Knowledge Base Management for Experiment Planning in Molecular Genetics

Nancy Martin
Computing and Information Science
University of New Mexico
Albuquerque, New Mexico

Peter Friedland, Jonathan King. Mark Stefik
Computer Science Department
Stanford University
Stanford, California

The use of a representation language involving schemata and associated derived models has been extended to include all aspects of domain knowledge and strategy and heuristic problem solving knowledge. This uniform representation will allow the extension of knowledge base management techniques for acquisition and retrieval of procedural knowledge.

Descriptive Terms: genetics, heuristic problem solving, knowledge bases, MOLGEN, planning systems, representation of knowledge, rule-based systems.

1 Introduction

The design of MOLGEN is based on the proposition that successful problem solving in complicated domains requires the use of large amounts of domain specific knowledge. As a result, the representation of this knowledge is a problem of practical importance. The domain specific knowledge includes knowledge about the objects, the transformations applicable to the objects and information about use, effectiveness, and cost of the transformations. The representation should facilitate the acquisition of the knowledge from the user and the retrieval of relevant knowledge during problem solving.

An innovative feature of MOLGEN will be its ability to represent all problem solving knowledge in a uniform manner. In addition, the system design insures that:

1. Planning strategies and heuristics will not be built into the code of the system.
2. Procedures will be expressed in such a way that the contents of the named component parts of the procedures can be referenced.

The paper first gives a brief description of domain objects, transformations, and the design of experimental plans. Then Section 3 describes the representation of knowledge and the techniques being developed to manage the knowledge base. Finally the current state of implementation is discussed.

2 Experimental Tasks

MOLGEN will design both synthesis and analysis experiments. Synthesis experiments build a specified DNA structure from completely or partially specified starting materials. Analysis experiments discriminate among competing structural hypotheses. An example is given in Section 2.3.

* This research was supported by the National Science Foundation under Grants NSF MCS76-11935 and MCS76-116M9. J. King is supported by an IBM graduate research fellowship.

In this work we are assuming that the competing hypotheses are given to the system and the task is to discriminate among them. See [Feitelson77] for a description of knowledge sources for the systematic formation and testing of hypotheses in a recent genetics experiment.

The design of molecular genetics experiments may be viewed within a familiar AI paradigm: state-space search. World states for these searches consist of sets of DNA structures along with descriptions of an experimenter s knowledge about them. State transformations correspond to laboratory techniques which, alter structures or increase an experimenter's knowledge about structures. An actual experiment as carried out in the laboratory can be represented by an initial world state and a sequence of transformations. A laboratory problem can be represented by an initial world state and desired final world slate. However, an experimental plan could be more complicated than the trace of a particular experiment as it was performed in the laboratory. The plan must express the fact that at a given point in an experiment the next transformation may depend on the outcome of the present: transformation. It also must allow transformations to remain unordered when the order of application is not important. Thus, an experimental plan is a more complicated structure than a simple sequence of transformations.

2.1 Genetics Knowledge

2.1.1 Domain Objects

The science of molecular genetics is centered around the structure and function of the deoxyribonucleic acid (DNA) molecule. DNA consists of two parallel strands composed of chemical groups called "nucleotides". All nucleotides are composed of a sugar part which forms the DNA "backbone" and a base which participates in "hydrogen bonds" between the parallel strands. Only four bases commonly exist in the DNA molecule, Adenine, Guanine, Cytosine, and Thymine — abbreviated as A, G, C, and T. Furthermore. A and T will only form hydrogen bonds with each other; likewise for C and G.

DNA structures have biological, topological and sequential attributes. The molecule can be viewed as a chromosome whose component parts are enes having specific biological and control unctions. In addition to the double stranded topology described above, DNA has many possible configurations. Hydrogen bonds and backbone bonds can be broken resulting in structural anomalies such as nicks (single backbone bond broken), gaps (several bases missing along one strand), bubbles (a sequence of internal hydrogen bonds broken so the strands separate). The final attribute of the molecule is the actual sequence of bases used to form the individual strands. In addition to the four bases described above, there are alternate bases which can occur.

