Machine Learning, 12, 89–116 (1993)

Design Methods for Scientific Hypothesis Formation and Their Application to Molecular Biology

PETER D. KARP PKARP@AI.SRI.COM
Artificial Intelligence Center, EJ229, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025

Abstract. *Hypothesis-formation* problems occur when the outcome of an experiment as predicted by a scientific theory does not match the outcome observed by a scientist. The problem is to modify the theory, and/or the scientist's conception of the intial conditions of the experiment, such that the prediction agrees with the observation. I treat hypothesis formation as a *design problem*. A program called HYPGENE designs hypotheses by reasoning backward from its goal of eliminating the difference between prediction and observation. This prediction error is eliminated by *design operators* that are applied by a planning system. The synthetic, goal-directed application of these operators should prove more efficient than past generate-and-test approaches to hypothesis generation. HYPGENE uses heuristic search to guide a generator that is focused on the errors in a prediction. The advantages of the design approach to hypothesis formation over the generate-and-test approach are analogous to the advantages of dependency-directed backtracking over chronological backtracking. These hypothesis-formation methods were developed in the context of a historical study of a scientific research program in molecular biology. This article describes in detail the results of applying the HYPGENE program to several hypothesis-formation problems identified in this historical study. HYPGENE found most of the same solutions as did the biologists, which demonstrates that it is capable of solving complex, real-world hypothesis-formation problems.

Keywords. Hypothesis formation, discovery, scientific reasoning, computational biology.

1. Introduction

This article reports on a computational investigation of scientific *hypothesis formation*. When the outcome of an experiment predicted by a scientific theory does not match the outcome observed by a scientist, the scientist formulates one or more hypotheses to eliminate the discrepancy between prediction and observation. I studied hypothesis formation in the context of a 15-year program of research in molecular biology that culminated in the discovery of a new mechanism of gene regulation in bacteria.

This article presents methods for solving hypothesis-formation problems that have been implemented in a computer program called HYPGENE (Hypothesis Generator). These techniques were developed during the course of my dissertation work, which included a historical study (summarized in section 2) of Dr. Charles Yanofsky's discovery of the gene-regulation mechanism called *attentuation* (Yanofsky, 1989). The purpose of this historical study was to precisely identify real-world hypothesis-formation problems, and to suggest ways in which the biologists might have solved these problems; however, this article does not make any cognitive-modeling claims. Section 6 describes two trials of HYPGENE on problems from the history of attentuation.

My dissertation research yielded a framework for representing theories of molecular biology, and methods for using those theories to predict experimental outcomes. These prediction methods and representations are embodied in a program called GENSIM (Genetics

Simulator), which is described in detail elsewhere (Karp, 1989, 1991), and is summarized in section 4. HYPGENE generates hypotheses that remove errors in GENSIM predictions, that is, GENSIM is the performance program that HYPGENE improves.

In this article, science is viewed as a goal-oriented activity, the objective of which is to improve the predictive performance of scientific theories. In pursuit of this objective, scientists solve a number of reasoning problems, such as selecting research goals, designing experiments,[1] interpreting raw data, predicting experimental outcomes based on a theory, and formulating hypotheses that revise a theory. This article is concerned with the hypothesis-formation reasoning that occurs when the observed and the predicted outcomes of an experiment do not match within established tolerances. Under these circumstances, a scientist might take a number of actions, such as altering his theory, or his beliefs about the laboratory experiment, or both, to eliminate the discrepancy between the two. The scientist might alter his beliefs about the experiment by deciding that impurities were present in the experiment, or that his laboratory instruments were miscalibrated, or that his knowledge of the initial experimental conditions was incomplete. If he trusted his experimental results, however, he would alter his theory. This article is concerned with hypotheses that revise a scientist's beliefs about the initial conditions of an experiment.

Figure 1 shows the relationship between the GENSIM and HYPGENE programs. The task of GENSIM is to predict the outcome of an experiment from the history of attentuation using the theory of gene regulation that was in use at the time that the experiment was performed. A human compares GENSIM's prediction to the actual outcome of the experiment as reported in the scientific literature. If a human detects that the prediction conflicts with the observation, he or she calls on HYPGENE to improve GENSIM's performance—to improve the

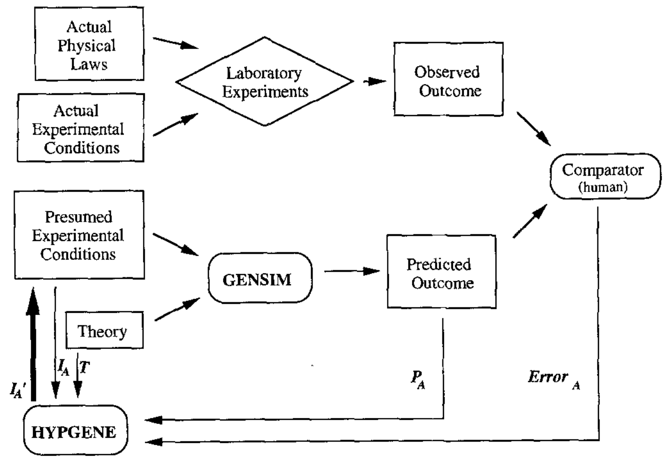

Figure 1. The relationship between the GENSIM and HYPGENE programs. GENSIM predicts the outcome of an experiment. A human compares the GENSIM prediction to the observed outcome of the experiment. HYPGENE's input is a tuple $\{I_A, Error_A, T\}$ that describes the experiment, the error in GENSIM's prediction as determined by a human, and the reaction theory in the process knowledge base. HYPGENE's output is a hypothesis I_A' that aligns GENSIM's prediction with the observation. The framework presented in Karp (1989) also encompasses hypotheses that produce a modified theory T', but such modifications have not been implemented in HYPGENE.

quality of GENSIM's predictions. HYPGENE formulates hypothetical modifications of what the biologists *thought* the initial conditions of the experiment were, such that GENSIM's prediction using the modified initial conditions matches the observation. It makes sense to postulate modifications to the initial conditions of an experiment because these conditions are often not known with certainty by scientists. One reason for the uncertainty is the incredible complexity of the objects in the initial conditions. For example, the sequence of much of the five million base pairs of *E. coli* DNA is unknown to scientists. This DNA is present in the initial conditions of all gene-regulation experiments. Some hypotheses in gene-regulation experiments assign a putative function to elements of that DNA, thereby postulating the presence of a new object in the initial experimental conditions. Another reason for the uncertainty is that experimental conditions are often tailored using laboratory techniques whose effects cannot be predicted with complete certainty, such as gene-splicing techniques.

1.1. Hypothesis formation as design

This article advances the view that we can solve the hypothesis-formation problem by treating it as a *design problem*, and by applying previously developed AI methods for solving design and planning problems to the hypothesis-formation task. Although the process of design is not completely understood by AI researchers, it is better understood than is the problem of hypothesis formation. Design is a creative activity in which a designer constructs an artifact[2] that satisfies a set of constraints. Traditionally, we think of designing tangible objects, such as digital circuits, bridges, and houses. We can also think of designing more abstract entities, such as a plan of action (as addressed by AI research in planning) or a computer program (addressed by research in automatic programming). All of these artifacts must satisfy constraints, and all are constructed from primitive components—the design primitives—such as transistors, suspension cables, and two-by-fours. To treat hypothesis formation as a design problem, we view a theory as an artifact that scientists construct. Theories (and hypotheses) are designed subject to the constraint that their predictions must match experimental outcomes. Theories should satisfy other constraints as well: their predictions should be testable, they should be consistent with other scientific knowledge, and they should satisfy requirements such as simplicity.

AI researchers have approached design and planning problems using the search paradigm. The search space for a design problem is the space of all possible ways of combining the design primitives. Solution states are those arrangements of the design primitives that satisfy the design constraints. The search operators are design operators that combine the design primitives into larger arrangements.

