Integrating Diverse Reasoning Methods in the BB1 Blackboard Control Architecture[1]

M. Vaughan Johnson Jr. and **Barbara Hayes-Roth**
Knowledge Systems Laboratory
Stanford University

Abstract

The BB1 blackboard control architecture has been proposed to enable systems to integrate diverse reasoning methods to control their own actions. Previous work has shown BB1's ability to integrate hierarchical planning and opportunistic focusing. We show how it can integrate goal-directed reasoning as well and demonstrate these capabilities in the PROTEAN system. We also compare BB1 with alternative control architectures.

I. Overview

Many researchers have recognized the need for AI systems to use diverse reasoning methods — individually or together — to control problem-solving actions [Corkill *et al.*, 1982, Davis, 1976, Durfee and Lesser, 1986, Erman *et al.*, 1980, Erman *et al.*, 1981, Genesereth and Smith, 1982, Hayes-Roth, 1985, Hayes-Roth and Hayes-Roth, 1979, McCarthy, 1960, Newell *et al.*, 1959, Stefik, 1981b, Stefik, 1981a, Terry, 1983, Weyrauch, 1980]. In previous papers, we proposed the BB1 blackboard control architecture [Hayes-Roth, 1985], which enables systems to construct control plans for their own actions in real time. We argued that BB1 can accommodate a range of reasoning methods and demonstrated its performance and integration of hierarchical planning and opportunistic focusing in several application systems [Garvey *et al.*, 1987, Hayes-Roth, 1985, Hayes-Roth *et al.*, 1986b, Hayes-Roth *et al.*, 1986a, Tommelein *et al.*, 1987].

In this paper, we extend the empirical evidence for BB1's capabilities. Specifically, we show how BB1 supports goal-directed reasoning and integrates it with hierarchical planning and opportunistic focusing. We demonstrate these capabilities within PROTEAN [Buchanan *et al.*, 1985, Hayes-Roth *et al.*, 1986b], a BB1 application system for protein structure modelling. Although we discuss only PROTEAN, we also have demonstrated these capabilities in other BB1 systems, including the FEATURE system [Altman, 1986] for identifying interesting features of protein structures. In fact, BB1 provides generic control mechanisms to support these three kinds of reasoning individually and in combination, in any BB1 application system.

II. Control Reasoning in BB1

A. The BB1 Blackboard Control Architecture

BB1 [Hayes-Roth, 1984, Hayes-Roth, 1985, Hewett and Hayes-Roth, 1987] is a domain-independent architecture based on the blackboard model [Erman *et al.*, 1980, Nii, 1986]. Multiple independent *knowledge sources (KSs)* post and modify solution elements on a commonly accessible *blackboard*. *Domain KSs* solve domain problems on the *domain blackboard*. *Control KSs* develop a dynamic control plan on the *control blackboard*. Both blackboards distinguish solution elements for different *solution intervals* and different *abstraction levels*. All knowledge sources operate simultaneously, generating *knowledge source activation records (KSARs)* when specified *trigger* events occur in the context of specified *precondition* states. A *scheduler* sequences the execution of pending KSARs in accordance with the current control plan. Thus, BB1 repeatedly executes the following basic cycle:

1. *Interpret* the action of the scheduled KSAR, producing modifications to the appropriate domain or control blackboard. If the KSAR changes the control blackboard, it may alter the criteria used to rate KSARs in the next step.
2. *Update the agenda* to include KSARs triggered by the recent blackboard modifications and rate all KSARs against the current control plan.
3. *Schedule* the highest-rated KSAR.

The next two sections describe how PROTEAN is built in BB1 and how BB1 enables PROTEAN to integrate hierarchical planning and opportunistic focusing.

B. PROTEAN

PROTEAN models the three-dimensional conformations of proteins in accordance with biochemical constraints. PROTEAN's domain-specific knowledge is layered upon ACCORD (a domain-independent framework for the class of arrangement problems), which is layered on BB1. (We refer to the growing set of compatible modules exemplified by these three as BB* [Hayes-Roth *et al.*, 1986a].)

ACCORD supports an incremental assembly method for solving arrangement problems. The problem-solver defines one or more *partial arrangements*, each comprising a subset of the

---

[1] The work was supported by the following contracts and grants: NIH Grant RR-00785; NIH Grant RR-00711; Boeing Grant W266875; NASA/Ames Grant NCC 2-274; DARPA Contract N00039-83-C-0136. We thank Mike Hewett, Alan Garvey (especially for his work on hierarchical strategies in PROTEAN), Bob Schulman, Jeff Harvey, Craig Cornelius, and Reed Hastings for their work on BB1 and PROTEAN. We thank Russ Altman for noticing the need for goal-directed reasoning in his FEATURE system. We thank Bruce Buchanan and Ed Feigenbaum for supporting our work in the Knowledge Systems Laboratory.

objects and constraints in the problem specification, chooses one object as the *anchor*, and positions the other objects (*anchorees*) relative to it. The problem-solver reduces the *family* of locations for each anchoree by *anchoring* it with constraints to the anchor, and *yoking* it with constraints to other anchorees. The problem-solver *integrates* partial arrangements with constraints among their constituent objects.

