAN currently operates under the following problem-solving strategy:

1. Establish the longest, most constraining helix as the anchor.
2. Position all other secondary structures in the protein relative to the chosen anchor, giving priority to actions that apply strong constraints to structures that are helices, that are long, that constrain many other structures, and that have many constraints with the anchor.

Working within the BB1 architecture, PROTEAN represents this strategy as a hierarchy of decisions on the control blackboard (see Figure 5). At the *strategy* level, PROTEAN records a decision to use this particular strategy, along with the information it needs to generate the prescribed sequence of steps at the appropriate time. PROTEAN records the steps as individual decisions at the *focus* level, encompassing sequential problem-solving time intervals. Each focus decision also records the information PROTEAN needs to generate the associated heuristics, which it records as decisions at the *heuristic* level. Each heuristic encompasses roughly the same time interval as its superordinate focus decision.



5. Control plan for the first twenty-five cycles of PROTEAN's efforst to solve the lac-repressor headpiece. On cycle 2, PROTEAN decides to use the strategy, Develop-PS-of-Best-Anchor. Accordingly, it establishes and initial focus, Create-Best-Anchor-Space, and associated heuristics, such as Prever-Strongly-Constrained-Anchors. After it has created an anchor space for the best anchor, Helix1, PROTEAN establishes a new focus, Position-All-Structures-Helix1, and associated heuristics, such as Prefer-Strong-Constraints. On each cycle, the BB1 scheduler chooses pending problem-solving actions that are favored by the current heuristics.

PROTEAN generates its strategy incrementally, one decision at a time, with the sixteen control knowledge described in Table 3. Four of these are generic BB1 control knowledge sources: Initialize-Focus, Update-Focus, Terminate-Focus, and Terminate-Strategy. The other twelve control knowledge souces are domain-specific. The next section illustrates PROTEAN's use of the knowledge sources to control its efforts to solve a small protein, the lac-repressor headpiece.

Table 3. PROTEAN's Sixteen Control Knowledge Sources

Knowledge Source	Behavior
Generic BB1 Control Knowledge Sources	
Initialize-Focus	Identifies the initial focus prescribed by a newly recorded strategy.
Update-Focus	Identifies each subsequent focus prescribed by a strategy.
Terminate-Focus	Changes the status of a focus to "inoperative" the focus's goal is satisfied.
Terminate-Strattegy	Changes the status of a strategy to "inoperative" when the strategy's goal is satisfied.
Domain-Specific Control Knowledge Sources	
Develop-PS-of-Best-Anchor	Records the develop-ps-of-best-anchor strategy.
Create-Best-Anchor-Space	Records the create-best-anchor-space focus.
Position-All-Structures	Records the position-all-structures focus.
Prefer-Helix>Sheet>Coil-Anchors	Records a heuristic that gives high ratings To KSARs that operate on helix anchors, intermediate ratings to KSARs that operate on beta-sheet anchors, and low ratings to KSARs that operate on random coil anchors.
Prefer-Long-Anchors	Records a heuristic that gives higher ratings to KSARs that operate on long anchors.
Prefer-Strongly-Constraining-Anchors	Records a heuristic that gives higher ratings to KSARs whose anchors have many constraints with many other structures.
Prefer-Strategically-Selected-Anchor	Records a heuristic that gives higher ratings to KSARs that operate on the strategically-selected anchor.
Prefer-Helix>Sheet>Coil-Anchoree	Records a heuristic that gives high ratings to KSARs that operate on helix anchorees, intermediate ratings to KSARs that operate on beta-sheet anchorees, and low ratings to KSARs that operate on random coil anchorees.
Prefer-Long-Anchoree	Records a heuristic that gives higher ratings to KSARs that operate on long anchorees.
Prefer-Strongly-Constrained-Anchoree	Records a heuristic that gives higher ratings to KSARs that operate on anchorees that have many constraints with the anchor.
Prefer-Mutually-Constraining-Anchoree	Records a heuristic that gives higher ratings to KSARs that operate on anchorees that have many constraints with other anchorees.
Prefer-Strong-Constraint	Records a heuristics that gives higher ratings to KSARs that apply strong constraints.