HYPGENE uses these same methods to design hypotheses. HYPGENE's initial *design goal* is to eliminate the error(s) in GENSIM's prediction. To satisfy this goal, HYPGENE employs *design operators* that reason backward from the design goal to determine what changes to the initial conditions would achieve the goal. Since HYPGENE operates by modifying an initial theory rather than by designing a theory from scratch, it is more precise to say that HYPGENE *redesigns* theories. HYPGENE examines a dependency trace created by GENSIM that shows how GENSIM derived its prediction from the initial conditions. The operators examine GENSIM's knowledge base of chemical reactions to determine what additional known reactions might have occurred in this experiment. The operators reason about what changes to the initial conditions would cause new reactions to occur, or would prevent predicted reactions from occurring, such that the prediction error is eliminated. This design problem is a search problem because often more than one operator is relevant to satisfying a given design goal, and a single operator often can be applied in several ways.

The synthetic, goal-directed search used here should prove more efficient than the generate-and-test approaches used by previous scientific-reasoning program, such as BACON, GLAUBER, STAHL, DALTON, and KEKADA (Langley et al., 1987; Kulkarni, 1988). Those approaches use heuristic search to guide hypothesis generators that blindly enumerate combinations of theory-building primitives, and test the resulting hypotheses for correctness. HYPGENE's search is focused on and constrained by the error in GENSIM's prediction, and the dependency structure of that prediction. This difference is analogous to the difference between the search-control strategies of dependency-directed backtracking and chronological backtracking—the former is usually more efficient because it incorporates information about what elements of a proposed problem solution caused the failure of that solution. Another viewpoint is that we have moved part of the test for solutions (the constraint of relevance of the prediction error) inside the solution generator. This approach was used in the Dendral project (Lindsay et al., 1980), and Bennett and Dietterich have discussed this approach as a general way of increasing the efficiency of a problem solver (Dietterich & Bennett, 1986).

1.2. The value of the design metaphor

This view of hypothesis formation as design is important both conceptually and pragmatically. Conceptually, design provides a new framework for thinking about what scientists do. Design becomes a metaphor for scientific activity in which scientists are viewed as architects of complex structures, namely, scientific theories. We view theory formation as a highly goal-directed endeavor.

More pragmatically, the design metaphor suggests a number of existing methods that we can apply to the problem of hypothesis formation. This article explores the use of only some design techniques to formulate hypotheses; among the techniques it does not explore are designing a theory at multiple levels of abstraction, maintaining a design history for a theory to facilitate future theory revisions, maintaining a library of theories so that we may create new theories by redesigning old ones, and planning about the theory-design process itself, as explored by Tong (1988).

2. A historical study of attenuation

I developed the hypothesis-formation methods described herein by studying historical instances of scientific reasoning, and I measure the success of my methods by their ability to solve reasoning problems that scientists have faced in the past. Karp (1989) reconstructed the process by which Yanofsky and his colleagues discovered attenuation. The biologists studied a set of bacterial genes called the *tryptophan operon*, or *trp operon*. The historical study is based on information obtained from the scientific publications that these biologists authored, and from interviews that Peter Friedland, René Bach, and I conducted with them. I produced a conceptual reconstruction of what knowledge the biologists possessed about the trp operon at different times. This reconstruction also describes the experiments that the biologists performed, and the alternative hypotheses that they proposed to explain the outcomes of their experiments. Thus, I identified sample hypothesis-formation problems and experiment-prediction problems that were used to drive the development of HYPGENE and of GENSIM.

One of the distinguishing features of this reconstruction is that it involves hypothesis-formation problems that are more complex and more realistic than those studied by most previous AI researchers (exceptions include MetaDendral (Buchanan & Mitchell, 1978) and Protean (Altman, 1989)). Measures of complexity include the size of object part structures, the number of object classes, the size of process preconditions and actions, and the number of processes that fire in a single experiment. Complexity is important not only because it ensures that we address realistic problems, but also because it allows us to dentify the context within which a research problem arose. In addition, the attentuation research consumed over 50 person-years of effort—the historical study contains ten times the number of hypothesis-formation examples discussed here, and could fuel future AI research in hypothesis formation. Also important is that this biological research was performed very recently (in the 1960s and 1970s), so we can be more certain of the accuracy of our knowledge of this research than of our knowledge of scientific discoveries made hundreds of years ago.

3. Molecular genetics background

This section represents a very brief overview of molecular genetics to help the reader understand the many references to that domain in this article.

The trp operon research was concerned with the regulation of the production of enzymes (which are proteins) that synthesize a nutrient called tryptophan. The *E. coli* cell requires tryptophan for growth. When tryptophan is not present in the cell's environment, it must synthesize tryptophan using three enzymes. Thus, the cell must regulate the production of these enzymes as a function of the availability of tryptophan in the cell's environment.

An *operon* is a set of genes whose synthesis is regulated as one unit. The trp operon genes encode three enzymes (proteins). Figure 2 shows the chemical reactions in the *biosynthetic pathway* for trp, the enzymes that catalyze these reactions, and the layout within the trp operon of the genes that code for these enzymes.

To understand how expression of the trp operon is regulated, we must understand how proteins are synthesized. The sequence of every protein in the cell is encoded by the cell's DNA, which is analogous to a Turing machine tape. The process of *protein synthesis* constructs proteins from the DNA blueprint in two steps. *Transcription* of DNA to mRNA is accomplished by an enzyme called *RNA polymerase*, which binds to regions of DNA called *promoters*. After binding, RNA polymerase moves along the DNA; it reads the DNA message and synthesizes a complementary mRNA copy of the DNA. *Translation* of mRNA to protein is accomplished by the *ribosome* in a similar fashion. It reads the mRNA message and synthesizes one or more corresponding proteins. Control sites within the mRNA delimit the mRNA regions that code for different proteins.

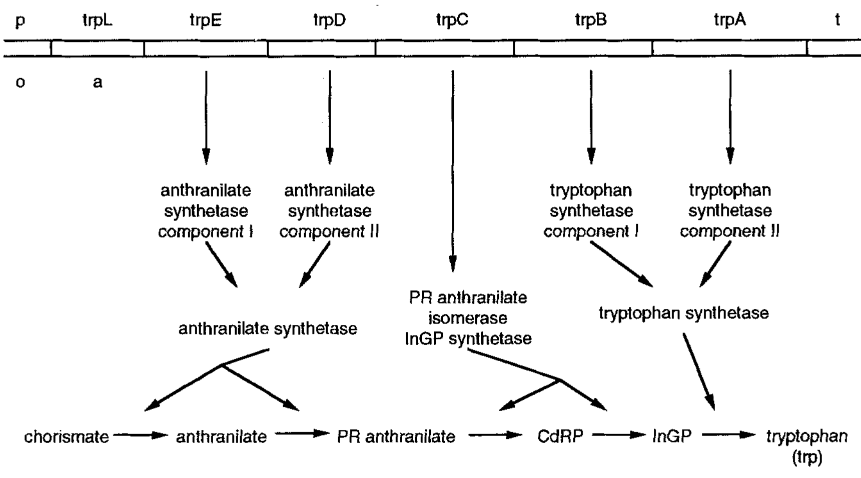
p
trpL
trpE
trpD
trpC
trpB
trpA
t
o
a
anthranilate synthetase component I
anthranilate synthetase component II
tryptophan synthetase component I
tryptophan synthetase component II
anthranilate synthetase
PR anthranilate isomerase InGP synthetase
tryptophan synthetase
chorismate
anthranilate
PR anthranilate
CdRP
InGP
tryptophan (trp)

Figure 2. The trp operon contains five structural genes: *trpA* through *trpE*. The genes are surrounded by genetic control regions called the *promoter* (p), *operator* (o), and *terminator* (t). The operon also contains a region of DNA called the *leader region* (trpL). The genes in the trp operon code for five polypeptides that form three enzymes: anthranilate synthetase, PR anthranilate isomerase–InGP synthetase, and tryptophan synthetase. These enzymes form a biosynthetic pathway that synthesizes trp from chorismate and other chemical precursors that are not shown. For example, the trpE gene codes for a component of the enzyme anthranilate synthetase; that enzyme catalyzes the conversion of chorismate to PR anthranilate in two steps. The GENSIM model of the trp operon incorporates all of the details shown in this figure.

