Report 85-07
Stanford -- KSL

Intelligent Computational Assistance for Experiment Design. Rene Bach, Yumi Iwasaki, Peter E. Friedland, 1985

card 1 of 1

Knowledge Systems Laboratory
Report No. KSL 85-7

1985

Intelligent Computational Assistance for Experiment Design

by

Rene Bach, Yumi Iwasaki, and Peter Friedland

Computer Science Department
Stanford University
Stanford, California 94304

Intelligent Computational Assistance for Experiment Design

René Bach, Yumi Iwasaki, and Peter Friedland

MOLGEN Project
Computer Science Department
Stanford University
Stanford, CA 94305

ABSTRACT

We have developed an automated system for the design of laboratory experiments in molecular biology. The system uses a planning method known as *skeletal plan refinement* that attempts to emulate the human cognitive task of experiment design. This paper describes the theory, history, and implementation of the design system and illustrates its function in the domain of DNA cloning experiments.

INTRODUCTION

The MOLGEN project is an eight-year collaborative effort among computer scientists and molecular biologists at Stanford University to explore computational problem-solving methods within the domain of molecular biology. A fundamental theme of the research has been the application of artificial intelligence methodologies to the problem of experiment design.

Experiment design is the process by which a scientist produces a detailed *plan* for conducting a laboratory experiment. For an analysis experiment, the plan consists of a series of actions that will elucidate some structural or functional feature of interest. For a synthesis experiment, the plan consists of a list of steps to construct a new structure.

Automating the process of experiment design is of interest and value for the two fields of computer science and molecular biology. For computer science, the process of experiment design is a difficult cognitive task that involves fundamental issues of knowledge acquisition, knowledge representation, problem-solving inference methods, and interaction with human experts. For molecular biologists the potential benefit comes from propagating experiment design expertise to those less expert, from combining the expertise of several individuals for the design of an experiment that spans several specialties, and from providing a level of thoroughness and unbiased technique selection that is very difficult for humans to achieve.

The precise domain of molecular biology under current study is the design of cloning experiments. This area was chosen because it is large enough to provide a reasonable expectation that a working system could be generalizable to other problems in molecular biology (and other disciplines), small enough to provide the *bounding* on the problem that makes construction of a knowledge base practical in a relatively short period of time, and interesting enough to current laboratory researchers to attract the molecular biology expertise needed to make the system successful on real problems.

The remainder of this paper will discuss the evolution of a problem-solving system for the task of experiment design. It will first describe one theory of how human scientists plan experiments and explain how a method embodying that theory was implemented within an automated system. (Readers interested in complete details should refer to [1] and [2].) Then the *knowledge base* that is central to the system will be detailed, followed by an example of the system solving a problem in the cloning domain. Finally, the current status of the system will be given along with a discussion of future work needed to complete a system that will be of practical utility.

THE SKELETAL PLAN DESIGN THEORY

An early lesson of research in artificial intelligence was that studying how human experts perform is a good initial step in the construction of an automated system to emulate that performance. Therefore, the research in automated experiment design began by carrying out extensive interviews over a one-year period with several expert molecular biologists in various departments at Stanford. In addition, the experts supplied dozens of literature references to other experiments that were thought of as "well-designed." The result of this informal study in human problem-solving behavior was a theory called *Skeletal Plan Refinement*.

The theory can be concisely stated as "Scientists rarely design things from scratch." They seem to first search for a strategy, an abstracted laboratory experiment we term a *skeletal plan*, that was useful for some related experimental goal. Then, they convert the skeletal plan into an actual experiment design by *refining* each step of the skeletal plan with an appropriate laboratory method for the specific goal and molecular and chemical environment of the experiment. The skeletal plan may be highly specific if the goal is very close to one for which a good experiment has already been designed. At times, it may be extremely general; a skeletal plan like "label the structural feature you are looking for and then look for the label" might lead to an experiment design if nothing more specific can be found.

An example from the cloning domain may help clarify the experiment design process. The list of abstract steps, shown below, forms the skeletal plan for an enormous variety of different cloning experiments.

1. Isolate the DNA to be cloned
2. Select a vector
3. Join the DNA and the vector

4. Select a host
5. Insert the recombinant molecule into the host
6. Select the clones.

The skeletal plan shown for cloning was discovered once and is converted into an actual experiment by choosing the appropriate objects and techniques to refine each step. An example of an actual experiment that results from the skeletal plan is:

1. Make a cDNA fragment using reverse transcriptase and DNA polymerase
2. Choose PBR322 as the cloning vector
3. Use a BAM linker and then T4-DNA ligase to join the fragment to the vector (this inactivates the tetracycline resistance gene while leaving the ampicillin resistance gene intact)
4. Choose E. coli as a host
5. Use cell transformation to insert the recombinant into the host
6. Select, among the cells resistant to ampicillin and sensitive to tetracycline, the clones of interest using RNA hybridization.

It should be noted that the skeleletal plan refinement design method has a general applicability to a variety of fields. For example, all good cooks have a "knowledge base" of recipe outlines which they make more specific to meet a variety of particular situation. The key difference between this method and previous artificial intelligence work in design is the emphasis on the knowledge used to refine individual steps, rather than on devising a complex inference method to produce the abstract outline of the design in the first place.

Skeletal Plan Selection

The process of finding a skeletal plan or strategy for solving a given problem is common to many disciplines. George Polya, in his book on mathematical problem-solving, *How to Solve It* [3], described "devising a plan" as follows:

> Have you seen it before? Or, have you seen the same problem in a slightly different form? Do you know a related problem? . . . Could you imagine a more accessible related problem? A more general problem? A more special problem?