5. Example: PROTEAN's Partial Solution of the Lac-Repressor Headpiece

The lac-repressor headpiece is a protein with fifty-one amino acids. Its true structure is unknown, but NMR data are available for it and several research groups have partially identified its structure [10, 8]. Interpreted data for the lac-repressor are shown in Table 4. This section describes the first 25 cycles of a program trace of PROTEAN's efforts to solve the lac-repressor headpiece.

Table 4. Interpreted Data for the Lac-Repressor Headpiece

Amino acids are numbered sequentially in the primary structure and named according to biochemical conventions. LYS2, for example, is the second amino acid in the sequence and is a lysine. NOEs identify particular atoms within particular amino acids that are within 2-5 angstroms of one another. For example, NOE 1 specifies the atom 3 of Valine 4 is within 2-5 angstroms of atom 5 of tyrosine 17.

Data Type	Data Value
PROTEIN-NAME	LAC-REPRESSOR-HEADPIECE
PRIMARY-STRUCTURE	MET1 LYS2 PRO3 VAL4 THR5 LEU6 TYR7 ASP8 VAL9 ALA10 GLU11 TYR12 ALA13 GLY14 VAL15 SER16 TYR17 GLN18 THR19 VAL20 SER21 ARG22 VAL23 VAL24 ASN25 GLN26 ALA27 SER28 HIS29 VAL30 SER31 ALA32 LYS33 THR34 ARG35 GLU36 LYS37 VAL38 GLU39 ALA40 ALA41 MET42 ALA43 GLU44 LEU45 ASN46 TYR47 ILE48 PRO49 ASN50 ARG51))

```
SECONDARY-STRUCTURE (Coil1  MET1 THR5)
                    (Helix1 LEU6 GLY14)
                    (Coil2  VAL15 SER16)
                    (Helix2 TYR17 ASN25)
                    (Coil3  GLN26 ARG35)
                    (Helix3 GLU36 LEU45)
                    (Coil4  ASN46 ARG51)

NOES                (1 VAL4 3 TYR17 5) (2 VAL4 3 LEU45 4)
                    (3 VAL4 3 TYR47 5) (4 THR5 3 TYR47 5)
                    (5 LEU6 4 TYR17 5) (6 LEU6 4 VAL24 3)
                    (7 LEU6 4 MET42 5) (8 LEU6 4 TYR47 5)
                    (9 TYR7 5 TYR17 5) (10 ASP8 3 LEU45 4)
                    (11 VAL9 3 MET42 5) (12 VAL9 3 LEU45 4)
                    (13 VAL9 3 TYR47 5) (14 ALA10 2 TYR17 5)
                    (15 ALA10 2 VAL20 3) (16 TYR12 5 ALA32 2)
                    (17 TYR12 5 ALA40 2) (18 TYR12 5 ALA41 2)
                    (19 TYR12 5 MET42 5) (20 TYR12 5 GLU44 4)
                    (21 TYR12 5 LEU45 4) (22 ALA13 2 VAL38 3)
                    (23 ALA13 2 ALA41 2) (24 VAL15 3 TYR47 5)
                    (25 TYR17 5 MET42 5) (26 VAL20 3 VAL38 3)
                    (27 VAL24 3 TYR47 5) (28 VAL30 3 MET42 5)
                    (29 MET42 5 TYR47 5)
```

Post-the-Problem initiates PROTEAN activity at the Molecular level by recording a new protein-analysis problem and all available constraints. This event triggers two knowledge sources: Post-Solid-Anchors and Develop-PS-of-Best-Anchor.

Since there are no control heuristics on the control blackboard yet, the scheduler uses the default scheduling rule: Prefer-Control-KSs. It schedules and executes Develop-PS-of-Best-Anchor, which records PROTEAN's strategy (see Figure 5). This event triggers Terminate-Strategy, which will not be executable until the strategy's *goal* (explained below) is satisfied. It also triggers Initialize-Focus.

The scheduler chooses Initialize-Focus, which uses the strategy's *generator* to identify the first focus it prescribes. It records the name of that focus, "Create-Best-Anchor-Space," as the strategy's *current-focus*. This event triggers Create-Best-Anchor-Space.

The scheduler chooses Create-Best-Anchor-Space, which records the corresponding focus (see Figure 5). This event triggers three control knowledge sources whose names are listed as the new focus decision's *heuristics*: Prefer-Helix>Sheet>Coil-Anchors, Prefer-Long-Anchors, and Prefer-Strongly-Constraining-Anchors. It also triggers Terminate-Focus, which will not become executable until the new focus's goal is satisfied.