The cell regulates the process of transcription through *Jacob-Monod repression*. At the start of the trp operon, near the trp operon promoter, is a region called the trp *operator*. A protein called the *trp-repressor* can recognize and bind to the trp operator. When bound there, RNA polymerase is unable to bind to the trp promoter. The trp-repressor protein contains a second site, to which trp can bind. When trp is *not* bound to trp-repressor, the protein is unable to recognize the trp operator, and thus cannot interfere with transcription. But when trp is bound to the repressor, the protein can bind to the operator. In this way, transcription of the trp operon varies inversely with the concentration of trp in the cell—the cell produces more of the trp enzymes when less trp is present in the cell's environment.

4. GENSIM—Representation and simulation of the trp operon

The GENSIM framework (Karp, 1989, 1991) describes the attributes, structures, and behaviors of the objects that make up the trp operon, and can predict the products of the biochemical reactions (bioreactions) that occur in different experiments. The GENSIM model of the trp operon includes a superset of both the objects and the reactions shown in figure 2 (Karp (1989, 1991) describes the exact contents of the GENSIM model). GENSIM predicts experimental outcomes by reasoning with three frame knowledge bases. A *class knowledge base* (called the CKB) defines a taxonomy of the biological objects in bacteria that are involved in regulation of the trp operon. Users describe a given biological experiment by creating a *simulation knowledge base* (SKB) whose frames represent the biological objects that are present in that experiment (each frame is an instance of a biological-object class in the CKB). Several frames may be arranged in part–whole relationships to represent the internal structure of a complex biological object. For example, the frame representation of the trp operon DNA in figure 2 consists of parts such as the *trpD* region and the *p* (promoter) region.

A *process knowledge base* (PKB) describes the bioreactions that can occur among the biochemical objects in the trp system. The PKB thus embodies a theory of the trp operon. The theory consists of a set of *processes*, where each process describes a single bioreaction. GENSIM processes are similar to the processes used in Forbus' qualitative process theory (Forbus, 1984), and also bear a similarity to production rules. A process contains preconditions that must be satisfied for a certain bioreaction to occur, and it contains actions that describe the products of that bioreaction. Process preconditions examine the properties of biological objects (frames) present in the SKB. A predicate-calculus precondition is evaluated with respect to the predicate-calculus interpretation of frames in the SKB. Process actions create new objects in the SKB with specified properties. For example, one process describes the bioreaction in figure 2 in which the tryptophan synthetase enzyme converts InGP to tryptophan.

The GENSIM simulator uses the process definitions in the PKB to determine what bioreactions occur among the objects in an experiment; bioreactions create new objects, which can cause additional bioreactions. The GENSIM framework defines a *qualitative biochemistry*—an ontology for biochemistry and a framework for reasoning about bioreactions. The ontology assigns one frame object to represent every distinct population of homogeneous molecules. The current implementation of GENSIM only reasons about reactions during an instant of time, and we assume that a population of molecules is never fully depleted during such a short period, so objects are never deleted from GENSIM simulations.

The sample process in figure 3 describes a binding reaction between the trp-repressor protein and the trp operator, which is the DNA region labeled *o* in figure 2. Table 1 explains the functions and predicates used within GENSIM processes. The process in figure 3 specifies that for any two molecules of type `Trp.Operator` and `Trp.Repressor`, if these molecules contain complementary binding sites, and if these binding sites are empty and are free of mutations that interfere with this binding reaction, then new instances of these molecules should be created and bonded together chemically as a new object. Creation of new bound instances of these two objects is called *object forking* because the population of `Trp.Repressor` molecules is split into two subpopulations—the fraction that participates in the reaction and the fraction that does not participate.

Each process is defined as a frame. The GENSIM process interpreter executes processes to simulate bioreactions in the following manner. The `Parameter.Object.Classes` slot of a process *R* (see figure 3) specifies the types of objects involved in the bioreaction that *R* describes. If the interpreter finds one frame in the SKB (the current state of the experiment) for every class in the `Parameter.Object.Classes` of *R*, then the interpreter binds the names of these frames to the variables listed in the `Parameter.Object` slot of *R*. If the interpreter finds every predicate-calculus expression in the `Preconditions` slot of *R* to be true under the variable bindings just established, then it will execute the LISP forms in the `Effects` slot of *R* to create the products of the bioreaction.

```
Parameter.Object.Classes:  Trp.Repressor Trp.Operator
Parameter.Objects:               $A          $B
Preconditions:          $B must contain an active site that interacts with objects of $A's type.
                        (EXISTS  $Bsite
                          (AND
                            (IS.PART.R $Bsite $B)
                            (OBJECT.EXISTS $Bsite Active.Sites)
                            (EXISTS $site.interaction.class
                              (AND
                                (MEMBER $site.interaction.class
                                        (GET.VALUES $Bsite Potential.Interacting.Objects))
                                (OBJECT.EXISTS $A $site.interaction.class)))))
                        That active site cannot be occupied.
                        (NOT (EXISTS $obj
                               (AND (MEMBER
                                      $obj
                                      (GET.VALUES $Bsite Object.Interacting.With.Site))
                                    (OBJECT.EXISTS
                                      $obj
                                      (GET.VALUE $Bsite Potential.Interacting.Objects))))
                        That active site cannot contain a mutation that disables the current
                        reaction.  The $Current.Process variable is bound externally by
                        the process interpreter.
                        (NOT (EXISTS $mutation
                               (AND (IS.PART $mutation $Bsite)
                                    (OBJECT.EXISTS $mutation Mutations)
                                    (MEMBER $Current.Process
                                            (GET.VALUES $mutation Processes.Disabled)))))
Effects:                Create a new object that contains $A and $B
                        as parts.
                        (CREATE.COMPLEX RepOp.Complexes  (LIST $A $B)  RBOUND)
                        Record that $A is interacting with $Bsite
                        (PUT.VALUE $Bsite Object.Interacting.With.Site $A))
                        Record that the promoters controlled by $B are
                        no longer able to bind RNA Polymerase
                        (PUT.VALUE (GET.VALUE $B Promoters.Controlled)
                                   Receptive.To.Polymerase
                                   NO)
```

Figure 3. The process Trp-Repressor.Binds.Operator. This process describes the binding of the activated trp-repressor protein to the trp operator region of DNA. Every variable used here has global scope within this process (such as $A and $Bsite). The preconditions field of the process consists of three conjuncts. The English comments describe the code fragments they precede.

Table 1. The predicates, functions, and quantifiers used within GENSIM process definitions. OBJECT.EXISTS, IS.PART, and MEMB are predicates. The symbols AND, OR, NOT may also be used, and have their standard logical meanings.

Predicate or function	Meaning
(OBJECT.EXISTS X Y)	True if object X exists within the simulation and is in the object class Y.
(IS.PART X Y)	True if object X is part of object Y.
(MEMBER X Y)	True if atom X is an element of list Y.
(GET.VALUES X Y)	The value of slot X of object Y.
(BINDV $X Y)	Binds variable $X to the value Y.
(CREATE.COMPLEX X Y)	Creates an object in class X containing the objects in list Y as parts.
(COPY.STRUCTURE X)	Creates a copy of object X.
(PUT.VALUE X Y Z)	Stores Z into slot X of object Y.
(EXISTS $X Y)	True if expression Y is true for some binding of $X.
(FORALL $X Y)	True if expression Y is true for all bindings of $X.

Process effects create new objects with specified properties in the SKB. Therefore, the final state of the SKB is the output of GENSIM—a prediction as to what new objects are created during the experiment. For experiments involving the trp operon, these new objects include the enzymes coded for by the trp operon, and the metabolic products of these enzymes, such as tryptophan.