ACCORD provides the following knowledge: (a) a skeletal concept network for domain-specific objects and constraints; (b) a vocabulary of arrangement roles (e.g., *partial arrangement, anchor, anchoree*); (c) a type hierarchy of assembly actions, events, and states (e.g., *Anchor is-a Position action*); and (d) linguistic templates for instantiating assembly action, event, and state sentences (e.g., *Anchor anchoree to anchor in partial-arrangement with constraints*).

PROTEAN instantiates ACCORD with domain-specific information to represent its KSs, actions, events, states, and control plans. For instance, one PROTEAN KS specifies:

```
Trigger:
   Did-Include Object (The-Object)
         in Partial-Arrangement (The-PA).

Preconditions:
   Has The-PA Anchor (The-Anchor).
   Is-Constrained-By The-Object The-Anchor
         with Constraints (The-Constraints).

Action:
   Anchor The-Object to The-Anchor
         in The-PA with The-Constraints.
```

Given the following event and states:

```
Did-Include RandomCoil4-1 in PA1.
Has PA1 Anchor (Helix1-1).
Is-Constrained-By RancomCoil4-1 Helix1-1
                  with Constraints (CSetH1R4).
```

this KS generates the action:

```
Anchor RandomCoil4-1 to Helix1-1 in PA1
                             with CSetH1R4.
```

PROTEAN might rate its action against a control decision to perform certain kinds of actions:

```
Perform:
   Position Anchorees in PA1
            with Strong Constraints.
```

by determining that:

```
Anchor is-a Position action.
RandomCoil4-1 is-a Anchoree.
PA1 is PA1.
CSetH1R4 is-a Constraint-Set,
                containing Constraints.
The Constraints in CSetH1R4 are
                moderately Strong.
```

C. PROTEAN's Control Reasoning

PROTEAN uses three of BB1's generic control KSs, *Initialize-Prescription*, *Update-Prescription*, and *Terminate-Prescription*, to integrate hierarchical planning and opportunistic focusing. Consider PROTEAN's hierarchical planning during the first eleven cycles of its reasoning about a protein called the lac-repressor headpiece (figure 1).

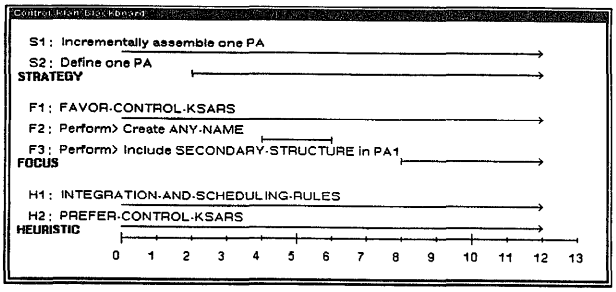


Figure 1: BB1/PROTEAN Control Blackboard at Cycle 12

On cycle 0, PROTEAN's control KS, Assemble-One-PA, initiates problem-solving activity by posting strategy S1 and focus F1, which favors control KSARs.

On cycle 1, Initialize-Prescription, which was triggered by S1, modifies S1 so that its first prescribed sub-strategy, Define-one-PA, is its Current-Prescription.

On cycle 2, PROTEAN's control KS, Define-the-PA, which was triggered by S1's new Current-Prescription, posts sub-strategy S2.

On cycle 3, Initialize-Prescription, which was triggered by the posting of S2, modifies S2 so that its first prescribed sub-strategy, Create-the-Space, is its Current-Prescription.

On cycle 4, PROTEAN's control KS, Create-the-Space, which was triggered by S2's new Current-Prescription, posts focus F2.

On cycle 5, the scheduler uses F2 to rate pending KSARs and chooses one that creates PA1. This satisfies F2's objective.

On cycle 6, Terminate-Prescription, triggered by satisfaction of F2's objective, deactivates F2.

On cycle 7, Update-Prescription, triggered by deactivation of F2, modifies S2 so that its second prescribed subordinate, Include-All-Structures, is its Current-Prescription.

On cycle 8, PROTEAN's control KS, Include-All-Structures, which was triggered by S2's new Current-Prescription, posts focus F3.

On cycle 9 and each of several subsequent cycles, the scheduler uses F3 to rate pending KSARs and chooses the best rated ones in turn.