On the next three cycles, the scheduler chooses three pending KSARs, each of which records a heuristic (see Figure 5). These events do not trigger any new knowledge sources.

The scheduler chooses the only pending KSAR, Post-Solid-Anchors, which creates a potential anchor representing each secondary structure in the protein. Each of these events triggers a corresponding KSAR involving Create-Anchor-Space.

Now the scheduler uses the three heuristics posted on the control blackboard to determine which of the Create-Anchor-Space KSARs to execute. Since Helix1 is the longest, most constraining helix, it chooses the KSAR that creates an anchor space for Helix1 (see Figure 4). This event satisfies the goal of the Create-Best-Anchor-Space focus (the best anchor space has been created), thereby making the corresponding KSAR for Terminate-Focus executable. The event also triggers the knowledge source Add-Anchoree-to-Anchor-Space once for each other secondary structure in the protein.

The scheduler chooses Terminate-Focus, which changes the *status* of the existing focus and its subordinate heuristics to "inoperative." It also records the focus name, "Create-Best-Anchor-Space," as the strategy's expired-Focus. This event triggers the control knowledge source Update-Focus.

The scheduler chooses Update-Focus, which uses the strategy's generator to identify the next focus it prescribes and records the name of that focus, "Position-All-Structures," as the strategy's current-Focus. This event triggers the knowledge source Position-All-Structures.

The scheduler chooses Position-All-Structures, which records the corresponding focus (see Figure 5). This event triggers the knowledge source Terminate-Focus, which will not become executable until its goal is satisfied. The event also triggers the six control knowledge sources named in the new focus decision's heuristics: Prefer-Strategically-Selected-Anchor, Prefer-Helix>Sheet>Coil-Anchorees, Prefer-Long-Anchorees, Prefer-Strongly-Constrained-Anchorees, Prefer-Mutually-Constraining-Anchorees, and Prefer-Strong-Constraints.

On the next six cycles, the scheduler chooses KSARs that record heuristics for the new focus.. These events do not trigger any new knowledge sources.

Now the scheduler uses the six new control heuristics to choose pending KSARs. At this point, the agenda contains only KSARs involving the knowledge source Add-Anchoree-To-Anchor-Space. The scheduler chooses the KSAR that adds Helix3 (see Figure 4). This event triggers several KSARs for Express-NOE-Constraint, one for each of the NOEs between Helix1 and Helix3.

The scheduler chooses a series of Express-NOE-Constraint KSARs. Each one records the family of positions in which the NOE contact site on Helix3 can lie, relative to Helix1. Each of these events triggers a corresponding KSAR for the knowledge source Anchor-Helix.

The scheduler continues using the six control heuristics to choose pending problem-solving knowledge sources, including many different triggerings of the knowledge sources: Add-Anchoree-to-Anchor-Space, Express-NOE-Constraint, Express-Tether-Constraint, Anchor-Helix, Anchor-Coil, and Yoke-Structures. Each such action triggers new KSARs, which are added to the agenda and compete for scheduling priority. All of these?r KSARs together position all secondary structures relative to Helix1 with all applicable constraints (see Figure 4).

Because the results of these actions satisfy the goal of the Position-All-Structures goal (all structures have been positioned), Terminate-Focus becomes executable and the scheduler chooses it. Terminate-Focus changes the status of the current focus and its associated heuristics to "inoperative." It also records the focus name as the strategy's expired-Focus. This event triggers Update-Focus.

The scheduler chooses Update-Focus, which uses the strategy's generator to identify the next focus it prescribes, which in this case is "None," and records it as the strategy's current-focus. This event satisfies the strategy's goal and makes the pending Terminate-Strategy KSAR executable.

The scheduler chooses Terminate-Strategy, which changes the strategy's status to "inoperative."

In performing the actions summarized above, PROTEAN produces a solid-level solution for the lac-repressor headpiece, specifying the positional families within which each of the protein's secondary structures can lie while satisfying the applicable constraints. PROTEAN's solution closely matches the manually identified solution described in [8].