Skeletal plans exist at many levels of generality. At the most general level, there are only a few plans, but these are used as "fall-backs," when easier to refine, more specific plans cannot be found. The problem is not just one of finding a plan that might provide a satisfactory solution, but finding a plan that will require the least refinement work. The skeletal plan finding process reduces to simple look-up when exactly the same problem has been solved before (even if on a completely different set of laboratory and molecular conditions), but becomes more difficult when only related problems have been solved. Then, the task may be in deciding whether to choose a detailed plan for a related problem, or a more general plan for a class of problems.

Skeletal Plan Refinement

Refining a skeletal plan means picking an appropriate "ground-level" instantiation for each step in the abstract plan. Scientists use three major criteria in making the refinement choices. In order of priority of application, they are:

1. Will the technique, if successfully applied, carry out the specific goal of the step? For example, will the separatory method chosen specifically separate linear from circular DNA if that was the goal of the step?

2. For all techniques which satisfy the first criteria, which ones can be successfully applied to the given molecule under the given laboratory conditions? In chemical terms, will the step "go?"

3. For all techniques which pass the first two tests, which one is "best?" This choice point, while perhaps the least important (since all techniques which make it to this point will do the required job and will work in the laboratory enviroment of the problem), seems to be the hardest for scientists to adequately define. It involves such metrics as reliability, convenience, accuracy, cost, and time to carry out a given technique. A computational system which models this process must also take into account the personal nature of the this decision and allow for different users to choose different heuristics.

The process of plan-step instantiation is aided enormously by the hierarchical nature of knowledge in molecular biology. An experimental scientist knows much about laboratory techniques and how to make choices among them. From our study of the process of experiment design by humans, it would appear that the knowledge about techniques is not randomly ordered by technique, but logically arranged in a taxonomy of techniques.

For example, consider the case of Exonuclease III. Exo III can be thought of as belonging to the following hierarchy:

```
Laboratory Techniques
          Modification Methods
                    Degradation Methods
                              Enzymatic Degradation
                                        Exonucleases
                                                  Double-Stranded Exonucleases
                                                            Exo III
```

From the scientist's knowledge that Exo III is a modification method, he knows that it will make some changes to nucleic acid structures. From the fact that it is a degradation method, he knows that it digests one or both strands of a molecule. The fact that it is an enzyme confers certain chemical properties on the technique. It is an exonuclease which means it only attacks nucleic acids at ends, nicks, or gaps. It is a double-strand specific exonuclease, so it attacks only double-stranded nucleic acids. Finally, the scientist may know (or more likely may look-up in the literature) properties unique to Exo III like optimum pH and temperature.

When a scientist chooses a technique like Exo III to satisfy a given plan goal, he proceeds down the hierarchy in making his selection. He may decide that he wants, in order to refine a plan-step, to degrade some double-stranded DNA at the ends. He uses his knowledge about choosing degradation methods--enzymes are much more substrate-specific than physical methods of degradation--to pick enzymatic degradation. His selection heuristics for enzymes lead him to exonucleases since he wants to degrade only at ends. His heuristics about exonucleases make him chose a double-strand specific exonuclease since his substrate is double-stranded. Finally, he may pick Exo III for a variety of reasons specific to his molecule or laboratory conditions; Exo III may be the most reliable, cheapest, or simply the most available at the moment.

The essential point is that the hierarchical nature of the scientist's knowledge of laboratory techniques provides a much more efficient means of storing information (and therefore allows him to retrieve more information about more techniques) than if each technique were considered an independent entity. The heuristics used in problem-solving are designed to allow an easy flow through the hierarchy, with consideration of details left until the end.

INITIAL IMPLEMENTATION OF THE SYSTEM

The first implementation of the skeletal plan theory of experiment design was completed by Friedland in 1979 [1]. The system functioned on a variety of analytical goals (strandedness determination, structural feature location, secondary structure determination, etc.) and was successfully tested on about 40 different experiment designs. The knowledge base, including skeletal plans, laboratory technique selection heuristics, nucleic acid descriptions, and laboratory condition descriptions, was built by expert molecular biologists entirely within the framework of the Unit System [4], [5], a general-purpose knowledge acquisition, representation, and manipulation package. The inference mechanism described above, skeletal plan selection followed by plan-step refinement, was embodied in a relatively short Interlisp program (on the order of 30 pages of code, including some facilities for the "genetic English" language for describing biological heuristics [6]. The system ran on the Digital Equipment Corporation PDP-10^{TM} and DecSystem 20^{TM} series of computers.

The system made a basic assumption of plan-step independence, that normally the steps in a skeletal plan could be considered independent entities with unfavorable interactions considered matters of detail to be fixed by creating subgoals (that in turn could be designed as small experiments). The system had some modest capabilities for explanation, in the form of specifying the rules used to make decisions ruling in or out techniques during a refinement step.

The first generation system achieved two purposes: demonstrating the validity of the computer science research that led to the skeletal plan refinement method, and showing the biological potential of an experiment design system. However, it clearly had many flaws as a practical system. Its knowledge base was limited to a relatively narrow range of analytical molecular biology. Interaction with the system was cumbersome; the user could only describe molecular structures and environmental parameters through the Unit System facilities. While the technique selection rules referred to the molecular structures and laboratory conditions relevant to the experiment, few facilities existed for simulating the actions of biological operators on those structures and conditions in order to accurately model the changing world state during an experiment. Finally, the system allowed only one basic strategy of skeletal plan refinement, that of depth-first technique refinement unless refinement was posponed because of explicit interactions with later, as yet undetermined, refinement choices. In other words, the first step of a skeletal plan was fully refined before work on the second started, and so on.