Eventually, the scheduled KSARs will satisfy F3's objective and trigger Terminate-Prescription. It will deactivate F3 and the process of plan elaboration will resume.

PROTEAN integrates opportunistic focusing with hierarchical planning when other control KSs, triggered by intermediate solution states, insert focus decisions into its evolving control plan. BB1's Terminate-Prescription KS deactivates those focus decisions when their objectives are satisfied, just as it does during hierarchical planning. Section 4 illustrates opportunistic focusing.

III. Goal-Directed Reasoning in BB1

A. The Semantics of Goal-Directed Reasoning

Goal-directed reasoning entails identifying and performing actions in order to perform other desirable actions. These other actions may be desirable *per se* or because of their effects.

For example, suppose that PROTEAN wishes to perform actions of this type:

```
Yoke several long Helices in PA1
                    with Constraints.
```

Suppose also that the current agenda contains an appropriate yoking action:

```
Yoke Helix2-1 with Helix3-1 in PA1 with CSetH2H3.
```

but that it requires satisfaction of one precondition prior to execution:

```
Is-Anchored Helix3-1 to Anchor in PA1
                  with strong Constraints.
```

PROTEAN might reason backward from its goal (to perform the designated class of yoking actions), identifying a subgoal to perform actions that satisfy the precondition:

```
Perform Actions that Promote:
   Is-Anchored Helix3-1 to Anchor in PA1
                    with strong Constraints.
```

Now suppose that the agenda contains no actions that promote the designated state. PROTEAN might continue reasoning backward to identify a new subgoal. For example, it might determine that the action:

```
Anchor Helix3-1 to Helix1-1 in PA1 with CSetH3H1.
```

would promote the desired state if it were executed, and that the event:

```
Did-Include Helix3-1 in PA1.
```

would trigger that action. PROTEAN could then identify a new subgoal to perform actions that cause the designated event:

```
Perform Actions that Cause:
   Did-Include Helix3-1 in PA1.
```

As in all goal-directed reasoning, this regression through enabling conditions could continue indefinitely.

This simple example illustrates two key aspects of the semantics of goal-directed reasoning. First, the goal that initiates goal-directed reasoning need not be the ultimate goal of the problem-solving process, but can be any intermediate goal along the way. A good control architecture must be able to integrate goal-directed reasoning capabilities with any other reasoning method that might produce intermediate goals. Second, a "goal" can be distinguished as a desire to perform an action, cause an event, or promote a state. A good control architecture must exploit knowledge of the relations between different types of actions, events, and states to guide goal-directed reasoning. The BB1 mechanism for goal-directed reasoning meets these two objectives.

B. Generic Control KSs for Goal-Directed Reasoning

BB1 can initiate goal-directed reasoning in two situations: (a) the system notices that it has an important focus, but no executable KSARs that satisfy it; or (b) the system notices that it has a highly rated KSAR with unsatisfied preconditions. The first situation corresponds to conventional goal-directed reasoning; the system deliberately sets about enabling itself to perform desirable actions. The second situation differs in motivation; here, the system notices an opportunity to enable itself to perform desirable actions. When there is exactly one very important focus on the control blackboard, the two kinds of goal-directed reasoning will favor the same kinds of actions. However, when there are several foci of varying importance, the two kinds may favor different types of actions. For example, in the first situation, the system would favor actions that satisfy the single most important focus, while in the second situation, it could favor actions that get a high overall rating against the set of active foci. In this section, we describe generic BB1 knowledge sources for both situations.

BB1's generic control knowledge source, *Satisfy-Priority-Focus*, is triggered whenever no executable KSARs rate highly against an important focus. When executed, it determines what potential actions could rate highly against the focus. If triggered (not executable) actions on the agenda match the potential actions, Satisfy-Priority-Focus posts a goal-directed focus decision for each unsatisfied precondition:

```
Promote: <state>
```

If no matching actions appear on the agenda, Satisfy-Priority-Focus identifies knowledge sources that specify a matching action and, for each one, posts a goal-directed focus decision for each KS triggering condition:

```
Cause: <event>
```

If any executable KSARs rate highly against a goal-directed focus posted by Satisfy-Priority-Focus, the BB1 scheduler will choose them. If their actions produce the specified state or event, this will trigger Terminate-Prescription, which will deactivate the goal-directed focus. On the other hand, if no executable actions rate highly against the goal-directed focus, this will retrigger Satisfy-Priority-Focus. It will then post subgoal foci to trigger and satisfy the preconditions of knowledge sources that could rate highly against the prior goal-directed focus. Thus, Satisfy-Priority-Focus can reason backward through a chain of subgoals to trigger and satisfy the preconditions of knowledge sources that match high-priority foci. We have not yet implemented Satisfy-Priority-Focus.