In any particular experiment, attributes of each type may be important* The geneticist often does not know enough about the structure of a molecule to know precisely where a feature, such as a gene or a specific base sequence, is on the molecule.

2.1.2 Domain Transformations

There are two types of domain transformations: modification and observation. By modification we mean a transformation which changes DNA structures. By observation we mean a transformation which changes the state of information available to the experimenter about those structures.

Modification transformations consist of making and breaking bonds and separating molecules. These actions can be carried out by a variety of physical, chemical and biological techniques. A commonly used group of naturally occuring reagents are called enzymes. These are used to catalyze reactions that make or break bonds

The specificities of the transformations vary over a wide range. For example, each restriction enzyme recognizes a specific base pattern and cuts the molecule at that specific site* In contrast, certain exonuclease enzymes cut the molecule from one end liberating fragments of unpredictable size.

Unfortunately, there is not a set of primitive transformations that will break or make any arbitrary specified bond on a molecule. A part of a molecule that is a conceptual entity to a geneticist, such as a gene or a region with a particular base sequence, may not be recognizable by any modification transformation. Therefore, for each situation the geneticist must find some combination of transformations that approximn'es the transformation desired.

A given modification transformation may produce many products in a single application and may not always carry out an identically reproducible action. An example is the enzyme Ligase which causes pieces of double stranded LNA to join forming longer pieces. The enzyme will not lust loin the two desired segments. The final sample will consist of segments of varying length and varying composition. If a particular composition is desired, it must be recognized and separated from the sample by another transformation.

Observational transformations are particularly important to the geneticist. They are used to verify or predict the results of modifications. Common observational transformations include: analytic electrophoresis (a laboratory technique which discriminates among DNA molecules based primarily upon their molecular weight and electric charge), ultraviolet fluorescence (exposing a sample of molecules to ultraviolet radiation to determine if a structure is single or double stranded). and electron microscopy (,to observe topological details).

2.2 *G*enetic Abstractions

The actual laboratory transformations are often nonspecific both in terms of the objects they act upon and the results they produce. This characterization of the domain transformations leads naturally to the use of abstract objects and transformations in planning experiments. The abstractions correspond to the conceptual entities and transformations of the geneticist and will be an important part of the knowledge base.

Similarly, the planning strategies used by the geneticist in creating a plan with these abstractions are important heuristics for the system. This encompasses a broad range of knowledge such as plan sketches for various contexts, design cost heuristics which predict the costs or transformations, and heuristics for evaluating the relevance and specificity of laboratory transformations to the current problem.

*2*3* Analysis Example

One method of analysis planning consists of focussing initial attention upon some feature of the problem and creating a model which emphasizes the attributes of that feature. In this example we illustrate how different attributes of DNA can lead to different plans for the same experimental task. The problem is to discriminate between the two hypotheses:

Hypothesis 1: The sample contains molecules with the sequence AAAAAAA...(called poly A), where the length of the poly A sequence is at least thirty bases.

Hypothesis 2: The sample contains no such molecules.

The usual focus in such a problem would be the poly A sequence.

Planl: One attribute of a long poly A sequence is that it will have weaker hydrogen bonds than other regions. This knowledge is easily deduced from the fact hat A's form considerably weaker hydrogen bonds with T's than do C s with G's. Exploiting this weakness leads to the idea of partially denaturing (breaking the weaker hydrogen bonds in) the molecules and looking for the resulting bubble shape under an electron microscope.

Plan2: Recognizing that an appropriate complementary base sequence would bind selectively to an embedded poly A region leads to a "probe" plan, A probe of radioactively labelled poly T (the complementary sequence) is obtained. The probe is mixed with the DNA sample. Finally, an observation technique is used to see if radioactivity has been incorporated into the DNA molecules.

Plan3: A final perspective makes use of the fact that the biological function of DNA is to create proteins. In the case of Poly A, the protein produced would be poly-valine. The experimental plan is to create proteins from the DNA sample and test for the presence of poly-valine.

3 Representation of Knowledge

We can summarize the knowledge which MOLGEN will use in experiment planning as follows:

Static Knowledge— Knowledge which is fixed during problem solving.