5. Hypothesis formation

This section explains how the general framework for treating hypothesis formation as design introduced in section 1.1 is instantiated in the HYPGENE program. We begin by defining the hypothesis formation problem; then we describe HYPGENE's design goals and design operators, and the space that HYPGENE searches.

5.1. The hypothesis-formation problem

In 1973, Jackson and Yanofsky created *E. coli* mutants from which a region of the trp operon DNA had been deleted (Jackson & Yanofsky, 1973). Their theory of the trp operon said that the deleted region of DNA had no function, so the rate at which the operon was transcribed should have been unaffected by this deletion. But they observed that the rate of transcription of the trp operon increased significantly.

A scientist takes several types of actions when the predictions or her theory are not consistent with her experimental observations. She may choose to discard certain experiments because she does not deem their results trustworthy—Jackson and Yanofsky might have decided that their experiments were contaminated. She may identify certain experiments as being outside the theory's domain of applicability, and therefore irrelevant to testing the theory. She may modify her theory (her understanding of what bioreactions the objects in the trp operon can undergo) to bring its predictions in line with her observations, such as by proposing that the deleted DNA region had been inhibiting transcription of the trp operon. In the GENSIM framework, such a hypothesis corresponds to a modification of the processes in the PKB. Finally, she may postulate that the initial conditions or observed outcomes of certain experiments are different from what she originally thought they were, in a way that allows her theory to predict their outcomes correctly—one of Jackson and Yanofsky's hypotheses was that the initial conditions of their experiment contained multiple copies of the trp operon, rather than the single copy they had assumed.

The hypothesis-formation problem considered in this article is, given:

1. A theory T
2. An anomalous experiment E_A, with initial conditions I_A, whose outcome O_A does not match the outcome P_A that was predicted by T

we are asked to produce a set of hypotheses, where each hypothesis I'_A postulates modifications to the initial conditions of the experiment such that the predicted outcome P'_A of E_A now matches its observed outcome:

$$I'_A \cup T \models P'_A \qquad P'_A = O_A$$

The notation is summarized in table 2.

Table 2. Notation for describing experiments.

Notation	Meaning
E_A	An anomalous experiment
I_A	Initial conditions of E_A
P_A	Predicted outcome of E_A
O_A	Observed outcome of E_A
$Error_A$	Error in predicted outcome of E_A

$$\begin{tikzcd}
\texttt{Trp.Operator.1} \arrow[rr, "\textit{(trp-repressor.binds.operator)}"] & & \texttt{RepOp.Complexes.1} \\
\texttt{Trp-ApoRepressor.1} \arrow[r] & \texttt{Trp-Repressor.2} \arrow[ur] & \\
\texttt{trp.4} \arrow[ur, "\textit{(trp-aporepressor.binds.trp)}"'] & &
\end{tikzcd}$$

Figure 4. The predicted outcome of the repression reactions in the Hiraga experiment. First tryptophan (`trp.4`) binds to the trp-aporepressor protein via the process `Trp-ApoRepressor.Binds.Trp`. Then the activated repressor binds to the trp operator region of DNA via the process `Trp-Repressor.Binds.Operator`.

We will use the following example throughout this section to illustrate HYPGENE's methods. In 1969, Hiraga reported on experiments that sought to characterize mutant *E. coli* strains in which the trp operon did not exhibit the same regulatory characteristics as did normal *E. coli* strains. The rate of transcription of the trp operon was not sensitive to the concentration of tryptophan because the repressor protein did not bind to the trp operator DNA (Hiraga, 1969). Hiraga sought hypotheses to explain what aspects of the cellular machinery involved in trp-operon regulation had been altered by those mutations.

The theory of the trp operon in 1969 predicted that the operon was regulated by the trp repressor protein, through the two bioreactions shown in figure 4 (Figure 3 shows the GENSIM process representing the second reaction). In the first reaction the aporepressor is activated by binding tryptophan; in the second reaction the activated repressor binds to the trp operator region of DNA. I_A for this experiment included tryptophan (`trp.4`), the trp aporepressor protein (`Trp-ApoRepressor.1`), and the trp operon DNA (`Trp.Operator.1`). GENSIM predicts that the two reactions in figure 4 occur and yield the activated repressor and the repressor-operator complex as P_A. But no repressor-operator complexes were observed (in O_A) in Hiraga's mutant strains, which is the anomaly to be explained.

5.2. *HYPGENE's design goals*

Because HYPGENE is a designer of hypotheses, its input is viewed as a design goal. That goal is to eliminate the difference between the predicted and observed outcomes of an experiment—the prediction error. This error consists of all assertions contained by the prediction *or* the observation that are not common to both. It can also be described as the assertions that are present in the prediction but should not be, plus the assertions that are present in the observation, but are missing from the prediction.

$$Error_A = (P_A \cup O_A) - (P_A \cap P_A)$$
$$Error_A = (P_A - O_A) \cup (O_A - P_A)$$

The two terms in the second equation can be thought of as a *delete list* (DL) and an *add list* (AL). We can transform the set P_A into O_A by adding AL to P_A, and by removing DL from P_A. Thus:

$$Error_A = AL \cup DL$$
$$O_A = P_A \cup AL - DL$$

This definiton of predicton error corresponds to one of HYPGENE's two classes of design goals—goals that involve false assertions in P_A (the delete list), or assertions that are missing from P_A (the add list). These assertions represent objects with specified properties that should be removed from or added to the prediction, respectively. In the Hiraga experiment, the $Error_A$ delete list contains an assertion that represents the existence of a repressor-operator complex (existence of this complex should be deleted from P_A). The AL and DL concepts in this article are much the same as those used by the STRIPS planner (Fikes & Nilsson, 1971). More generally, HYPGENE's design framework is modeled after the STRIPS planning framework in a number of respects, and much of HYPGENE's operation can be understood by analogy to STRIPS.

The second type of design goal generates hypotheses that would alter the predicted concentrations (quantities) of existing objects in P_A. Even though GENSIM does not compute quantitative predictions, HYPGENE can compute quantitative hypotheses that explain why the amount of an object in a prediction is too high or too low. Such hypotheses do not completely remove a population of molecules from a prediction, nor create a population where it did not exist before, so the assertions about what objects exist in a prediction are unchanged. Although we could think of the prediction of the concentration of an object present in O_A as an assertion, these quantitative assertions combine in a special way (according to the laws of arithmetic), so a special set of operators is required to reason about them.

5.2.1. Add and delete goals

Design goals that involve the addition or removal of assertions from P_A are represented using predicate calculus. Existentially quantitied formulae are used to specify that some object with given properties should exist in P_A, and universally quantified goals are used to specify that no objects with given properties should exist in P_A.

In the Hiraga experiment, the goal of removing the repressor-operator complex from P_A would be encoded as follows:

```
(FORALL $X (NOT (OBJECT.EXISTS $X RepOp.Complexes)))
```

5.2.2. *Quantitative design goals*

GENSIM predictions do not contain a quantitative component because GENSIM does not predict the concentrations of the objects it creates in a simulation. But if the user of HYPGENE has a separate quantitative theory that predicts the concentration of an object in a simulation, and if the user determines that this concentration differs from the observed concentration of the object, then the user can employ HYPGENE to formulate hypotheses to account for this discrepancy. Quantitative design goals direct HYPGENE to either increase or decrease the concentration of some object. These goals are specified using the predicates `Increase.Quantity` and `Decrease.Quantity`.

For example, in the Jackson–Yanofsky experiment the observed concentration of mRNA was higher than the concentration they predicted. This anomaly is described to HYPGENE by using GENSIM to predict the outcome of the experiment (that is, what reactions occur and what objects are created), and by then giving HYPGENE the following goal (where `Messenger.RNAs.15` is the mRNA in the GENSIM prediction whose concentration must be increased):

```
(Increase.Quantity Messenger.RNAs.15)
```

5.3. *Design operators*

The *goal list* maintains a conjunctive list of the goals that HYPGENE is currently trying to satisfy. Each goal is a predicate calculus formula that contains no conjunction or disjunction. During the design process, the HYPGENE planner matches design goals on the goal list against design operators in order to satisfy these goals. Most design operators have preconditions that must be satisfied in order for the operator to be executed. When HYPGENE selects an operator to satisfy a certain goal, it adds the operator preconditions to the goal list (a form of subgoaling). These additional goals will lead to the selection of additional operators, causing HYPGENE to reason backward from its initial design goals. Once the preconditions of an operator have been satisfied, the operator is executed to yield changes to I_A that may satisfy other goals on the goal list.