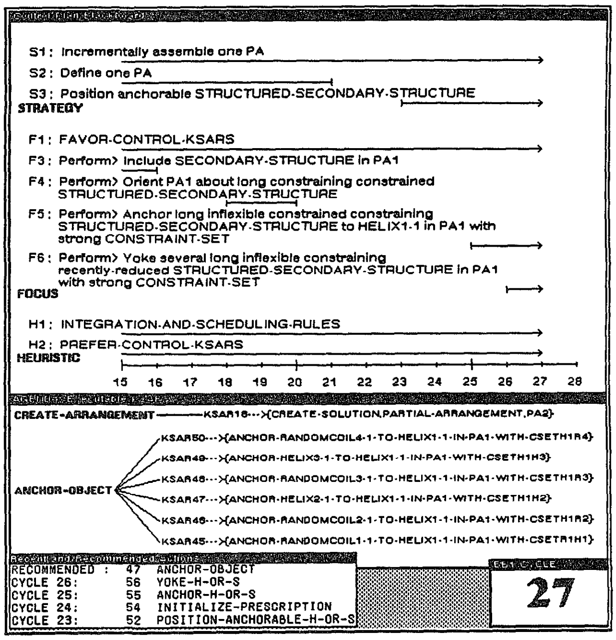
BB1's generic control knowledge source, *Enable-Priority-Action*, is triggered whenever a highly rated KSAR has unsatisfied preconditions. When executed, it posts a goal-directed focus for each unsatisfied precondition:

```
Promote: <state>
```

If any executable KSARs rate highly against a goal-directed focus posted by Enable-Priority-Action, the BB1 scheduler will choose them. If their actions produce the specified state, this will trigger Terminate-Prescription, which will deactivate the goal-directed focus. In addition, if the goal-directed focus raises the ratings of any triggered (but not executable) KSARs, this can retrigger Enable-Priority-Action, which will post another goal-directed focus for each of their unsatisfied preconditions. Thus, Enable-Priority-Action can reason backward through a chain of subgoals to satisfy the preconditions of high-priority actions. We have implemented Enable-Priority-Action and used it in several systems, including PROTEAN.

IV. Integrating Three Reasoning Methods

BB1 integrates diverse reasoning methods by permitting independent control knowledge sources to contribute decisions to an explicit, dynamic control plan on the control blackboard. For example, consider PROTEAN's reasoning during part of its work on the lac-repressor headpiece (figure 2). At cycle 27, PROTEAN has elaborated its hierarchical plan through sub-strategy S3. It is busy anchoring and yoking structured secondary structures in PA1.



```
  S1 : Incrementally assemble one PA
  S2 : Define one PA
  S3 : Position anchorable STRUCTURED-SECONDARY-STRUCTURE
STRATEGY

  F1 : FAVOR-CONTROL-KSARS
  F3 : Perform> Include SECONDARY-STRUCTURE in PA1
  F4 : Perform> Orient PA1 about long constraining constrained
       STRUCTURED-SECONDARY-STRUCTURE
  F5 : Perform> Anchor long inflexible constrained constraining
       STRUCTURED-SECONDARY-STRUCTURE to HELIX1-1 in PA1 with
       strong CONSTRAINT-SET
  F6 : Perform> Yoke several long inflexible constraining
       recently-reduced STRUCTURED-SECONDARY-STRUCTURE in PA1
       with strong CONSTRAINT-SET
FOCUS

  H1 : INTEGRATION-AND-SCHEDULING-RULES
  H2 : PREFER-CONTROL-KSARS
HEURISTIC
           15  16  17  18  19  20  21  22  23  24  25  26  27  28

CREATE-ARRANGEMENT ———KSAR18--->{CREATE-SOLUTION,PARTIAL-ARRANGEMENT,PA2}

                   KSAR50--->{ANCHOR-RANDOMCOIL4-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R4}
                   KSAR49--->{ANCHOR-HELIX3-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1H3}
ANCHOR-OBJECT      KSAR48--->{ANCHOR-RANDOMCOIL3-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R3}
                   KSAR47--->{ANCHOR-HELIX2-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1H2}
                   KSAR46--->{ANCHOR-RANDOMCOIL2-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R2}
                   KSAR45--->{ANCHOR-RANDOMCOIL1-1-TO-HELIX1-1-IN-PA1-WITH-CSETR1H1}

RECOMMENDED :   47   ANCHOR-OBJECT
CYCLE 26:       56   YOKE-H-OR-S
CYCLE 25:       55   ANCHOR-H-OR-S
CYCLE 24:       54   INITIALIZE-PRESCRIPTION
CYCLE 23:       52   POSITION-ANCHORABLE-H-OR-S                27
```

Figure 2: BB1/PROTEAN Control Blackboard and Executable Agenda at Cycle 27

On cycle 27, PROTEAN has two active hierarchical plan foci, F5 and F6. The BB1 scheduler chooses to execute KSAR47:

Anchor Helix2-1 to Helix1-1 in PA1 with CSetH1H2.

