Proceedings of the International Joint Conference on Artificial Intelligence, 1985.

EMPIRICAL AND MODEL-BASED REASONING IN EXPERT SYSTEMS

P. A. Koton

Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

ABSTRACT

Many expert systems are now being written which rely on highly-compiled, empirical knowledge for their reasoning power. Model-based reasoning has significant theoretical advantages. I constructed two expert systems for the same domain, one using large-grained compiled knowledge, the second using model-based reasoning. The use of model-based reasoning resulted in improved knowledge accessibility and flexibility, and expanded problem-solving ability.

Many current expert systems rely on highly-compiled, large-grained knowledge (heuristics, empirical associations, "rules of thumb") for their reasoning power.* An alternative approach, model-based reasoning, uses a detailed model of the objects in the domain and the operations that act on those objects. The relative problem solving abilities of these two different types of reasoning have not previously been experimentally compared. I constructed two programs that solve problems in the same domain, with the same objectives, but each using a different reasoning method. GENEX (Koton, 1983) and GENEX II (Koton, 1985) solve problems about the behavior of bacterial operons,** a subfield of molecular biology. Both programs take as input a description of an operon (real or imaginary) and information on whether or not the genes of that operon are expressed, then attempt to deduce the biological control mechanism causing the observed behavior of the operon. GENEX uses large-grained empirical knowledge to solve problems, and GENEX II uses model-based reasoning. GENEX II can solve a greater variety of problems and more difficult problems than GENEX.

The expert knowledge in the GENEX program consists of 62 pieces of knowledge relating observed phenomena to possible causes. Each piece of information is highly compiled and, in general, incorporates several inference steps. For example:

```
IF the mutation is in the promoter
AND the gene is not expressed
THEN possibly RNA polymerase
     cannot bind to the promoter.
```

GENEX performs well on many problems. However, the types of problems which it can be used to solve is limited. It can only be *guaranteed* to yield a correct solution for simple problems involving a prototypical operon containing at most one mutation. For some more difficult problems, involving more than one mutation, GENEX gives the correct answer, but for others, it gives an incorrect solution. This limitation may be attributed to the program's reliance on reasoning via empirical associations.

In order to overcome the limitations of GENEX, GENEX II was implemented. This program contains a description of each object and mechanism involved in operon control. For instance,

```
the BINDING-SITE of RNA-POLYMERASE
is the PROMOTER.

a MOLECULE(M) BINDS to a SITE(S) if
S is the BINDING-SITE for M and
M and S have COMPLEMENTARY-CONFORMATIONS and
nothing else is BOUND to S and
M is FREE to BIND.
```

This research was supported in part by the National Institutes of Health Grant No. 1 P01 LM 03374-04 from the National Library of Medicine, and in part by National Institutes of Health Grant No. 1 P41 RR 01096-05 from the Division of Research Resources.

*These concepts are not strictly logically equivalent, but in current practice systems based on empirical or heuristic associations typically do express their knowledge in large-grained, compiled form.

**The term *operon* refers to a genetic control system in which the activity of a set of *structural genes*, coding for metabolically related proteins required by the cell, is governed by the interaction of a regulatory protein with an adjacent region of DNA (called the operator). In order for the operon proteins to be made, the enzyme RNA polymerase must be able to bind to the DNA at a specific site, known as the *promoter*. It can only do this if the operator, which overlaps the promoter, is not blocked by the presence of the bound regulatory protein. Whether or not the regulatory protein binds at the operator depends on whether the cell needs to make more of the operon proteins.

```
the EXPRESSION of OPERON(O) INITIATES if
PART-OF(O) is P and
P is a PROMOTER and
RNA-POLYMERASE BINDS to P.

an OPERON(O) is EXPRESSED if
O is INITIATED and
O is not ATTENUATED and
O is TRANSCRIBED and
O is TRANSLATED.
```

This knowledge base, written in Prolog, contains just over 250 clauses. The descriptions include the different parts of the operon, the role that each part plays in the functioning of the operon, how the parts interact, and how lower-level mechanisms combine into higher-level behavior. The knowledge base for GENEX II contains essentially the same information that GENEX has, but the information that is "compiled into" GENEX is stated explicitly in GENEX II. These descriptions enable the program to reach its solutions by constructing a model of the operon and simulating the processes that act on it rather than doing a match between the observed phenomena and the knowledge in the system, as GENEX does.

GENEX II can deal with two large classes of problems which GENEX was unable to solve: the combined effect of multiple mutations on gene expression, and problems involving non-prototypical operons. An example of the type of problem that GENEX II can solve and on which GENEX fails is shown below.*

Bacterium pedantia produces a protein, Sporulin, that makes other bacteria sporulate. Sporulin synthesis is inducible by chalk. In the absence of chalk a repressor produced by gene *spoR* prevents expression of the Sporulin structural gene spoS.

A variation carrying two mutations can be made in *B. pedantia* by recombination. Most double mutants that carry both a constitutive *spoR* and a constitutive *spoO* mutation [mutations in the regulatory gene and the operator of the *spo* gene that cause its product to be made continually] are constitutive, as expected. However, one particular double carrying mutations *spoR*43 and *spoO*97 is anomalous: it produces Sporulin *only* when chalk is *absent*. *spoR*43 is known to be missense [a missense mutation is one that changes the structure of the resulting protein]. Explain briefly how the repressor and operator are functioning in the double mutant.