Every change to the goal list, as caused by selection or execution of an operator by HYPGENE, yields a new state in the HYPGENE search space. This space can be thought of as an AND/OR tree, since in every state the goal list contains a conjunctive list of goals. New OR states are created in two situations: when multiple operators are applicable to a given goal, and when the preconditons of operators that are added to the goal stack contain disjunctions. The branching factor of the search space (the number of OR branches that may lead to alternative solutions) depends on the number of operators that are applicable to satisfying each design goal, and to the number of disjunctions within the preconditions of each operator.

Currently HYPGENE performs an exhaustive search of its search space. But in anticipation of hypothesis-formation problems for which this policy is infeasible, the search is conducted in a best-first fashion using an agenda mechanism. The evaluation function gives a better rating to search states with fewer changes to I_A. Thus, simpler hypotheses are generated first. The computational complexity of HYPGENE is analyzed in Karp (1989).

A given search state results from a series of operator executions and subgoaling operations. Each state contains a description of the goal list, of I'_A, and of P'_A in the context of that state. The description of I'_A in a given state reflects the changes made to I_A by the operators whose execution yielded that state. For example, an operator might postulate the presence of an additional object in I'_A. As I'_A evolves, HYPGENE calls on GENSIM to incrementally recompute P'_A to reflect the predicted outcome of I'_A. In the course of executing an operator, that operator may examine I'_A, P'_A, the PKB, and the (state specific) simulation dependency trace that reflects the derivation by GENSIM of P'_A from I'_A.

The operators that achieve each type of goal alter the causal structure of the prediction P_A in several ways. This causal structure is reflected in the GENSIM simulation dependency trace, which links the firing of a particular process (the occurrence of a bioreaction) with the objects created by that process, and with the objects that caused the process to fire. For example, an operator that is invoked to remove an object from P_A might analyze the preconditions of the process whose firing created that object to determine how to violate some precondition, thus inhibiting the process from firing.

The word "operator" is used in a different sense in this article than in the STRIPS literature. A STRIPS operator has an add list and a delete list that is matched against the goal stack directly. The HYPGENE analogue to a STRIPS operator is a process—HYPGENE computes an add list for each process in a manner described the next section. Some HYPGENE operators retrieve a process whose execution will achieve a particular goal on the goal stack. Other operators manipulate I'_A directly to achieve a particular goal, without affecting the execution of a process.

5.3.1. Design operators that add or remove assertions

An assertion B could be present at the end of the simulation (and thus be an element of P_A) for only one of two reasons:

- Because B was present in I_A (the presence of B in P_A follows from the monotonicity of GENSIM simulations, i.e., the fact that objects are never deleted from simulations)
- Because the execution of a process asserted B

Thus, one HYPGENE operator adds an assertion B to P_A by addition B to I_A; several other operators modify I_A such that a process that asserts B fires in the current prediction. Preventing an assertion B from existing in P_A is more complex than adding B because *all* causes of B must be eliminated, that is, B must not be present in I'_A and no process may assert B. Note that because the GENSIM ontology guarantees that objects are never deleted from simulation by the firing of a process, there is no operator that attempts to remove an assertion from P_A by firing a process.

Figure 5 shows HYPGENE's assertion-altering design operators. Although the operators are implemented as LISP functions, they are organized conceptually in a class-subclass hierarchy that is used to control their execution (discussed in more detail in Karp (1989)). Only leaf nodes of this tree are operators that can be executed during problem solving. We now describe each operator for adding and removing assertions in detail.

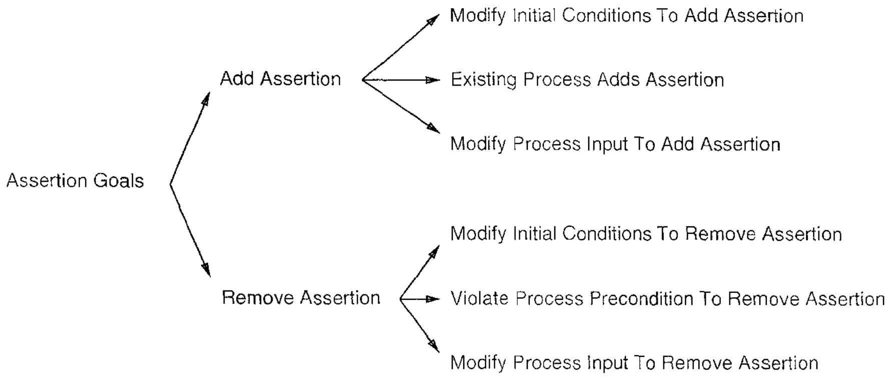
Assertion Goals
Add Assertion
Remove Assertion
Modify Initial Conditions To Add Assertion
Existing Process Adds Assertion
Modify Process Input To Add Assertion
Modify Initial Conditions To Remove Assertion
Violate Process Precondition To Remove Assertion
Modify Process Input To Remove Assertion

Figure 5. The hierarchy of assertion design operators. The leaves in the tree are executable operators; they are grouped into more general classes, such as the set of operators that remove an assertion from P_A.

Modify Initial Conditions Directly. The operator "Modify Initial Conditions To Add Assertion" satisfies the goal of adding assertion B to P_A by postulating that B was present in I_A. Conversely, operator "Modify Initial Conditions To Remove Assertion" removes B from P_A by postulating that B was absent from I_A. A precondition of this operator is that B was initially presented in I_A. For example, the goal of removing all tryptophan objects from P_A would be represented as

```
(FORALL $X (NOT (OBJECT.EXISTS $X trp)))
```

Enable a Process to Add an Assertion. The operator "Existing Process Adds Assertion" attempts to add an assertion B to P_A by searching the PKB for a process that, if executed, would assert B. For example, if the goal is that an object of type repressor-operator complexes existed in P_A:

```
(EXISTS $x (OBJECT.EXISTS $x RepOp.Complexes))
```

then this operator finds all processes in the PKB that create an object of type `RepOp.Complexes`. Once the operator has found a process R that can satisfy its goal, it must attempt to make the process fire. To do so, it posts a new design goal on HYPGENE's goal list that specifies the conditions that must be true for R to fire: objects of the types specified in the parameter-object classes of R (explained in section 4) must exist, and the preconditions of R must be satisfied.

The task of finding a process with effects that satisfy a given goal is not as simple for HYPGENE as it is for STRIPS (Fikes & Nilsson, 1971). To find an operator that satisfies a goal from its goal stack, STRIPS matches the goal against the add list and delete list of each operator. This task is conceptually simple because the goal and the add list are represented in the same declarative language. Not so for HYPGENE, because the `Effects` field of many PKB processes do not consist of fixed lists of propositions to add and delete. Some process `Effects` call LISP functions that perform complicated tasks, such as copying an object and all of its parts when those parts are arranged in arbitrary tree structures (as is the case for a number of objects in this domain). These functions contain recursion, so it is difficult to reason about their effects to infer what objects (and assertions) they create.

HYPGENE employs an incomplete solution to this problem. A preprocessor computes an add list for most processes (no delete list is needed because of the GENSIM assumption about object deletion (section 4)) by simulating the execution of processes on typical parameter objects. For each process R, the preprocessor creates an object from each parameter-object class of R (in an otherwise empty SKB). Then the preprocessor executes R on these objects. The add list of R is created from a post-mortem analysis of the objects created by R.