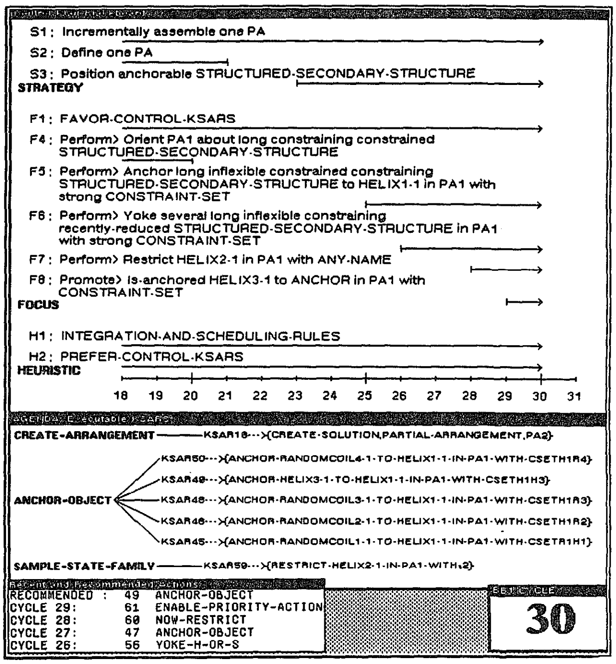
because it is the highest-priority (92) executable KSAR. Actually, KSAR34:

Yoke Helix3-1 and Helix2-1 in PA1 with CSetH2H3.

has a higher priority (196) because of PROTEAN's preference for yoking actions (focus F6), but it is not yet executable because it has an unsatisfied precondition:

Is-Anchored Helix3-1 to Anchor in PA1 with Constraints.

On cycle 28, a number of new KSARs are triggered, including two control KSARs. First, the anchoring of Helix2-1 produced an intractably large family of locations and this triggers PROTEAN's control KS, Now-Restrict. In the context of its hierarchical strategy, PROTEAN occasionally triggers Now-Restrict to post an *opportunistic* focus for *restricting* (statistically sampling the legal locations) of positioned objects. PROTEAN introduces this focus only when it identifies an unmanageably large family of locations for an object. Second, KSAR34's high priority triggers Enable-Priority-Action. Since PROTEAN has an overriding preference for control actions (focus F1), the BB1 scheduler chooses to execute these two KSARs on cycles 28 and 29, producing focus decisions F7 and F8 (figure 3). F7 is an opportunistic focus to restrict Helix2-1's family of locations. F8 is a goal-directed focus on actions that can satisfy KSAR34's outstanding precondition.



```
  S1 : Incrementally assemble one PA
  S2 : Define one PA
  S3 : Position anchorable STRUCTURED-SECONDARY-STRUCTURE
STRATEGY

  F1 : FAVOR-CONTROL-KSARS
  F4 : Perform> Orient PA1 about long constraining constrained
       STRUCTURED-SECONDARY-STRUCTURE
  F5 : Perform> Anchor long inflexible constrained constraining
       STRUCTURED-SECONDARY-STRUCTURE to HELIX1-1 in PA1 with
       strong CONSTRAINT-SET
  F6 : Perform> Yoke several long inflexible constraining
       recently-reduced STRUCTURED-SECONDARY-STRUCTURE in PA1
       with strong CONSTRAINT-SET
  F7 : Perform> Restrict HELIX2-1 in PA1 with ANY-NAME
  F8 : Promote> Is-anchored HELIX3-1 to ANCHOR in PA1 with
       CONSTRAINT-SET
FOCUS

  H1 : INTEGRATION-AND-SCHEDULING-RULES
  H2 : PREFER-CONTROL-KSARS
HEURISTIC
           18  19  20  21  22  23  24  25  26  27  28  29  30  31

CREATE-ARRANGEMENT ———KSAR18--->{CREATE-SOLUTION,PARTIAL-ARRANGEMENT,PA2}

                   KSAR50--->{ANCHOR-RANDOMCOIL4-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R4}
                   KSAR49--->{ANCHOR-HELIX3-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1H3}
ANCHOR-OBJECT      KSAR48--->{ANCHOR-RANDOMCOIL3-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R3}
                   KSAR46--->{ANCHOR-RANDOMCOIL2-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R2}
                   KSAR45--->{ANCHOR-RANDOMCOIL1-1-TO-HELIX1-1-IN-PA1-WITH-CSETR1H1}

SAMPLE-STATE-FAMILY ———KSAR59--->{RESTRICT-HELIX2-1-IN-PA1-WITH,2}

RECOMMENDED :   49   ANCHOR-OBJECT
CYCLE 29:       61   ENABLE-PRIORITY-ACTION
CYCLE 28:       60   NOW-RESTRICT
CYCLE 27:       47   ANCHOR-OBJECT
CYCLE 26:       56   YOKE-H-OR-S                              30
```