SECOND GENERATION IMPLEMENTATION--SPEX

An attempt to correct many of the experiment design system flaws discussed above, as well as to take advantage of new software and hardware methods, led to the second generation experiment design system, named the Skeletal Planner of EXperiments or SPEX [7]. First of all, SPEX utilizes a problem-solving framework, developed by Mark Stefik, known as *meta-planning* [8]. Meta-planning is an attempt to separate the types of decisions made during experiment design: strategic decisions on control strategies, domain-independent tactical decisions, such as whether to refine a particular skeletal plan step or postpone refinement while working on another step, and domain-dependent decisions, such as which exonuclease will best serve a given purpose. SPEX allows the user to select from among several control strategies; those currently implemented include depth-first, breadth-first, and heuristic (based upon a perceived importance of the kind of tactical operation chosen). Experiments are currently underway on a control structure based upon biological importance of the particular plan steps being refined.

Second, SPEX keeps an extensive record of all decisions made during the planning process. This allows planning to be restarted at any given point after modifying the knowledge base, and also provides the basis for experiment debugging tools that analyze experiment designs that failed during actual laboratory implementation.

Third, SPEX keeps a thorough, ongoing model of the molecular and environmental state of the world during experiment design. The effect of each biological operator is simulated and detailed representations, in the form of units, show the predicted state of the world before and after the application of each operation chosen.

Finally, in an attempt to cope with the interaction and size problems discussed above, SPEX operates on the Xerox 1100 AI workstation. The 1100 has a much larger address space than the DEC 2060 (approximately 4 million vs. 250000 words), allowing the knowledge bases to be much larger. It also has a fully-supported bit-map graphics display with a window and menu package and a "mouse" pointing device. This allows for the display of a much greater variety of information during experiment design; bit-map displays are treated like electronic desks with many overlapping display areas analogous to sheets of paper that can be instantly referenced.

SPEX is essentially domain-independent; a new design area may be studied by "plugging in" a different specific knowledge base. Work is continuing on improving the molecular biology specific portions of the cloning domain knowledge base as described in the next section. As the knowledge base improves, so does system performance and range of applicability.

THE KNOWLEDGE BASE

As was described above, large amounts of expert knowledge are essential to the experiment design process. The skeletal plan refinement method utilizes a relatively simple and straightforward *inference* method combined with an extensive domain-specific knowledge base. Construction of the knowledge base for the cloning experiment domain has taken by far the majority of research time spent building the entire system.

A molecular biology knowledge base used by SPEX contains the following categories of information:

1. Skeletal Plans--The abstracted plans along with the knowledge needed to determine when a given plan is applicable to a particular experimental goal.

2. Nucleic Acid Structures--The structural and functional information relevant to experiments and the procedural knowledge used to determine if information supplied by a user is both consistent and complete.

3. Laboratory Techniques--Relevant properties of the tools and techniques of molecular biology (enzymes, instrumentation, sequencing methods, etc.).

4. Technique Selection Heuristics--Expert knowledge on how to choose among alternative refinements of a skeletal plan step.

5. Simulation Knowledge--Information on how to simulate the effects of laboratory techniques on nucleic acid structures to model the course of an experiment.

A complete description of the cloning knowledge base is beyond the scope of this paper, but examples of these types of information in the cloning knowledge base will be illustrated below. A brief review of Unit System knowledge representation terminology may aid the reader in understanding the examples. Within a knowledge base, information is organized in the form of descriptions known as *frames* or *units*. Within a single unit, individual properties are stored in structures known as *slots*. Among other attributes, each slot has a name, a value or *restrictions* on possible values, and an indication of the syntactic form of the information represented in the slot, called a *datatype*. For example, the unit called **BAM** might have slots called **Restriction-site**, **Optimal-pH**, and **Literature-references**. The datatype of **Optimal-pH** might specify that it is an **Interval**.

Units are connected together through a *hierarchy* that represents a class, sub-class, sub-sub-class, and so on, relationship down to the level of individual entities. The previously shown example, beginning at **Laboratory-Techniques** and terminating at **EXO-III**, illustrates this relationship. Information within slots is *inherited* down the hierarchy. If a slot has been given a value in a parent unit, its children inherit that value exactly. If the slot has been given restrictions, then the restrictions may be narrowed or an actual value may be specified as long as the restrictions or value were included in the restrictions inherited from the parent. For example, one might specify that the **Optimal-Temperature** slot of the **Enzyme** unit had a restriction of from 0 to 100 degrees C. A subclass of enzymes, call them **Z-Enzymes**, might futher restrict this to 10 to 25 degrees C. Finally, a particular Z-enzyme, call it **ABC**, might have an actual value of 15 degrees C. (but not 8 or 30 degrees C.).

Note that the kinds of information represented can be procedural and heuristic as well as declarative and factual. Strategies (in the form of skeletal plans), selection heuristics, and simulation procedures are as much a part of the knowledge base as are more straightforward pieces of information like restriction enzyme cleavage sites and nucleic acid sequences.

A Skeletal Plan

Skeletal plans are described to the system by using an interactive editor that attempts to ensure that the plan is consistent, complete, and capable of being refined in the current knowledge base. It makes sure that the *utility* of the skeletal plan is understood to the knowledge base and that objects manipulated by the plan are clearly defined. The plan editor also functions as a general-purpose knowledge base editing tool, since it points out areas where selection heuristics are not well enough defined to allow reasonable refinement choices to be made or where skeletal plan operations have not yet been defined in the knowledge base.

An example of a skeletal plan (with annotations in italics) is shown below:

AMPLIFY-GENE

UTILITY: GENE-AMPLIFICATION
This skeletal plan is the model for gene amplification experiments. It is a straightforward cloning plan, but note that a vector selection step exists and a host selection step does not. The choice of host is driven by the choice of vector in the plan.