The weakness of this method is that it considers only a single execution of R, because the add list it computes for R is based on a single set of prototypical parameter objects. Consequently, this method will work when either of the following conditions holds of a process R: the effects of R are not dependent on the properties of its parameter objects (such as their parts or slot values); or, the effects of R do depend on the properties of its parameter objects, but every combination of parameter objects can be pre-enumerated—an add list must be computed for each combination. These conditions hold for most but not all of the processes in the trp system, so this operator cannot reason fully about some of the processes in the PKB.

Disable a Process to Remove an Assertion. The operator "Violate Process Precondition to Remove Assertion" is used to remove an assertion from P_A that was asserted by a process R. It does so by preventing R from firing. R fired because its parameter objects were present in the SKB, and its preconditions were satisfied. To prevent R from firing, HYPGENE must make one of these conditions false. To do so, this operator posts as a new goal the negation of the predicate-calculus formula that specifies the conditions under which R fired; those conditions are retrieved from the simulation dependency trace.

In the Hiraga experiment, the repressor-operator complex was created by the firing of the process in figure 3. One precondition of this process is that the trp operator does not contain a mutation that disables this bioreaction. HYPGENE causes this precondition to be violated by postulating that such a mutation does in fact exist.

Modify Process Inputs to Alter Process Effects. HYPGENE employes the operators "Modify Process Input To Add Assertion" and "Modify Process Input To Remove Assertion" when a design goal requires the modification or deletion of an object C that was created by a process R. These operators, which change the outputs of the process by modifying its inputs, are applied when the existence and properties of C can be varied by varying the properties of the parameter objects of R, that is, when the effects of R are a function of its inputs (which is usually the case). This type of operation has been studied in planners by Chapman and Pednault (Pednault, 1988; Chapman, 1987).

In order to infer what modifications (if any) will be sufficient, these operators reason about the effects of R. Such reasoning is very difficult in general because of the procedural nature of some process effects. These operators employ an approximate solution to this problem that is widely applicable in this domain. As discussed in section 4, most chemical reactions in this domain involve object forking; thus, many processes copy (fork) their parameter objects and modify the copies in some way. If process R copies parameter object

B to a new object B', the properties of B' are largely derived from the properties of B. Therefore, if B is modified before R fires, the properties of B' will likely be modified in a similar fashion. These operators replace all occurrences of B' in HYPGENE's goal list by B. Since *all* properties of B' were not derived from B—and in some cases, B' may not have been copied from any parameter object—these operators sometimes fail to modify B' in the desired way. In the process in figure 3, for example, the new `RepOp.Complexes` object contains a copy of the `Trp.Repressor` and `Trp.Operator`. The copies differ from the original objects in that they are bound together into a complex.

5.3.2. Quantity-hypotheses design operators

Quantitative design goals aim to increase and to decrease the amount of an object in a prediction. Since cells almost never function at chemical equilibrium, and since GENSIM predictions span very short time intervals, these operators formulate changes to quantities that are not at equilibrium.

An understanding of these operators requires an understanding of a few basic principles of the kinetics of chemical reactions, and of how these principles *would be* modeled with processes that operate on quantities, such as those of Forbus (1984). GENSIM processes do not compute with quantities, and therefore utilize only a subset of the Forbus model. Nevertheless, HYPGENE is able to construct quantitative hypotheses based on quantitative design goals formulated by a human. In Forbus' framework, the concentration of a chemical compound (object) would be represented as a quantity. Bioreactions that create that object as a product increase its concentration, and bioreactions that consume that object as a precursor decrease its concentration. These effects are referred to as *positive influences* and *negative influences*, respectively. The rate at which bioreaction occurs and, therefore, the rate at which the precursors of the bioreaction are consumed and the products are created, is proportional to the concentrations of the precursor objects. In some cases the relationship may be nonlinear; the constant of proportionality is unique to every reaction. The final concentration of every object is influenced by three factors: the negative influences, the positive influences, and the initial concentration of the object. Each quantity-hypothesis operator works by altering one of these three factors.

A detailed description of the quantity-hypothesis design operators is given in the appendix.

5.3.3. Possible process and class-KB modification operators

The preceding operators are unable to design changes to the process definitions that comprise HYPGENE's theory T, and they are unable to propose changes to the object class KB. My dissertation (Karp, 1989) gives a detailed proposal for operators of both these types, although none of these operators have been implemented.

Like the initial-condition design operators, each process-design operator addresses design goals of either adding assertions to or removing assertions from P_A. The process-modification operators modify the effects, preconditions, and parameter objects of a process to achieve these goals. For example, in order to remove from P_A an assertion that was created by a process firing, one operator would specialize the preconditons of that process in order to prevent the process from firing.

5.4. *Detection of valid hypotheses*

Whenever HYPGENE proposes a change from I_A to I'_A, GENSIM incrementally computes the new prediction P'_A associated with the new I'_A. HYPGENE then examines each design goal on its goal list to see if it is true in the context of P'_A, that is, to see which of its design goals are satisfied. When all the goals are satisfied, HYPGENE has found a solution to its current hypothesis-formation problem.

6. Trials of HYPGENE

This section describes experiments that I conducted with the HYPGENE program. These experiments demonstrate that the methods employed by HYPGENE are sufficient to solve some realistic hypothesis-formation problems in the trp operon domain. The word *experiment* is potentially ambiguous in this section because it can refer to both computer-science experiments that involve HYPGENE, and to biological experiments. Henceforth, I use the word *trial* to refer to experiments with HYPGENE.

A general point of interest is that for the two trials discussed, both HYPGENE and the biologists generated more than one acceptable hypothesis. A number of past AI researchers have assumed that a hypothesis-formation program should generate a single best hypothesis. But in the publications describing the two experiments discussed here, and in a number of other experiments from the attenuation literature, biologists published a set of roughly 2-4 hypotheses, all of which they deemed worthy of significant consideration.

Two of the inputs to GENSIM and HYPGENE were the same for all these trials—the theory (PKB) and the class knowledge base (CKB). The PKB contained 16 processes that described the following bioreactions: transcription of DNA (initiation, elongation, and termination), translation of mRNA (initiation, elongation, and termination), regulation of transcription via repression (such as the process in figure 3), synthesis of tryptophan (figure 2), and a few other miscellaneous bioreactions. The input that varied was the anomalous biological experiment presented to the program. Both of these biological experiments are taken from the history of attenuation in Karp (1989).[3]

6.1. *Transcription-initiation hypotheses*

This trial involves the transcription-initiation experiment performed by Hiraga that was introduced in section 5.1. HYPGENE produced 15 solutions to this problem that are summarized in figure 6. These hypotheses are complete and correct with respect to the hypotheses proposed by Hiraga. A detailed trace of the reasoning that HYPGENE used to derive one of these solutions is given by Karp (1989, 1990).

In brief, the hypotheses that HYPGENE formulated reflect the fact that the repressor-operator complex was created by two bioreactions (figure 4). HYPGENE reasoned backward through these process executions and dissected their preconditions to determine all possible ways that either process might be prevented from firing. The resulting hypotheses postulate that certain objects present in I_A were in fact absent (such as the trp repressor), and also

Prevent firing of process Trp-Repressor.Binds.Operator by:

1. Retracting existence of trp operator DNA in I_A.
2. Retracting that Trp.Operator.Repressor.Binding.Site.1 is able to bind to Trp-Repressors.
3. Retracting that Trp-R.Operator.Binding.Site.2 is able to bind to Trp.Operators.
4. Asserting that another object is already bound to Trp.Operator.Repressor.Binding.Site.1.
5. Asserting that a mutation exists in Trp.Operator.Repressor.Binding.Site.1 that disables process Trp-Repressor.Binds.Operator.
6. Asserting that a mutation exists in Trp-R.Operator.Binding.Site.2 that disables process Trp-Repressor.Binds.Operator.

Prevent firing of process Trp-ApoRepressor.Binds.Trp by:

1. Retracting existence of trp aporepressor in I_A.
2. Retracting existence of tryptophan in I_A.
3. Retracting that Trp-R.Trp.Binding.Site.2 is able to bind to tryptophan.
4. Asserting that another object is already bound to Trp-R.Trp.Binding.Site.2.
5. Asserting that a mutation exists in Trp-R.Trp.Binding.Site.2 that disables process Trp-ApoRepressor.Binds.Trp .