Figure 3: BB1/PROTEAN at Cycle 30

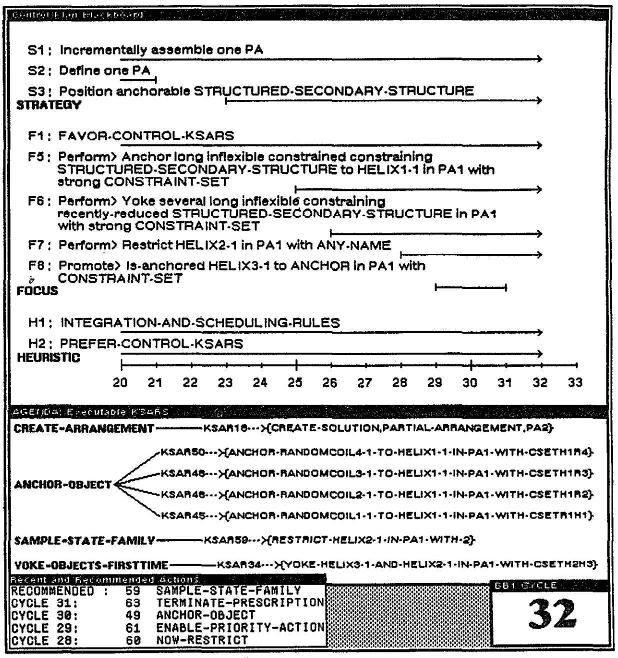
On cycle 30, the BB1 scheduler uses all five active foci to rate pending KSARs and chooses to execute KSAR49:

Anchor Helix3-1 to Helix1-1 in PA1 with CSetH1H3.

On cycle 31, the anchoring of Helix3-1 produces the state targetted by the goal-directed focus, F8. This allows goal KSAR34 to become executable and triggers Terminate-Prescription. The BB1 scheduler chooses to execute the Terminate-Prescription KSAR, deactivating F8.

On cycle 32 (figure 4), the scheduler uses the remaining four active foci to rate KSARs. It chooses to execute KSAR59:

```
Restrict Helix2-1 in PA1
                    with Sampling-Constraint-2.
```



STRATEGY

S1 : Incrementally assemble one PA

S2 : Define one PA

S3 : Position anchorable STRUCTURED-SECONDARY-STRUCTURE

FOCUS

F1 : FAVOR-CONTROL-KSARS

F5 : Perform> Anchor long inflexible constrained constraining STRUCTURED-SECONDARY-STRUCTURE to HELIX1-1 in PA1 with strong CONSTRAINT-SET

F6 : Perform> Yoke several long inflexible constraining recently-reduced STRUCTURED-SECONDARY-STRUCTURE in PA1 with strong CONSTRAINT-SET

F7 : Perform> Restrict HELIX2-1 in PA1 with ANY-NAME

F8 : Promote> Is-anchored HELIX3-1 to ANCHOR in PA1 with CONSTRAINT-SET

HEURISTIC

H1 : INTEGRATION-AND-SCHEDULING-RULES

H2 : PREFER-CONTROL-KSARS

CREATE-ARRANGEMENT — KSAR18--->{CREATE-SOLUTION,PARTIAL-ARRANGEMENT,PA2}

ANCHOR-OBJECT
- KSAR50--->{ANCHOR-RANDOMCOIL4-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R4}
- KSAR48--->{ANCHOR-RANDOMCOIL3-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R3}
- KSAR46--->{ANCHOR-RANDOMCOIL2-1-TO-HELIX1-1-IN-PA1-WITH-CSETH1R2}
- KSAR45--->{ANCHOR-RANDOMCOIL1-1-TO-HELIX1-1-IN-PA1-WITH-CSETR1H1}

SAMPLE-STATE-FAMILY — KSAR59--->{RESTRICT-HELIX2-1-IN-PA1-WITH-2}

VOKE-OBJECTS-FIRSTTIME — KSAR34--->{YOKE-HELIX3-1-AND-HELIX2-1-IN-PA1-WITH-CSETH2H3}

```
RECOMMENDED :   59   SAMPLE-STATE-FAMILY
CYCLE 31:       63   TERMINATE-PRESCRIPTION
CYCLE 30:       49   ANCHOR-OBJECT
CYCLE 29:       61   ENABLE-PRIORITY-ACTION
CYCLE 28:       60   NOW-RESTRICT
```

Figure 4: BB1/PROTEAN at Cycle 32

On cycle 33, the reduction in Helix2-1's locations achieves the objective of opportunistic focus F7 and triggers Terminate-Prescription. The scheduler chooses to execute it, deactivating F7.