1. SELECT-VECTOR V
The user's request to a "select a vector" was translated into a shortened form and the name "V" was given for further reference to the chosen vector.

2. MODIFY-ENDS DNA DNA1
The user asked for end modification on the DNA fragment if necessary. The system called the starting fragment "DNA" and the resultant fragment "DNA1".

3. JOIN-TO-VECTOR DNA1 V V1
The fragment "DNA1" is joined to the vector "V" resulting in the recombinant structure "V1".

4. HOST-INSERTION V1 CELL
"V1" is now inserted into a nost, named "CELL" by the system.

5. CLONE-SELECTION CELL CLONE
Finally, "CELL" is searched for the products of interest called "CLONE".

The Nucleic Acid Model Unit

One of the major research problems during the course of the MOLGEN project has been how to adequately represent the structural and functional properties of nucleic acids. For the problem of cloning experiment design, a representation is needed both to provide a description of molecular properties that are used to help make skeletal plan step refinement decisions and to store the simulated changes in molecular properties that form the record of what should be happening when the experiment design is implemented in the laboratory. During a cloning experiment, the molecular model first must describe the target DNA, then both the target and the chosen vector as they are modified, and finally the recombinant molecule that results from inserting the target into the vector.

We have previously described and illustrated our basic mechanism for representing nucleic acid structures [5]. Each structure is represented by a single unit with slots for properties like length (an integer number of base pairs), nucleotide sequence (the sequence itself or a pointer to one of the sequence databases), restriction map (in a specially engineered map datatype), and so on. In addition, the unit also contains slots of the rules datatype which provide heuristics for filling in information when not all slots are explicitly provided by the user--a simple example would be instructions on how to determine length from nucleotide sequence--and for checking consistency of the information provided--for example, a single stranded structure cannot have nicks.

We have extended our previous work on representation mostly in the area of precise description of the ends of molecules; the exact nature of the terminii of target molecules and vectors very strongly influences many of the choices made during design of a cloning experiment. Part of the solution is to subdivide the model into left, central, and right segments, with the central segment describing the contiguous, usually double-stranded portion of the molecule, and the left and right segments describing the structural and chemical properties of the terminii. This solution has been adequate to represent such structures as looped ends, restriction enzyme fragments, and primed single-strands, but is not fully adequate for multiply nicked and/or gapped double-stranded molecules. A portion of one model structure is shown below:

```
DESCR:  This describes a primed RNA molecule. There is some restriction site information
        available which has been deduced from the genomic DNA.

LENGTH: 550

TYPE:   UPPER-STRAND-RNA

TOPOLOGY:       LINEAR
```

The following three sets of slots describe some properties of respectively the left end, double-stranded middle segment, and right end of the molecule. note that in this case, the left end nucleotide sequence was unknown

```
L-MAP:  Linear
        Length 400 base pairs

CODING region from 69 TO 300 indicated by ( and ) .
5'UNTRANSLATED region from 1 TO 68 indicated by [ and ] .
3'UNTRANSLATED region from 301 TO 400 indicated by < and >

                100      150                                    350
|[---------]----------------------------------------<----------------->
0               ↑        ↑                                      ↑         400
                ECOR1    BAMH1                                  SAL

L-5'-STRUCTURE: PROTRUDING

L-5'-CHEMICAL:  CAPPED

L-3'-CHEMICAL:  OH

M-MAP:  Linear
        Length 50 base pairs

PRIMED region from 1 TO 50 indicated by <PR and PR> .

|<PR----------------------------------------------------------------PR>
0                                                                      50

M-SEQUENCE:     Linear sequence 50 basepairs long
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

M-STRUCTURE:    DOUBLE-STRANDED


R-MAP:  Linear
        Length 100 base pairs

POLY-A region from 1 TO 100 indicated by < and > .

|<-------------------------------------------------------------------->
0                                                                      100

R-SEQUENCE:     Linear sequence 100 basepairs long

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAA

R-3'STRUCTURE:  PROTRUDING

R-3'CHEMICAL:   OH

R-5'CHEMICAL:   PO4
```

Cloning Vectors

We have chosen the hierarchy of cloning vectors as representative of the laboratory techniques portion of the knowledge base. In outline (with indentation representing parent-child relationship in the hierarchy) a part of the vectors hierarchy is:

```
VECTORS
    COSMIDS
        MUA-3
        PHC79
    PBR322-DOUG
    PHAGES
        LAMBDA-S
            C1-0
            CHARON-1
            CHARON-2
            CHARON-3
        M13BLA6
    PLASMIDS
        PA03
        PBR322
        PBR325
        PBR327
        PBR328
        PJJS000
        PJJS100
        PJJS200
        PJJS300
        PJJS350
        PJJS500
        PJJS550
        PJJS600
        PJJS700
        PJJS800
        PTR262
```

The current knowledge base includes examples of plasmids, bacteriophages, and plasmids; it still lacks any good descriptions of eucaryotic virus-based vectors. Each unit describing a vector contains structural properties of the vector (similar to the information for nucleic acid structures described above) as well as information relevant to decision-making for cloning experiment design. The latter category includes the minimum and maximum acceptable length of an insert, a list of selectable markers, a list of essential regions of the vector, and information relating to availability, stability, and cost of the vector. Some of the slots of a typical vector unit are shown below:

```
MUA-3
Generalization is COSMIDS

DESCR:  This unit describes the cosmid MUA-3 used to clone large segments of
eucaryotic DNA.  It contains all of pBR322 and approx. 400 bp of Lambda DNA
(cI857).  The Lambda cos ends were ligated into the Pst1 site of pBR322.
It can take about 48 kb of DNA probably in any sites which do not inactivate
the Tet gene. Minimum expected insert size is 38 kb.

REFERENCES:     Meyerowitz E.M, Guild G.M., Prestidge L.S., and Hogness D.S.,
GENE vol 11, (1980) pp. 271-282.

1-CUTTERS:      AVA1, BAL1, BAMH1, CLA1, HIND3, HINF3, PST1, PVU1, PVU2, RRU1, SAL1, . . .

DNA-RECOVERY:   2
```