GENEX II solves this problem by first noting that the mutations have introduced uncertainty about the three-dimensional shape of objects in the domain. Complementarity of shape partially determines which objects can bind to which others. The program generates all possible pairs of shapes (in the abstract sense; i.e. to which other objects is this object's shape complementary) which are consistent with the mutations given and the observed behavior of the operon. One possibility it generates is that the regulatory protein has been mutated in such a way that only when it is bound to chalk can it bind to the operator. GENEX II simulates the behavior of the operon under this set of circumstances. It reasons along the following lines:**

In order for the operon to be expressed, RNA polymerase must be able to bind to the promoter. It cannot bind to the promoter if something is already bound there. Because the operator overlaps the promoter, anything bound to the region where the operator and promoter overlap is bound to the promoter. The system then looks to see if anything can be bound to the operator. The hypothesis that only when chalk is bound to the mutated regulatory protein it is in the proper conformation to bind to the mutated operator, results in the system reaching the conclusion that in the presence of chalk, RNA polymerase cannot bind and the gene product is not made. This is consistent with the observed behavior of the operon, and will be reported as a possible solution.

When the same problem is given to GENEX, it recognizes that the *spo* gene is being repressed by chalk, and thus when chalk is present the gene product is not made. GENEX can not discover the mechanism by which this occurs, because this particular combination of mutations does not correctly match any of its heuristics. It partially matches two heuristics, one stating that the mutated operator cannot bind a normal regulatory protein, and the other stating that the mutated regulatory protein cannot bind to a normal operator, and can go no further.

Certainly, any given problem which is currently unsolvable by GENEX, or other systems which use compiled knowledge, could be solved by giving the program a new piece (or pieces) of information. However, this would improve the performance of the system only for that one type of problem. Compiled knowledge captures domain knowledge as is; it does not capture the underlying model. Because such programs solve problems by matching the current situation against a set of predetermined situations, the knowledge base *anticipates* situations that may arise. For small domains, it may be possible to enumerate every possible situation and encode a response for each one.*** In a large domain, it may be extremely difficult to

*This problem is from an MIT undergraduate genetics exam. It has been simplified for this paper. The full text of the problem can be found in (Koton, 1985).

**The program's complete solution is 13 pages long. Interested readers are referred to (Koton, 1985).

***This technique was used successfully in PUFF (Kunz, 1978).

predict every possible state of the system. In contrast, a system which uses model-based reasoning, such as GENEX II, can *generate* new situations based on its underlying model of the domain.

Empirical "rules of thumb" often contain implicit preconditions. For example, one such rule in GENEX states that "deleting the operator results in increased expression of the operon." The implicit precondition here (as it clearly must be in similar heuristics for other programs) is "all other things being equal." If all other things are *not* equal, the validity of the conclusion is uncertain. Implicit preconditions can make it difficult, if not impossible, to determine the conditions for which a "rule of thumb" can be applied correctly. However, explicitly stating every precondition may be tedious — consider having to explicitly encode "all other things being equal" in a system with hundreds of components.

Another limitation of compiled knowledge is that the domain knowledge is contained implicitly, and thus is inaccessible. For instance, the GENEX rule shown above encodes the rule of thumb that a mutation in the promoter and a decrease in transcription of the operon suggests that RNA polymerase cannot bind to the promoter. If given a problem that included a promoter mutation and a decrease in gene product, GENEX applies this rule and suggests that the promoter mutation caused the decrease in gene product. However, GENEX has no knowledge of the mechanism which makes this true, i.e. that the promoter is where the RNA polymerase binds, if the promoter is deleted the polymerase cannot bind, and if the polymerase cannot bind the operon cannot be transcribed. This information is explicitly encoded in GENEX II, as is illustrated above. The advantages of such increased accessibility have been extensively discussed in (Swartout, 1981).

Knowledge encoded implicitly is also inflexible. One piece of knowledge may be contained in any number of places in the system, so in order to change that knowledge it must be changed every place it is used.

When model-based reasoning is used, the program creates its solution anew for each problem, based on its model of the domain, rather than relying on pre-compiled solutions. Theoretically, the program should be able to solve any problem that is derivable from the model of the domain.

As always, there are tradeoffs. Model-based systems such as GENEX II require a more complicated control structure to reduce the amount of search for possible solutions. The finely-grained knowledge in GENEX II creates inference chains that are about four times longer than those generated by GENEX in the solution of the same problem. This results in a slower-executing program. Perhaps the ideal system would use both empirical associations for speed and model-based reasoning for improved problem-solving ability and better understanding of its reasoning, as described by (Barnett, 1982).

Acknowledgments

I would like to thank Peter Szolovits, who supervised the research on which this paper is based, and Ramesh Patil, who introduced me to many of the issues raised in this paper.

References

1. Barnett, J. "Some Issues of Control in Expert Systems." In *Proc. of the International Conference on Cybernetics and Society.* Seattle, Washington, October, 1982, pp. 1–5.

2. Koton, P. "A System to Aid in the Solution of Problems in Molecular Genetics." SM Thesis, MIT Laboratory for Computer Science, May 1983.

3. Koton, P. "Towards a Problem Solving System for Molecular Genetics," Technical Report 338, MIT Laboratory for Computer Science, Cambridge, Massachusetts, 1985.

4. Kunz, J. "A Physiological Rule-based System for Interpreting Pulmonary Function Test Results," Heuristic Programming Project Report HPP-78-19, Stanford University Computer Science Dept., Stanford, California, 1978.

5. Swartout, William R. "Producing Explanations and Justifications of Expert Consulting Programs," Technical Report 251, MIT Laboratory for Computer Science, Cambridge, Massachusetts, 1981.