On cycle 34, the BB1 scheduler uses the two remaining strategic foci, F5 and F6, to rate pending KSARs and returns to its planned anchoring and yoking activities. It first chooses the previous goal KSAR34, because of its high priority as determined by F5 and F6.

V. Discussion

We have developed a goal-directed reasoning mechanism that goes beyond the syntactic method of backward-chaining through rules [Buchanan and Shortliffe, 1984]. The BB1 mechanism follows semantic links relating actions, events, and states to determine which actions will achieve a specified goal.

In addition, the goal-directed reasoning mechanism operates in two conceptually different situations: deliberate efforts to perform particular kinds of actions; and detection of opportunities to perform generally desirable actions. The mechanism exploits BB1's control semantics (decisions to perform an action-class, cause an event-class, or promote a state-class in order to achieve specified objectives) and ACCORD's representation of the relations (cause, promote, trigger, enable) among particular actions, events, and states.

Finally, BB1 distinguishes itself from other control architectures in its ability to integrate diverse reasoning methods with a uniform mechanism. Although some systems permit multiple reasoning methods, they provide separate mechanisms that must be selected for any given problem-solving system [Genesereth and Smith, 1982, Nii and Aiello, 1979] or combined modally within a system [Newell and Simon, 1972, Pohl, 1969, Pohl, 1971, Rosenbloom and Newell, 1982]. For example, Corkill, Lesser, and Hudlicka's vehicle-tracking system [Corkill *et al.*, 1982] performed goal-directed reasoning whenever it could generate a goal, and otherwise performed data-driven reasoning. Although the Hearsay-II blackboard system [Erman *et al.*, 1980] integrated reasoning methods similar to those we have demonstrated in PROTEAN, it engineered each method in a different domain-specific tailoring of its underlying architecture. By unifying these several methods within a principled architecture, we make them available to many different application systems, with whatever form of integration is appropriate.

The utility of a control architecture depends upon three factors: (1) the architecture's functional capabilities; (2) the utility of these capabilities for particular application systems; and (3) the cost of these capabilities. In this paper, we have produced empirical evidence of BB1's value on the first two factors. We have shown that it supports three different reasoning methods (hierarchical planning, opportunistic focusing, and goal-directed reasoning) individually and in fully integrated combinations. We also have shown that at least one application system, PROTEAN, can usefully exploit integrated reasoning with all three methods. Other reports [Hayes-Roth *et al.*, 1986a, Schulman and Hayes-Roth, 1987] demonstrate the utility of BB1's capabilities for explaining and learning about its own actions — both of which rely critically upon its control architecture. Although we do not address the cost of BB1's control reasoning in this paper, that issue is addressed in detail elsewhere [Garvey *et al.*, 1987], with current evidence suggesting that the computational advantages of BB1's control reasoning outweigh the computational costs.

References

[Altman, 1986] R. Altman. *FEATURE.* Technical Report, Stanford, Ca.: Knowledge Systems Laboratory, Stanford University, 1986.

[Buchanan and Shortliffe, 1984] B.G. Buchanan and E.H. Shortliffe. *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project.* Menlo Park, Ca.: Addison-Wesley, 1984.

[Buchanan *et al.*, 1985] B.G. Buchanan, B. Hayes-Roth, O. Lichtarge, M. Hewett, R. Altman, P. Rosenbloom, and O. Jardetzky. *Reasoning with Symbolic Constraints in Expert Systems.* Technical Report, Stanford, Ca.: Stanford University, 1985.

[Corkill *et al.*, 1982] D.D. Corkill, V.R. Lesser, and E. Hudlicka. Unifying data-directed and goal-directed control: an example and experiments. *Proceedings of the AAAI*, :143–147, 1982.

[Davis, 1976] R. Davis. *Applications of Meta Level Knowledge to the Construction, Maintenance, and Use of*

Large Knowledge Bases. Technical Report Memo AIM-283, Stanford University Artificial Intelligence Laboratory, 1976.

[Durfee and Lesser, 1986] E. Durfee and V. Lesser. Incremental planning to control a blackboard-based problem solver. *Proceedings of the AAAI*, :58–64, 1986.

[Erman *et al.*, 1981] L.D. Erman, P.E. London, and S.F. Fickas. The design and an example use of Hearsay-III. *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, :409–415, 1981.