An index from 1 to 4 to indicate how efficient is the recovery of recombinant DNA from the vector (with 1 being the most efficient)

```
ESSENTIAL-REGIONS:      COS, ORIGIN

HOST:   E.COLI

LENGTH: 4762
```

```
MAXIMUM-INSERTION-SIZE: 48 kb

MINIMUM-INSERTION-SIZE: 38 kb

REPLICON:       COLE1

SELECTABLE-MARKERS:     Tetracycline-Resistance, Ampicillin-Resistance

SELECTION-MODE: Colony, Insert

STABILITY:      2


UNIQUE-RESTRICTION-SITES:

Circular
Length 4762 base pairs

TETRACYCLINE-RESISTANCE region from 23 TO 1400 indicated by <TET-R and TET-R>
AMPICILLIN-RESISTANCE region from 1 TO 3297 indicated by <AMP-R and AMP-R>
ORIGIN region from 2200 TO 2400 indicated by <ORI and ORI>
COS-L-ARM region from 3835 TO 4015 indicated by <L and L>
COS-R-ARM region from 3616 TO 3834 indicated by <R and R>

    376    821
 30     661         1446        2222                          4248
 25    566   939    1425     2068                   3613    4138
|<AMP-R-------TET-R>-----------<ORI>-------AMP-R>---<R-R>L>-----------|
0↑  ↑  ↑↑  ↑ ↑      ↑        ↑  ↑                   ↑       ↑ ↑        4762
 CLA1  SPH1  XMA3   AVA1     PVU2                   PST1    PVU1
 HIND3  SAL1        BAL1        TTH1111                       RRU1
    BAMH1  HINF3
```

Technique Selection Heuristics

Two different methods for describing technique selection heuristics are provided in the knowledge base: top-down and bottom-up. Normally, skeletal plan step refinement begins at some point in the hierarchy and moves downward from classes of techniques through more detailed sub-classes to finally reach actual instances of techniques. Top-down selection heuristics are stored with a class of techniques, and describe how to choose among the children of that technique. Bottom-up selection heuristics are stored with the children of a class of techniques and describe for what purposes the children are particularly useful. In some sense, top-down heuristics are equivalent to "compiled" knowledge that was originally in a bottom-up form. The choice of which method (or both) to employ is left to the expert biologist and seems to be mostly one of personal preference. It is too early to make any judgment about efficiency or accuracy of the two methods.

The selection heuristics of either form can be described in two languages: the "genetic-English" rules language or Interlisp (or on the DEC 2060 version of the system any available computer language). The rules language provides an easy-to-learn way of describing simple heuristics like:

IF THE GOAL OF DIGESTION IS SPECIFIC THEN AVOID MECHANICAL METHODS

and seems to suffice for most of the selection heuristics. When the rules for selection get more complicated and algorithmic, the experts have found it useful to learn enough Interlisp to write a formal selection procedure. The best example of this is the vector selection process where the matching of end-types on both the target and the vector along with available restriction endonuclease sites can get complex. We have written an Interlisp function which takes into account insertion size compatibility, selectable markers, and other properties to make an optimal choice. Designing this function involved a substantial research undertaking in itself which will be described in a later report.

Simulation of Structural Modification Operations

Simulating the results of applying laboratory operations involves modifying the slots of the units modeling the ongoing world state of the experiment. In some cases, simulation is straightforward; for example, total digestion with a restriction enzyme can be simulated by breaking double-stranded DNA molecules at the proper cleavage sites. In other cases, simulation can be very complex; for example, determining the percentage of single circular inserts of target into vector involves knowledge about ends, concentrations, type of ligase used, etc. We have previously described [5] how the rules language provides "verbs" which model the results of common actions on nucleic acid sequences and restriction maps. This has been extended to providing procedures for modifying all of the slots of the nucleic acid structure units in order to simulate ligation, circularization, complementation, restriction digestion, reverse-transcription, kinase and phosphatase activity, and the actions of a few specific enzymes, like Bal31. The simulation functions, mostly written in Interlisp, but transparent to biologist users, also check substrate compatibility before acting. The user only need write:

LIGATE FRAGMENT TO VECTOR

to apply the simulation procedure "LIGATE."

AN EXAMPLE OF AUTOMATED EXPERIMENT DESIGN

The following annotated example will illustrate the SPEX design system at work on a typical cloning experiment. The experiment we have chosen is one where the user wants to generate a library of clones containing fragments which were generated with an XBA1/SAL1 double digest. The average length of the fragments is 10 kb. Currently, for experiment simulation purposes, we only pick a single representative fragment in such a situation. Since the refinement strategy chosen in the example is breadth-first, decision-making tends to be fairly well spread out among the steps in the plan. As was previously discussed, we are now attempting to develop heuristics for optimal stategy selection by analyzing the empirical results of many different computer-produced experiment designs.