Figure 6. Hypotheses formulated by HYPGENE to account for the absence of repressor–operator complexes in a transcription-initiation experiment. Every hypothesis prevents one of the two processes in figure 4 from firing. The last hypothesis proposes that a mutation object exists, that the mutation is part of the binding site within the trp-repressor protein that binds to trp, and that the functionality of this mutation is such that it interferes with the process trp-ApoRepressor.Binds.Trp.

propose modifications to the properties of objects in I_A, such as that binding sites lost their activity because they were occupied by inhibitors, or contained mutations.

6.2. Attentuation hypotheses

The experiments described by (Jackson and Yanofsky (1973) were among the most important experiments involved in the discovery of attenuation. Section 3 summarizes what biologists knew about the trp operon in the early 1960s. The experiments of Yanofsky and his coworkers showed that the model was flawed in the following respects. Although the rate of transcription of the trp operon is indeed regulated by repression, the operon is also regulated by a second, previously unknown mechanism called *attentuation*. Just past the trp operator is a region of DNA called the *attentuator* (region *a* in figure 2). As RNA polymerase reads the trp operon during transcription, the attenuator sometimes signals RNA polymerase to prematurely terminate transcription. The frequency of termination depends on the concentration of trp within the cell. At high trp concentrations, the attenuator causes frequent termination of transcription (an attenuation of the transcription process), whereas when trp concentration is low, RNA polymerase usually continues transcription.

This trial involved a pair of experiments by Jackson and Yanofsky (1973) that were instrumental in the development of the theory of attenuation. Those experiments compared the rate of trp-mRNA production (the rate of transcription) in two different *E. coli* strains. The first strain had a nonfunctional trp-repressor protein; the second strain had both a nonfunctional repressor and a deletion in the leader region of the trp operon DNA. Jackson and Yanofsky predicted that the rate of transcription should be the same for both experiments because the leader region was not known to have any function at all, so deleting this region should have no effect. However, Jackson and Yanofsky were startled to observe that the second strain transcribed the trp-mRNA at a significantly higher rate than did the first strain—the reason being that they had deleted the attenuator region.

Jackson and Yanofsky proposed several hypotheses to account for the discrepancy in trp-mRNA concentrations:

1. The second strain contained multiple copies of the trp operon, as opposed to the single copy in the first strain
2. The second strain degraded trp-mRNA at a lower rate than did the first strain
3. The gene-splicing technique that they had used to delete the leader-region from the second strain had created a second promoter in the leader region, yielding a second start point for transcription.
4. The leader region contained a site that decreased mRNA production in the first strain; this site was removed by the deletion, yielding an increased rate of transcription in the second strain (later experiments shows that this hypothesis was correct; it describes attenuation)

GENSIM prediction is shown in figure 7. HYPGENE was given the goal of explaining the elevated quantity of trp-mRNA in the second experiment.

```
(Increase.Quantity Messenger.RNAs.30)
```

HYPGENE used its Increase Quantity operators to reason backward through the dependency trace in figure 7. These operators are listed in figure 11 and described in the appendix. HYPGENE's solutions are shown in figure 8. HYPGENE formulated all the hypotheses formulated by Jackson and Yanofsky except for hypothesis 3; later in this section we consider why it missed that hypothesis. HYPGENE also formulated additional hypotheses that the biologists did not propose. Most of these hypotheses were then rejected by HYPGENE when

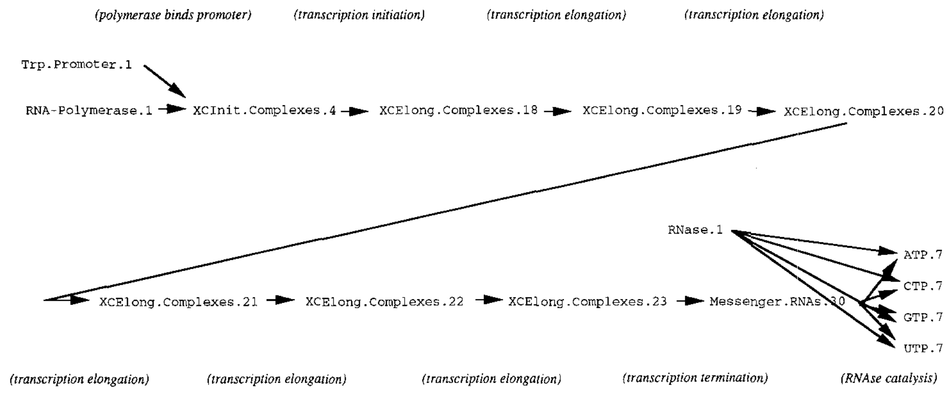

Figure 7. The outcome of the Jackson-Yanofsky experiment as predicted by GENSIM. Each `ICElong.Complex` object represents a transcription-elongation complex that includes a growing trp-mRNA. `Messenger.RNAs.30` is the full length, released trp-mRNA. The names of the processes that describe a reaction are in italics and parentheses.

A. Decrease intrinsic rate of process RNAse.Catalysis
B. Increase starting concentration of Trp.Promoter.1
C. Increase starting concentration of RNA-polymerase.1
D. Assert existence of object Trp.Leader.ED102.1, an instance of class Leaky.Terminators, in I_A
E. Increase intrinsic rate of process Polymerase.Binds.Promoter
F. Increase intrinsic rate of process Transcription.Initiation
G. Increase intrinsic rate of process Transcription.Elongation
H. Increase intrinsic rate of process Transcription.Termination

Figure 8. Hypotheses formulated by HYPGENE to account for the unexpectedly high transcription of the trp operon (the high amount of trp-mRNA) in the Jackson-Yanofsky experiment.

it used one of the hypothesis-filtering mechanisms associated with reference experiments (described in Karp (1989)). Later in this section we identify additional knowledge that can be used to reject the remaining extra hypotheses. First, we consider how HYPGENE produced hypothesis 4 (hypothesis D in figure 8).

6.2.1. Formulation of hypothesis D

The problem described by Jackson and Yanofsky (1973) involves accounting for a *relative* increase in the mRNA present in *two* experiments with very similar initial conditions: the reference experiment E_R and the anomalous experiment E_A. In the situation of a relative quantitative difference between the outcomes of two similar experiments, we invert the normal use of HYPGENE's quantitative operators. Hypothesis D is generated from the design goal of *decreasing* the trp-mRNA in E_R, rather than the goal of *increasing* the trp-mRNA in E_A. The resulting hypothesis involves a change to the initial conditions of E_R (I_R). We further postulate that this change was not present in I_A, thereby accounting for the differing outcomes of the two experiments.

HYPGENE's operator "Add Consumption Path To Decrease Quantity" formulated this hypothesis by proposing that an existing process called "leaky transcription termination" occurs in E_R, thus diverting away some of the transcription elongation complexes (XCElong.Complexes) that eventually produce a full-length trp-mRNA (see figure 9). HYPGENE also proposes that this additional reaction is *missing* in E_A, thus removing the diversion of transcription-elongation complexes, and increasing the amount of mRNA in E_A relative to E_R. Note that this hypothesis asserts that the additional process in E_R had been present all along, so the rate of transcription initiation must have been higher than they had thought to have yielded the observed amount of mRNA. This model says that only some of the transcription-initiation complexes transcribe the full length of the trp operon.