(1) Object: Conceptual entities of the system like DNA molecules, enzymes, samples, lab techniques.

(2) Action: World State transformations corresponding to the effects of laboratory techniques.

(3) Strategy/Control: Genetic planning strategies and general problem solving heuristics.

Dynamic Knowledge— Knowledge which can change during problem solving.

(4) World State: A description of the simulated genetics experimental environment and measurements of it.

(5) Planning State: A representation of the partially designed experiment and the activity of the problem solving process.

A similar classification of knowledge applies for many problem solving systems. Besides representing objects and actions, all systems must represent problem solving strategies and states. These are often embedded in the program or control structure rather than being explicit entries in the knowledge bases. However, embedding knowledge in this way may seriously impair the system s flexibility. Changing, adding, or deleting an instance of any type of problem solving knowledge can have wide ranging effects on knowledge of all types throughout the system. In a large knowledge base, it is vital for the system to assist the user in locating these effects. In addition, any system which will be extended to work on new problems must have facilities for adding new instances of each category of knowledge.

3.1 Common Representation for All Problem Solving Knowledge

To represent these types of knowledge, we adopt the use of a structured, object-centered knowledge base. This idea has recently been advanced for problem solving in KRL [Bobrow77] and for knowledge acquisition by TEIRESIAS [Davis76]. building on the ideas, among others, of SIMULA classes [Dahl72], FRAMES [Minsky74], and SMALLTALK classes [Goldberg76]. The main components of these representations, variously called "frames", "units", or "schemata", can be thought of (following [Davis76]) as record definitions extended to support interrelationships and to accommodate attached procedures. We will refer to them as "schemata" , Each schema will be composed of slots with associated value or type specifications and attached procedures. All schemata will be organized in a generalization/specialization hierarchy similar to that of other systems. Associated with each schema will be a model which summarizes the ways in which the various slots have been filled. These schemata and their associated models provide the knowledge needed by management routines for acquiring and updating the knowledge base and for many problem solving tasks. The issues involved in representing knowledge in such a schema system are similar to the issues when the representation is a semantic net. See [Woods75] and [Brachman76] for discussions of these issues.

A novel aspect of the design of the MOLGEN knowledge base is the representation of all types of problem solving knowledge in a common formalism — as instances of schemata. Procedural knowledge will be represented in such a way that the system can easily inspect any procedure. The schema system provides a mechanism for breaking procedures into component parts which can be addressed separately, and are thus accessible to the system (see section 3.2). This will aid acquisition and use during problem solving. Schemata for procedural knowledge are discussed more fully below.

The information gathering process at the beginning of a problem solving session is similar to the process of acquiring new instances of domain objects and transformations. By using a uniform representation, the gathering of this initial information from the user will be handled by the same mechanisms which acquire genetic knowledge for the permanent knowledge base.

3.1.1 The Schema Hierarchy and the Creation of

The geneticist classifies domain objects and processes according to their common properties. For example, all physical measurements are grouped together; all enzymes are grouped together; all chemical techniques are grouped together. The schemata will be organized in a hierarchy that reflects these classifications. Schemata will be linked from most general to most specialized category, with specializations inheriting designated properties (slots,restrictions on their values. attached procedures) from their generalizations.

Every individual entity in the system will be created as an instance of some schema (known as its "prototype"). The instance will inherit its set of slots from its prototype schema and from the generalizations of the prototype. The values filling the slots must also obey the restrictions specified in the prototype and its generalizations. These values will be instances of other schemata. For example, we might have a "growth medium" slot in the schema for "culture growth" which will be filled by an instance of the schema "gel"; the "resolving power" slot of the schema for "electrophoresis" (an analytical technique) might be filled by an instance of the schema for "real number".

3.1.2 Grouping Entities bv Function

The geneticist's classification hierarchy does not always reflect the functions that domain objects have in common. For example, restriction enzymes and ligases are both classes of enzymes, but the former performs cutting actions on DNA, while the latter performs sealing actions. From the point of view of seeking a tool to perform cutting during an experiment, restriction enzymes are more closely related to some physical methods than to enzymes such as ligase. Functional relationships will be represented separately from the classification hierarchy. The schema hierarchy is not meant to indicate all possible relations among entities or all possible ways in which something can be viewed.