Note that the only user interaction in the example is in choosing a strategy, an experiment goa' and a starting DNA fragment. He had previously described the starting DNA fragment, although he could have requested guidance in producing such a description within the Unit System. The user also could have requested much ..iore verbose output describing every operation in the experiment design process.

```
Currently available goals are :
1. CLONE-CDNA-FULL-LENGTH
2. CLONE-LARGE-FRAGMENTS
3. FAVOR-EXPRESSION
4. FAVOR-SECRETION
5. GENE-AMPLIFICATION
6. MAKE-LIBRARY
7. SEQUENCING
8. CLONE-CDNA
Please choose a goal by typing in the number:  6
```

The system provides a list of the currently understood goals for experiment design. These are automatically updated as the skeletal plan portion of the knowledge base grows. The user selected MAKE-LIBRARY. The system then searched for those plans which are useful for the selected goal and found LIBRARY-CONSTRUCTION.

```
Expanding the skeletal plan :  LIBRARY-CONSTRUCTION

1.  SELECT VECTOR
2.  MODIFY-ENDS
3.  JOIN-TO-VECTOR
4.  SELECT HOST
5.  HOST-INSERTION
6.  CLONE-SELECTION
```

Here, the system asks if the initial description of the molecule to be cloned and the associated laboratory environment exists.

Refinement of the skeletal plan now proceeds. First the system collects the next level choices in the knowledge base for each skeletal plan step, pruning the choices if top-down selection rules are applicable. Then it applies bottom-up selection rules, environmental rules, and general selection heuristics to make a single choice at the next level in the hierarchy.

```
Refining the technique SELECT VECTOR
Top-down selection rules apply
Legal Choices are:  CHARON-1, CHARON-2, CHARON-3, MUA-3, PBR322
```

Vector selection proceeds in a top-down manner, with the VECTOR unit pointing to the procedure previously described in this paper. Charon-1 through charon-3 are the only vectors with both sites found at the end of the fragment. MUA-3 and pBR322 are listed because they contain one of the sites (Sal1) which could be used if no other vectors were available, albeit with further technical complications.

```
Refining the technique MODIFY-ENDS
Possible refinements are: ADD-LINKERS GENERATE-BLUNT-ENDS NO-MODIFICATION TERMINAL-TRANSFERASE

Refining the technique JOIN-TO-VECTOR
Possible refinements are : T4-DNA-LIGASE T4-RNA-LIGASE

Refining the technique HOSTS
Possible refinements are : BACTERIA MAMMALIAN-CELLS

Refining the technique HOST-INSERTION
Possible refinements are : INFECTION TRANSFORMATION

Refining the technique CLONE-SELECTION
Possible refinements are : ANTIBODY-SELECTION HYBRIDIZATION-SELECTION
```

The first top-down phase of the breadth-first refinement is complete: the system has located the first-level choices for each plan step. Some of the choices are actual laboratory methods, some are narrower sub-classes of methods.

Now the system applies bottom-up selection criteria in order to further discriminate among choices.

```
Deciding the status of CHARON-1
The status is NO-RULE
Deciding the status of CHARON-2
```

```
The status is NO-RULE
Deciding the status of CHARON-3
The status is NO-RULE
Deciding the status of MUA-3
The status is NO-RULE
Deciding the status of PBR322
The status is NO-RULE
```

NO-RULE means that no bottom-up rules that matched the current world model were found for any of the vectors remaining from the earlier top-down process. It does not mean that no rules were found, only that none were applicable to the current experimental context. The bottom-up rules can either rule techniques in or out (CHOOSE or AVOID).

```
CHARON-1 selected by general selection heuristics
```

CHARON-1 was chosen among the remaining vectors by the type of convenience and reliability heuristics discussed earlier.

```
Interpreting consequences rules for:
CHARON-1, MODIFY-ENDS, HOSTS, JOIN-TO-VECTOR, HOST-INSERTION, CLONE-SELECTION
```

Whenever a choice is made, the simulation rules, called "consequences" are applied for that choice as well as all following choices. This makes sure that all succeeding decisions are made from the most accurate world model.

Now the MODIFY-ENDS step is further refined. In this case several bottom-up selection rules do apply.

```
Deciding the status of ADD-LINKERS
The following rules apply:
IF LEFT-CUTTER IS NOT LEFT-RECIPIENT
OR RIGHT-CUTTER IS NOT RIGHT-RECIPIENT THEN CHOOSE LINKERS
The status is RULED-OUT

Deciding the status of GENERATE-BLUNT-ENDS
The status is NO-RULE

Deciding the status of NO-MODIFICATION
The following rules apply:
IF  LEFT-CUTTER IS LEFT-RECIPIENT
AND RIGHT-CUTTER IS RIGHT-RECIPIENT THEN
          CHOOSE TECHNIQUE ELSE AVOID TECHNIQUE
The status is RULED-IN

Deciding the status of TERMINAL-TRANSFERASE
The status is NO-RULE
NO-MODIFICATION is chosen
```

Since NO-MODIFICATION was the only technique with a positive bottom-up selection rule, it is chosen.

```
Interpreting consequences rules for:
NO-MODIFICATION, HOSTS, JOIN-TO-VECTOR, HOST-INSERTION, CLONE-SELECTION
```

Again, a decision has been made so simulation rules are applied.