6. Current Status of PROTEAN

The current PROTEAN system demonstrates the appropriateness of the blackboard architecture for protein-structure analysis. Although PROTEAN currently reasons only about helices and random coils, we anticipate that the its representational conventions and geometric reasoning methods will apply to other protein structures as well. The current system incorporates reasoning about a variety of constraints: the known architectures of helices, covalent bonds, NOEs, the known architectures of amino-

acid sidechains, and van der Waals' radii. However, we anticipate a need to introduce qualitatively different representational conventions and geometric reasoning methods to handle the global constraints on the overall size, shape, and density of the molecule. The blackboard architecture easily incorporates different solution representations at different blackboard levels and incorporates different methods in its functionally independent knowledge sources.

The current system also suggests that the BB1 blackboard control architecture will support the critical control reasoning PROTEAN must perform. PROTEAN currently uses a single control strategy that is well captured in control knowledge sources and produces a perspicuous control plan during problem solving. This strategy works well enough for reasoning about the secondary structures of a small protein with a subset of the available constraints. However, when reasoning about all constituent structures in larger proteins with all available constraints, PROTEAN will need a new strategy. It will have to reason about multiple partial solutions and their relationships to one another. It will have to sequence its constraint-satisfaction operations intelligently to avoid a computationally intractable explosion of hypothesized structures. It will have to reason about alternative protein conformations corresponding to constraints that are not satisfied simultaneously. Since we do not know an optimal general control algorithm for this problem, we must experimentally evaluate alternative control strategies. To support this investigation, we are developing learning mechanisms to aquire control knowledge from experts automatically [6] and to comparatively evaluate different control strategies. We are also developing explanation mechanisms that explicate the relationships between problem-solving actions and the underlying control strategy [4].

References

[1] Braun, W., Bosch, C., Brown, L.R., and Wutrich, K.
Biochemistry and Biophysics 667, 1981.

[2] Brinkley, J., Cornelius, C., Altman, R., Hayes-Roth, B., Lichtarge, O., Buchanan, B., and Jardetzky, O.
Application of constraint satisfaction techniques to the determination of protein tertiary structure.
1986.

[3] Buchanan, B., Hayes-Roth, B. Lichtarge, O., Altman, R., Brinkley, J., Hewett, M., Cornelius, C., Duncan, B., Jardetzky, O.
The heuristic refinement method for deriving solution structures of proteins.
Technical Report, Stanford, Ca.: Stanford University, 1985.

[4] Hayes-Roth, B.
BB1: An architecture for blackboard systems that control, explain, and learn about their own behavior.
Technical Report HPP-84-16, Stanford, Ca.: Stanford University, 1984.

[5] Hayes-Roth, B.
A blackboard architecture for control.
Artificial Intelligence Journal 26:251-321, 1985.

[6] Hayes-Roth, B., and Hewett, M.
Learning Control Heuristics in a Blackboard Environment.
Technical Report HPP-85-2, Stanford, Ca.: Stanford University, 1985.

[7] Hayes-Roth, B., Buchanan, B Lichtarge, O., Hewett, M, Altman, R., Brinkley, J., Cornelius, C., Duncan, B., Jardetzky, O.
Elucidating protein structure from constraints in PROTEAN.
Technical Report KSL-85-35, Stanford, Ca.: Stanford University, 1985.

[8] Jardetzky, O.
Definition of the tertiary structure of proteins by NMR: The DNA binding domain of the lac-repressor.
Technical Report, Stanford, Ca.: Stanford University, 1984.

[9] Jardetzky, O., Lane, A., Lefevre, J-F., Lichtarge, O., Hayes-Roth, B., and Buchanan, B.
Determination of macromolecular structure and dynamics by NMR.
Proceedings of the NATO Advanced Study Institue: NMR in the Life Sciences , 1985.

[10] Kaptein, R., Zuiderweg, E.R.P., Scheek, R.M. and Boelens, R.
Journal of Molecular Biology 182:179-182, 1985.

[11] Roberts, G.C.K. and Jardetzky, O.
Advances in Protein Chemistry , 1970.

[12] Terry, A.
Hierarchical control of production systems.
PhD thesis, UC, Irvine, 1983.

[13] Wagner, G. and Wutrich K.
Journal of Magnetic Resonance 33:675-679, 1979.

[14] Wutrich, K.
Advances in Protein Chemistry 24:447-545, 1976.

[15] Zuiderweg, E.R.P., Kaptein, R., and Wutrich, K.
Proceedings of the National Academy of Sciences 80:5837-5841, 1983.