3.1.3 Derived Models of Typical Instances of Schemata

The TEIRESIAS program [Davis76] maintained tabulations of the individual and joint occurrences of specific tokens within the premises and actions of each rule type. These tabulations were called "rule models". They were used to create expectations about the contents of new rules. Models are derived from actual instances and record the values which have been entered in each slot of a particular schema. We will use models for prompting the user for possibly missing information or to suggest default values. For example, the derived model for the "initial world state" schema might note that eighty percent of the instances which mention pH of the sample also mention the concentration of magnesium ions. A user who creates an initial world state mentioning pH but not magnesium concentration could be asked whether that should be included too, based on the correlation of occurrences.

3.1,4 Uses of the Sehema Hrarchy

A major use of the schema hierarchy will be to guide the acquisition of all types of domain knowledge. This use of the schema hierarchy is motivated by its use in the Teiresias system. In the process of entering a new enzyme, a user will be guided through the hierarchy, from objects to enzymes to a particular class of enzymes, until the proper prototype is located. If the necessary prototype does not exist, the system will guide the user in creating it and inserting it correctly in the hierarchy. The prototype will then give instructions to the system on how to prompt the user for information about the new instance, including documentation. The prototype also will give instructions on how to incorporate the new instance into system lists or tables as necessary.

A second important use of the schema hierarchy will be for organizing retrieval of objects matching particular descriptions during problem solving. Derived models can play a role in increasing the efficiency of this search process. By comparing the desired description with the statistical information in the model, a measure of the likelihood of finding an instance matching the description can be obtained. When a match is unlikely, the search could be directed to another branch of the hierarchy.

3.2 Schemata for Procedural Knowledge

One of the novel aspects of the design of the MOLGEN knowledge base is the use of schemata to represent procedural knowledge. Instances of these schemata will be the "rules" that express transformational and problem solving knowledge. In most similar systems, the knowledge about rule types is generally in the program code. These rules range from general QLISP procedures to highly restricted formats. For example, TEIRESIAS expects all rules to be "IF-THEN" rules. Knowledge about the two named components of the rule, the premise and the conclusion, is in the control program. Using this knowledge, the system can prompt the user for information to fill these components and can retrieve rules by reference to the content of the components. Rather than express rules in a traditional production system format, we use explicit iteration statements, conditional statements and assignment statements. The granularity of the rules depends on the type of rule. Rules expressing modification transformations will be relatively large, reflecting the genetic process being modeled. Planning rules will vary in size from small traditional rules to more complex rules. Our design extends TEIRESIAS' knowledge management capabilities of acquisition and content referencing to handle the complex types of rules. To do this we will use the slot mechanism. One slot will correspond to each component of a rule and may be filled by instances of other entities — that is, oblects or other operations. This design is used for all procedures in the system from predefined system operations to the domain rules which will be entered by the user. For example, the components of an iteration primitive (FOREACH) are described by slots corresponding to an ITERATION-LIST and an ITERATION-BODY. Slots of domain rules correspond to domain relevant components as illustrated by the schema for detection techniques described in the next section. In both cases specialized schemata will guide the filling of the slots.

As a result of this representation paradigm, strategies can be written to retrieve rules by reference to the content of named components. This will insure that rules are not incorrectly retrieved because of a faulty external descriptor. Also, it will be possible to retrieve a rule using a variety of patterns, rather than by lust a single name. Content referencing is motivated by a need to associate a new rule with appropriate problem solving contexts without specifying those contexts at the time the rule is acquired.

Rule retrieval by means of content matching may be very inefficient in a large system. In some instances we will precompute retrieval by associating with each strategy rule a "ruleset" containing the rules it most frequently retrieves. Each ruleset will have an attached pattern which will be checked against a newly entered rule to check whether it should be added to the ruleset.

3.2.1 A Schema for Detection Tecftnjqv^g

An important class of actions in the genetics domain is what we term "detection techniques". The detection rule schema (see figure 1) summarizes MOLGEN s representation for a class of laboratory techniques which measure properties of DNA structures. When a MOLGEN rule for a particular detection technique is executed, it will transform a world state instance by recording the result of the observation. For example, a detection technique might increase the experimenter s knowledge about some of the topological features or the structures such as their length or substructures. This new knowledge must be added to the world state.