Now refinement proceeds on the JOIN-TO-VECTOR step.

```
Deciding the status of T4-DNA-LIGASE
The status is NO-RULE
Deciding the status of T4-RNA-LIGASE
The status is NO-RULE

T4-RNA-LIGASE ruled-out by environmental rules
```

Both ligases pass the goal-directed bottom-up selection rules, but T4-RNA-LIGASE is ruled-out by the environmental rules (in this case the wrong substrate, DNA instead of RNA) that were discussed earlier.

```
T4-DNA-LIGASE is chosen

Interpreting consequences rules for:
T4-DNA-LIGASE, HOSTS, HOST-INSERTION, CLONE-SELECTION
```

Now refinement proceeds on the host selection problem .

```
Deciding the status of BACTERIA
The status is NO-RULE

Deciding the status of MAMMALIAN-CELLS
The status is NO-RULE

BACTERIA selected by general selection heuristics
```

Normally, the earlier choice of CHARON-1 would drive the selection of E. Coli as a host. This skeletal plan forces independent consideration of hosts; we wanted to determine if E. Coli would still be chosen.

```
Interpreting consequences rules for:
BACTERIA, HOST-INSERTION, CLONE-SELECTION
```

Now refinement proceeds for HOST-INSERTION .

```
Deciding the status of INFECTION
The following rules apply:
IF VECTOR-TYPE IS VIRUS OR VECTOR-TYPE IS PHAGE THEN CHOOSE INFECTION
The status is RULED-IN

Deciding the status of TRANSFORMATION
The status is NO-RULE
INFECTION is chosen

Interpreting consequences rules for:
INFECTION, CLONE-SELECTION
```

Now refinement of CLONE-SELECTION occurs .

```
Deciding the status of ANTIBODY-SELECTION
The status is NO-RULE

Deciding the status of HYBRIDIZATION-SELECTION
The following rules apply:
IF SELECTION-PROBE INCLUDES DNA THEN CHOOSE HYBRIDIZATION-SELECTION
The status is RULED-IN
HYBRIDIZATION-SELECTION is chosen

Interpreting consequences rules for:
HYBRIDIZATION-SELECTION
```

The system has finished its first bottom-up refinement pass. Note that some of the techniques chosen are terminal, that is, they are actual laboratory methods, and some need further refinement. Now the second top-down refinement pass begins; we have deleted the system comments that indicate CHARON-1, NO-MODIFICATION, and INFECTION are terminal.

```
Refining the technique T4-DNA-LIGASE
Possible refinements are : T4-BLUNT-D-LIGASE T4-COHESIVE-D-LIGASE

Refining the technique BACTERIA
Possible refinements are : COLIFORMS

Refining the technique HYBRIDIZATION-SELECTION
Possible refinements are : COLONY-SELECTION PLAQUE-SELECTION
```

The top-down refinement pass is complete and the next bottom-up process starts for those techniques which are not terminal .

```
Deciding the status of T4-BLUNT-D-LIGASE
The status is NO-RULE

Deciding the status of T4-COHESIVE-D-LIGASE
The following rules apply:
IF L-5'STRUCTURE IS STICKY OR R-5'STRUCTURE IS STICKY
OR L-5'STRUCTURE IS PROTRUDING OR R-5'STRUCTURE IS PROTRUDING
   THEN CHOOSE STICKY-END-LIGATION
   ELSE AVOID TECHNIQUE
The status is RULED-IN
```

```
T4-COHESIVE-D-LIGASE is chosen

Interpreting consequences rules for:
T4-COHESIVE-D-LIGASE, BACTERIA, INFECTION, HYBRIDIZATION-SELECTION

Deciding the status of COLIFORMS
The status is NO-RULE
COLIFORMS is chosen
```

COLIFORMS was chosen because it was the only child of BACTERIA in the current knowledge base. If it had been explicitly ruled-out by goal-directed or environmental rules, then the system would have backtracked up a level and made an alternate choice.

```
Interpreting consequences rules for:
COLIFORMS, INFECTION, HYBRIDIZATION-SELECTION

Deciding the status of COLONY-SELECTION
The status is NO-RULE

Deciding the status of PLAQUE-SELECTION
The following rules apply:
IF SELECTION-MODE INCLUDES PLAQUE THEN CHOOSE TECHNIQUE
                                  ELSE AVOID TECHNIQUE
The status is RULED-IN
PLAQUE-SELECTION is chosen

Interpreting consequences rules for:
PLAQUE-SELECTION
```

The bottom-up refinement pass is complete. Only COLIFORM and PLAQUE-SELECTION are not terminal laboratory techniques.

Another top-down pass begins; T4-COHESIVE-D-LIGASE was terminal.

```
Refining the technique COLIFORMS
Possible refinements are : E.COLI

Refining the technique PLAQUE-SELECTION
Possible refinements are : PLAQUE-LIFT

Deciding the status of E.COLI
The status is NO-RULE
E.COLI is chosen
```

Again only one choice remained, as our knowledge base is still sparse in knowledge about bacterial hosts for cloning.

```
Interpreting consequences rules for:
COLIFORMS, INFECTION, PLAQUE-SELECTION

Deciding the status of PLAQUE-LIFT
The following rules apply:
IF SELECTION-MODE INCLUDES PLAQUE THEN CHOOSE TECHNIQUE
                                  ELSE AVOID TECHNIQUE
The status is RULED-IN
PLAQUE-LIFT is chosen

Interpreting consequences rules for:
PLAQUE-LIFT
```

The bottom-up refinement pass is complete and all steps are terminal.

SPEX now shows the user the original skeletal plan and the chosen techniques that are final refinement of each step. If he wished, the user could now examine the complete record of the design process, stored as units in the knowledge base, or the various units representing the ongoing world simulation model. He could modify any of the selection or simulation rules if he disagreed with a SPEX decision and restart the design process.

```
The instantiated plan is:
```

```
goal : MAKE-LIBRARY

1. SELECT VECTOR V
       using CHARON-1
2. MODIFY-ENDS DNA DNA1
       using NO-MODIFICATION
3. JOIN-TO-VECTOR DNA1 V V1
       using T4-COHESIVE-D-LIGASE
4. SELECT HOST CELL
       using E.COLI
5. HOST-INSERTION V1 CELL
       using INFECTION
6. CLONE-SELECTION CELL CLONE
       using PLAQUE-LIFT
```

RESULTS

Currently, the system is operational on the following classes of cloning experiments: library construction, gene amplification, and cDNA cloning. Its performance is bound by the extent of the knowledge base, which currently comprises about 300 units covering basic laboratory techniques, enzymes, and vectors. The knowledge base is weak in several major areas, particularly expression vectors and detailed knowledge about alternative hosts for cloning. The knowledge base is approaching adequacy on the procedural knowledge for technique simulation during the modeling of recombinant DNA experiments.