[Erman *et al.*, 1980] L.D. Erman, F. Hayes-Roth, V.R. Lesser, and D.R. Reddy. The Hearsay-II speech-understanding system: integrating knowledge to resolve uncertainty. *Computing Surveys*, 12:213–253, 1980.

[Garvey *et al.*, 1987] A. Garvey, C. Cornelius, and B. Hayes-Roth. *Computational Costs versus Benefits of Control Reasoning.* Technical Report KSL-87-11, Stanford, Ca.: Knowledge Systems Laboratory, Stanford University, 1987.

[Genesereth and Smith, 1982] M.R. Genesereth and D.E. Smith. *Meta-Level Architecture.* Technical Report HPP-81-6, Stanford, Ca.: Stanford University, 1982.

[Hayes-Roth, 1984] B. Hayes-Roth. *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.

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

[Hayes-Roth and Hayes-Roth, 1979] B. Hayes-Roth and F. Hayes-Roth. A cognitive model of planning. *Cognitive Science*, 3:275–310, 1979.

[Hayes-Roth *et al.*, 1986a] B. Hayes-Roth, A. Garvey, M.V. Johnson Jr., and M. Hewett. *A Layered Environment for Reasoning about Action.* Technical Report KSL-86-38, Stanford, Ca.: Knowledge Systems Laboratory, Stanford University, 1986.

[Hayes-Roth *et al.*, 1986b] B. Hayes-Roth, B.G. Buchanan, O. Lichtarge, M. Hewett, R. Altman, J. Brinkley, C. Cornelius, B. Duncan, and O. Jardetzky. Protean: deriving protein structure from constraints. *Proceedings of the AAAI*, 1986.

[Hewett and Hayes-Roth, 1987] M. Hewett and B. Hayes-Roth. *The BB1 Architecture: A Software Engineering View.* Technical Report KSL-87-10, Stanford, Ca.: Knowledge Systems Laboratory, Stanford University, 1987.

[McCarthy, 1960] J. McCarthy. Programs with common sense. *Proceedings of the Teddington Conference on the Mechanization of Thought Processes*, 1960.

[Newell and Simon, 1972] A. Newell and H. A. Simon. *Human Problem Solving.* Englewood Cliffs, N.J.: Prentice-Hall, 1972.

[Newell *et al.*, 1959] A. Newell, J.C. Shaw, and H.A. Simon. Report on a general problem-solving program. *Proceedings of the International Conference on Information Processing*, 1959.

[Nii, 1986] H.P. Nii. Blackboard systems. *AI Magazine*, 7(numbers 3 and 4):3:38–53 and 4:82–107, 1986.

[Nii and Aiello, 1979] H.P. Nii and N. Aiello. AGE [attempt to generalize]: a knowledge-based program for building knowledge-based programs. *Proceedings of the International Joint Conference on Artificial Intelligence*, 6:645–655, 1979.

[Pohl, 1969] I. Pohl. *Bi-directional and Heuristic Search in Path Problems.* Technical Report SLAC-104, Stanford, Ca.: Stanford University, 1969.

[Pohl, 1971] I. Pohl. Bi-directional search. In B. Meltzer and D. Michie, editors, *Machine Intelligence 6*, pages 127–140, New York, NY.: American Elsevier, 1971.

[Rosenbloom and Newell, 1982] P.S. Rosenbloom and A. Newell. Learning by chunking: summary of a task and a model. *Proceedings of the American Association for Artificial Intelligence*, :255–258, 1982.

[Schulman and Hayes-Roth, 1987] R. Schulman and B. Hayes-Roth. *ExAct: A Module for Explaining Actions.* Technical Report KSL-87-8, Stanford University: Knowledge Systems Laboratory, 1987.

[Stefik, 1981a] M. Stefik. Planning and meta-planning [MOLGEN: Part 2]. *Artificial Intelligence*, 16:141–169, 1981.

[Stefik, 1981b] M. Stefik. Planning with constraints [MOLGEN: Part 1]. *Artificial Intelligence*, 16:111–140, 1981.

[Terry, 1983] A. Terry. *Hierarchical Control of Production Systems.* PhD thesis, UC, Irvine, 1983.

[Tommelein *et al.*, 1987] I. D. Tommelein, M.V. Johnson Jr., B. Hayes-Roth, and R.E. Levitt. SIGHTPLAN: a blackboard expert system for the layout of temporary facilities on a construction site. *Proceedings of the IFIP WG5.2 Working Conference on Expert Systems in Computer-Aided Design, Sydney, Australia*, 1987.

[Weyrauch, 1980] R.W. Weyrauch. Prolegomena to a theory of mechanized formal reasoning. *Artificial Intelligence*, 13:133–170, 1980.