In this example of a domain transformation, we will illustrate the kind of knowledge contained in the rule, the organization of the schema for the rule and the way that the creation of the rule as an instance of a schema lets it be retrieved in more than one way.

A simplified schema for detection techniques is shown in Figure 1. The bookkeeping slots provide a place for documentation of the knowledge base. Next are slots which must be filled when an instance of the schema (a specific detection rule) is created. Adiacent to the slot names are type restrictions. The slots name those parts of the rule which are determined by the instances of the rule. In this example, there are three such slots: Preparatory-Method whose value is to be an instance of the Preparatory-rule schema; Min-Sample-Size whose value is to be an instance of the Mass schema; and Feature-List whose value is to be a list of instances of the DNA-Features schema. In acquiring an instance of the Detection-Technique schema, only these slots need be filled. An instance of the Detection-Technique schema is given in Figure 2. The procedural part of the schema is contained in the form slot. The code in Figure 2 in the Form slot consists of the Figure 1 code with the appropriate substitutions made for the named components.

The process of creating the prototype FORM is more complicated. It involves using the schemata for each of the component parts of the rule.

3.2.2 The OBSERVE Operation

An important part of the Detection-Technique rule is the operation "OBSERVE". OBSERVE is a function that makes world state changes that reflect an increase in the experimenter's knowledge at any point in the experiment. OBSERVE has two basic actions: first it must call the appropriate procedures to make an observation on One world state. Then, it must make appropriate changes to the world state to reflect this observation. This p cess is guided by the schemata for the arguments of the function. In this case, each DNA feature which can be observed will have an associated pattern matching rule which can check for the feature in a given structure. For example, many of the DNA-Features will be instances of the schema for DNA-Nucleotide-Graph and these schemata will inherit a pattern matching rule from their prototype which performs a subgraph matching process. This association of the pattern-matching procedure with the pattern to be matched is similar in philosophy to constructions in the SMALLTALK [Goldberg76] system and will help make the OBSERVE function quite general.

3.2.3 Multiple Access by Means of Content

Decomposing a rule into accessible slots will make it retrievable by content for different purposes. We will illustrate this with the rule for Electron-Microscopy (figure 2), This rule is an instance of Detection-Technique schema. Like other types of detection techniques, electron microscopy is used for detecting certain features in molecules. Its ability to detect features is limited by various constraints, illustrated in our simplified example by the slots Min-Sample-Size and Feature-List. The Detection-Technique schema (figure 1) emphasizes both how its instances are similar and how they differ. Thus, in an experiment where two samples must be distinguished, a strategy rule might choose a detection technique which can work with a very small sample size if the sample material is precious. Alternatively, if the sample is large, the detection technique which can recognize the broadest range of features might be desirable.

3.3 Example Schew and. Instance

```
—BOOKKEEPING SLOTS FILLED AT SCHEMA CREATION—

Name:             Detection-Technique
Generalization:   Laboratory-Technique
Description:      "Rules for detecting structural
                  features of DNA"

---SLOTS TO BE FILLED WHEN CREATING AN INSTANCE—

Preparatory-Method:     Preparatory-Technique
Min-Sample-Size:        Mass
Feature-List:           LIST OF DNA-Feature

Form:
 IF WS-SAMPLE.Size > Min-Sample-Size

 THEN  APPLY Preparatory-Method TO WS-SAMPLE
       FOREACH Structure IN WS-SAMPLE.Structures
          FOREACH Feature IN Feature-List
             OBSERVE(Feature,Structure)
```