Much of our effort has centered on improving the vector selection process that seems to drive most cloning experiments. We have implemented a comprehensive set of selection heuristics for vectors which take into account experimental goals and the utility of each particular vector. For example, if the goal of an experiment is library construction, one wants to use a vector which is easy to handle in large quantities and easy to screen. We need to extend this comprehensive approach to plan instantiation to those cases where other factors, say the choice of host for expression, drives the experiment design process more than vector selection.

The system suffers from several other limitations. First, it runs best on a machine, the Xerox 1100, which is still limited in availability. This means we have not yet had experience with the experiment design system in a routine cloning laboratory environment. Such experience will undoubtedly lead to many minor changes and probably several major ones, especially in the design system interface to biologist users. This has certainly been true for all previous MOLGEN systems.

Second, the system runs too slowly for routine use; it normally takes close to an hour for SPEX to design an experiment in detail. The growth of the knowledge base exacerbates this problem, but rapid improvements in hardware and software speed should eliminate the speed problem shortly.

Third, the heuristics for compromise selections are very weak. The system allows a single important choice, usually vector selection, to drive other decisions. Often the best total design decision is a compromise; for example, the best vector may require a very expensive or hard-to-obtain linker, whereas a slightly inferior vector may be much more flexible on the choice of linkers. The modular strategy space of SPEX allows for experimentation with such compromise strategies, but the research is only beginning.

Finally, the system lacks the second-order or *meta* heuristics needed to combine methods when a single method fails. For example, the system knows about both sticky-ended and blunt-ended ligation of a target fragment to a vector. It also knows that sticky-ended ligation is generally the preferred method. However, unless a method which specifically provided for sticky-ended ligation on one end and blunt-ended ligation on the other was explicitly part of the knowledge base, the system would currently not find it for a fragment which only had one "good" end. Other work in artificial intelligence has made progress in building systems which "discover" new methods by plausible combination of existing methods (see [9]), and we hope to extend the experiment design system in that direction.

CONCLUSIONS

The MOLGEN experiment design work has been extremely fruitful in developing general methods for the acquisition, representation, and manipulation of complex, domain-specific knowledge. In addition, a domain-general system for design has been constructed and tested. The practical utility of the system for experiment design in molecular biology is still to be proven. The knowledge base must be made substantially more extensive and complete, and the user interface must be developed to the point where the system is almost entirely self-teaching.

The system has demonstrated potential for several problems related to experiment design. The associated knowledge bases have proven to be useful as *intelligent encyclopedias*, teaching aids, and experiment simulation aids [5]. In addition, the MOLGEN group has recently completed work on a parallel system which aids in the *debugging* of unsuccessful experiments. This system analyzes experiments in order to determine whether failure was the result of technical errors in implementation (too much salt added to a reaction mix), knowledge base selection errors (the wrong enzyme was chosen for single-strand specific digestion), or overall plan errors (the skeletal plan steps did not fit together as a coherent whole).

ACKNOWLEDGEMENTS

We gratefully acknowledge the contributions of our many collaborators in this research. Profs. Douglas Brutlag, Laurence Kedes, and John Sninsky have provided continual biological expertise and critical insight throughout the course of the MOLGEN project. Dr. Andrew Buchman has assisted in the construction of the cloning design knowledge base. Profs. Bruce Buchanan and Edward Feigenbaum have contributed valuable advice on the cognitive science and artificial intelligence aspects of the work. Mr. Armin Haken built the current editing system for the acquisition of skeletal plans that was described in this paper.

The MOLGEN project is part of the Heuristic Programming Project in the Stanford Computer Science Department. The research has been supported under NSF grant MCS80-16247. Computation resources have been provided by the SUMEX-AIM National Biomedical Research Resource, NIH grant RR-00785-08.

REFERENCES

1. Friedland, P., "Knowledge-Based Experiment Design in Molecular Genetics," Computer Science Department Report CS-79-771, Stanford, October 1979.
2. Friedland, P. and Iwasaki, Y., "The Concept and Implementation of Skeletal Plans", To Appear in *Artificial Intelligence*
3. Polya, G., *How to Solve It*, Doubleday Anchor Books, New York, 1957.
4. Smith, R. G. and Friedland, P., "A User's Guide to the Unit System," Heuristic Programming Project Memo HPP-80-28, Stanford, December 1980.
5. Friedland, P., Kedes, L., Iwasaki, Y., and Bach, R., "GENESIS, a Knowledge-Based Genetic Engineering Simulation System for Representation of Genetic Data and Experiment Planning," *Nucleic Acids Research*, Vol. 10, No. 1, 1982, pp. 323-340.
6. Friedland, P., "Acquisition of Procedural Knowledge from Domain Experts," *Proceeding of the Seventh International Joint Conference on Artificial Intelligence*, IJCAI, 1981, pp. 856-861.
7. Iwasaki, Y., and Friedland, P., "SPEX: A Second-Generation Experiment Design System," *Proceedings of AAAI-1982*, AAAI, 1982, pp. 341-344.
8. Stefik, M.J., "Planning and Meta-Planning," HPP-memo HPP-80-13, Stanford University Heuristic Programming Project, 1980.
9. Lenat, D. B., "EURISKO: A Program That Learns New Heuristics and Domain Concepts," *Artificial Intelligence*, Vol. 21, 1983, pp. 61-98.

Table of Contents