Figure 1
** Schema for Detection Technique **

(Comment on notation: the notation A.B refers to the contents of slot B of schema A.)

```
—BOOKKEEPING SLOTS FILLED AT SCHEMA CREATION-
NAME:            Election-Microscopy
Instance-of:     Detection-Technique
Description:     "Technique for observing and
                  measuring structural features."

—SLOTS FILLED WHEN RULE WAS CREATED—

Preparatory-Method:   Protein-Stain
Min-Sample-Size:      1 microgram
Feature-list:         (bubble, hairpin-loop,
                       circular.single-stranded,
                       double-stranded,length,...)

Form:
 IF  WS-SAMPLE.SIZE > 1 microgram

 THEN  APPLY  Protein-Stain TO WS-SAMPLE
       FOREACH Structure IN WS-SAMPLE.Structures
          FOREACH Feature IN  (bubble,
                     hairpin-loop,circular,...)
             OBSERVE(Feature,Structure)
```

Figure 2
Electron-Microscopy Rule

3.4 Procedural Attachment

Procedural attachment is a versatile mechanism for managing an object-centered knowledge base. The idea [Bobrow77] is to associate the procedure which performs a particular operation with the specification of the data objects involved. In KRL[Bobrow77], attached procedures are associated with the schemata for obiects and can be inherited by instances* The KRL work emphasized the use of procedural attachment for general use in problem solving. In TEIRESIAS, "slot-experts" played an important role in knowledge acquisition. In MOLGEN. attached procedures can be activated when acquiring a new prototype or instance; during pattern matching; when trying to fill a slot during problem solving; after a slot has been filled: when accessing a value for a slot; and when printing values. An attached procedure can either be a complicated program as in example 1 below or a rule in the knowledge base as in example 5, We give a few examples of the use of attached procedures:

Acquisition of prototvpe or instance

(1) SPECIAL EDITORS: Acquiring schema instances from the user will usually be guided by a general procedure interpreting from the schema what information the user must provide. However, some objects will have fairly complicated structures. For example, models of DNA will be graph structures with segments represented by nodes and chemical bonds represented by links. For these objects it will be useful to attach special procedures to their schemata to guide acquisition. We have already built and tested such a program for acquiring DNA structures. It is a picture-oriented editor and has proven easy for geneticists to use.

(2) RESTRICTION EXPERTS: The schema hierarchy discussed in a previous section is based on a strict generalization/specialization relationship between schemata. This is reflected not only in the inheritance of slots but also in the inheritance of restrictions on the values for slots. We will require that the value specification in the slot of a specialization be more restrictive than the value specification in the corresponding slot of the generalization. The question of whether one value specification is more restrictive than another is difficult to determine in a general way. For example, a list specification is more restrictive than another if it is a subset of the latter; a real number interval specification is more restrictive than another if its endpoints are contained within the other; in some cases a DNA graph specification might be considered to be more restrictive than another graph specification if it contains the other as a subgraph. This variety of comparisons between value restrictions suggests that the restriction comparison should be associated with the schema for the data type being restricted.

(3) PRE-COMPUTING ASSOCIATIONS: The MOLGEN knowledge base design has emphasized flexible content reference as a means for accessing domain transformation rules. This can be the basis for strategic selection of domain rules during the planning process and is motivated by a need to (1) index new rules to existing strategies and (2) permit new strategies to index existing rules in new ways. Recognizing the inefficiencies of content referencing during problem solving, we will pre-compute the indexing by associating rule sets with strategies. Each rule set could have attached to it an attached procedure which would consist of a content referencing rule. These attached procedures could be activated whenever a new rule is added to the knowledge base to check for each rule set whether the new rule should be added to the set. Other procedures attached to the ruleset will check whether updating is necessary when a rule is deleted or modified.

Matching

(4) SUBGRAPH MATCHER: Another example of procedural attachment was discussed in the example of a Detection-Technique schema. In that example, "observation" procedures were attached to features which could check a given structure for the DNA-feature. Because many of the features were instances of a DNA-graph, they inherited their observation procedure from their generalization (in those cases, the observation procedure is a DNA subgraph matcher).

when Filled

(5) SIDE EFFECTS: The attached procedure may also be a "rule" in the knowledge base which describes the result of changing the value of a parameter. Many properties or samples can be changed in numerous ways, and changing the properties can have many effects on the rest of the sample. For example, sample pH can be altered by the addition of a wide range of chemicals. Once pH is changed, it can alter the activity of every enzyme in the sample and denature many structures (cause their hydrogen bonds to break). These changes of course can greatly affect the course of an experiment. It would be possible to

associate the procedures to carry out the effects of changed pH with each of the chemicals that changes it. Alternatively, procedures might be associated with each structure or enzyme which could possibly be changed by pH, and these could check for altered pH whenever properties of the structure or enzyme were checked, changed, or used. Neither of these alternatives appears to be as natural or efficient as attaching a procedure to the schema for pH. which carries out the appropriate alterations in world state whenever pH itself is changed.

(6) PLAN CRITICS: New world state conditions which may affect the plan could be detected by attached procedures which offer advice on how to alter the plan to take the conditions into account. Such attached procedures could be called "domain specific plan critics". For example, suppose that sample temperature had been altered and as a result, certain structures had become denatured (the effect of denaturation being carried out by attached procedures as indicated above). A plan critic might be activated after the effects of temperature had been propagated through the world state. It might examine the new situation and conclude that the continuation of the plan required that the affected structures be renatured (hydrogen bonds resealed). It might be possible to provide the critic with specific repairs to specific problems. For example, it might know that if structures of a given type are denatured by raising the sample temperature, and if the pH is in a given range, then the structures can be renatured by adding a calculated quantity of a certain ion,so that the plan can be continued.

4 Implementation

Many of the utility programs for accessing and modifying the knowledge base have been implemented, A special editor, EDNA, for acquiring and manipulating DNA structures is completed and has been tested and used by geneticists. An initial schema editor has been created. The initial sets of genetic objects and transformations have been determined.

The first system will carry out the task of experiment checking. This means that a set of input samples and a specific sequence of transformations will be given. The system will then simulate the sequence of transformations on the representations of the samples terminating with a final set of samples. The final samples can be compared with actual laboratory results as a test of the initial hypotheses or of the accuracy of the transformations in the knowledge base. This first system will be used for debugging the transformation knowledge base and by geneticists for comparing the predicted results from the MOLGEN system against actual laboratory experiments.

5 Summary

The use of a representation language involving schemata and associated derived models has been extended to include all aspects of domain knowledge and strategy and heuristic problem solving knowledge. This uniform representation will allow the extension of knowledge base management techniques for acquisition and retrieval of procedural knowledge.

As with any problem solving system, the success will depend on the knowledge it has available. The purpose of our design is to create a system which can facilitate the process of transferring knowledge from the user to the system, and use this knowledge effectively in problem solving.

5.1 Acknowledgments

Other members of the Stanford community have contributed time and ideas to the MOLGEN project. We particularly want to thank Bruce Buchanan, Randy Davis, Ed Feigenbaum, Jerry Feitelson, Joshua Lederberg, Nils Nilsson and Terry Winograd.

6 BIBLIOGRAPHY

Bobrow D.G. , Winograd T. An Overview of KRL. a Knowledge Representation Language. Cognitive Science No.1 Vol.1 (Jan 1977)

Brachman R.J., What's in a Concept: Structural Foundations for Semantic Networks, BBN Report No. 3433 (October 1976)

Dahl O.J., Dijkstra E,, Hoare C.A.R, Structured Programming. New York: Academic Press (1972)

Davis R. Applications of Meta Level Knowledge to the Construction, Maintenance, and Use of Large Knowledge Bases. AI Memo 283. Computer Science Department, Stanford University. (July 1976)

Feitelson J,. Steflk M.J., A Case Study of the Reasoning in a Genetics Experiment. Heuristic Programming Project Memo HPP-77-18 (working paper) (May 1977)

Goldberg A., Kay A., Learning Research Group. SMALLTALK-72 Instruction Manual. Xerox Corporation (1976)

Minsky M. A Framework for Representing Knowledge In B. Winston Winston (Ed.) The Psychology '^of Computer Vision New York: McGra

Stefik. M.J., Martin N. A Review of Knowledge Basea Problem Solving As a Basis for a Genetics Experiment Designing System, CS Report 77-596 Computer Science Department, Stanford University. (Feb 1977)

Woods W.A., What's in a Link: Foundations for Semantic Networks, in Representation and Understanding, Sidles in Cognitive Science, Bobrow and Collins (eds.) (1975)