Two other points are of interest for this hypothesis. First, to make the leaky transcription termination process fire in E_R, HYPGENE postulates that the leader region of the operon contains a leaky transcription terminator. This location is essential because the only difference between the two experiments is the deletion of the leader region in E_A—that deletion must be responsible for the differing outcomes of the two experiments. This hypothesis captures half of the properties of the attenuator region as formulated by the trp researchers; the researchers also proposed that the rate of attenuation was dependent on trp concentration, as shown by later experiments (Bertrand & Yanofsky, 1976). Second, HYPGENE did

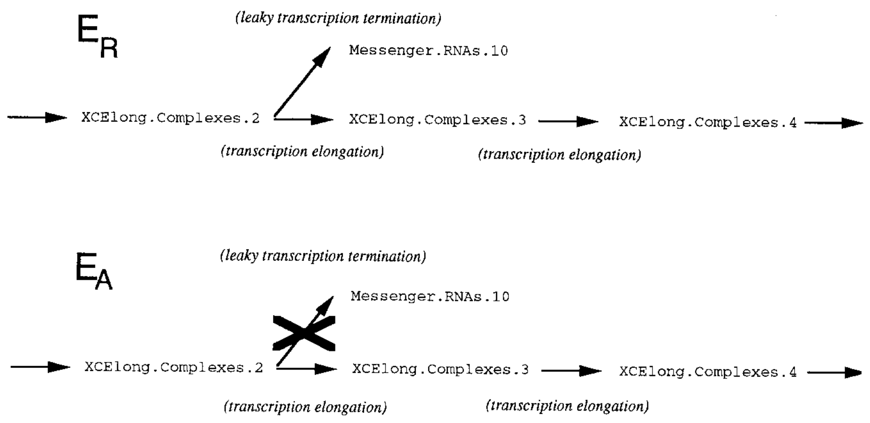
E_R
(leaky transcription termination)
Messenger.RNAs.10
XCElong.Complexes.2
XCElong.Complexes.3
XCElong.Complexes.4
(transcription elongation)
(transcription elongation)
E_A
(leaky transcription termination)
Messenger.RNAs.10
XCElong.Complexes.2
XCElong.Complexes.3
XCElong.Complexes.4
(transcription elongation)
(transcription elongation)

Figure 9. One of HYPGENE's solutions to the Jackson–Yanofsky experiment—hypothesis D. HYPGENE accounted for an increase in trp concentration in E_A by postulating that a leaky transcription termination process was decreasing the rate of synthesis of the full-length trp-mRNA in E_R, but that this process was not occurring in E_A. The leaky transcription termination process could occur because HYPGENE postulated that the trp leader region (Trp.Leader.ED102.1) was really a leaky transcription terminator (a modification to I_A).

not formulate the attenuator out of thin air; it combined the existing concept of the trp operon with the existing concept of the leaky transcription terminator. The CKB contains descriptions of both leaky and nonleaky transcription termination sites, and the PKB describes processes of both leaky and nonleaky transcription termination (the difference being that only nonleaky transcription termination can occur at a nonleaky terminator, whereas *both* leaky transcription termination and transcription elongation can occur at a leaky transcription terminator). Yanofsky (1989) confirms that this knowledge is historically accurate; in the early 1970s biologists believed that both types of terminators might exist—biologists had not shown whether or not transcription terminators were leaky, and they acknowledged both possibilities.

6.2.2. The missing hypothesis

HYPGENE missed hypothesis 3. This hypothesis could not be generated from the goal (Increase.Quantity Messenger.RNAs.30), because it does not precisely satisfy this goal. Hypothesis 3 proposes that the deletion of the leader region coincidentially produced a new promoter in the trp operon where the edges of the operon that bordered on the deletion were spliced together (biologically, it is improbable, but possible, that juxtaposition of two random sequences would produce a promoter). The new promoter would produce an mRNA that is similar to, but slightly shorter than, the normal mRNA produced by the trp operon (Messenger.RNAs.30). Strictly speaking, if the cell produced more of this new mRNA, the cell would *not* increase the quantity of Messenger.RNAs.30 because the sequences of these mRNA molecules differ. However, the experimental techniques that Jackson and Yanofsky used to detect mRNA would not distinguish the shorter mRNA from Messenger.RNAs.30, so the biologists's measurements *did* indicate an increase in trp-mRNA.

HYPGENE must be given a slightly different design goal if it is to generate this hypothesis, namely, it must have the goal of increasing the quantity of either `Messenger.RNAs.30` or any other mRNA that would be indistinguishable from `Messenger.RNAs.30`, *given the experimental techniques in use for detecing mRNA*. To satisfy this goal, HYPGENE would need knowledge of the techniques that the biologists used to measure mRNA, and of what mRNA species these techniques would and would not be able to distinguish.

6.2.3. *The extra hypothesis*

HYPGENE generated several hypotheses that were not proposed by the biologists. One hypothesis explains the increased mRNA in E_A by postulating that the level of RNA polymerase was elevated in I_A (figure 8, hypothesis C). The biologists did not propose it because they know that RNA polymerase is generally present in excess in the cell; that is, increasing the concentration of RNA polymerase will not alter the rate of transcription intiation. This type of knowledge could easily be added to HYPGENE if the GENSIM ontology included a notion of quantity such as that developed by Karp and Friedland (1989). Similarly, HYPGENE proposed (figure 8, hypotheses F, G, H) that the rates of the transcription initiation, elongation, and termination processes, and that of the polymerase-binds-promoter process, were decreased. The biologists ruled out these explanations because previous studies of transcription had showed that the rates of all these processes were generally constant across all *E. coli* strains (Yanofsky, 1989).

The other extra hypotheses produced by HYPGENE were similar to hypotheses D in figure 8. As HYPGENE worked backward through the reaction network in figure 7, attempting to find an explanation for the increased concentration of `Messenger.RNAs.30`, it created hypotheses that proposed that the leaky transcription terminator lay within every DNA segment within the trp operon, such as the gene trpA in figure 2. The biologists ruled out these hypotheses because the deletion occurred in the leader region, and thus the deletion could remove only a leaky transcription terminator that was present in the leader region, and not one that was in the trpA gene.

7. Related work on hypothesis formation

The previous work most similar to HYPGENE's hypothesis-formation methods includes Rajamoney's COAST (Rajamoney, 1988), Dietterich's PRE (Dietterich, 1984), and Simmons' GORDIUS (Simmons, 1988) (see Karp (1989) for a detailed analysis). All of these researchers have used what can be called dependency-directed or causal-based reasoning to formulate hypotheses, rather than a generate-and-test approach. These systems reason backward through a dependency structure that was created during the computation of a prediction, in order to rectify errors in that prediction. Usually the dependency structures bear some similarity to the structures maintained by a truth maintenance system—Simmons in fact uses an ATMS to represent dependencies in GORDIUS' geologic predictions. GORDIUS first determines what assumptions each error in a prediction depends on. It then formulates a hypothesis by modifying an assumption in a way that can be predicted to eliminate the error in the prediction. Simmons developed a taxonomy of assumption types and mechanisms for altering assumptions that correspond to HYPGENE's design operators.

The PRE program uses experiments in the form of commands typed to the UNIX operating system, in conjunction with a theory of UNIX, to infer the state of the UNIX file system. PRE uses constraint-propagation techniques to reason backward through its theory of a given UNIX command to determine what the file-system input to a command must have been to produce a given output. Rajamoney's COAST revises theories of naive physics that are expressed using Forbus' qualitative process theory (QPT). COAST analyzes a dependency trace to identify the process responsible for an erroneous prediction, and uses process-modification operators to alter the definition of that process. For example, they might specialize the preconditions of a process in order to prevent that process from firing. Rajamoney's process-modification operators bear a great deal of similarity to the process-design operators proposed for HYPGENE in Karp (1989).

Despite these basic similarities in the reasoning mechanisms that these programs use to formulate hypotheses, the systems vary in a number of respects. Because they use different languages to represent their domain theories, their reasoning mechanisms vary somewhat according to the language constructs they must interpret. For example, because of the more quantitative nature of its geologic predictions, GORDIUS' hypothesis-formation reasoning includes a significant amount of algebraic manipulation that is absent from the other systems.

Another variation among these systems is the exact sets of inputs and outputs that they produce. Recall that HYPGENE's input is a tuple $\{I_A, Error_A, T\}$, and its outputs are modifications to I_A. In contrast, COAST proposes modifications to T only, and not to I_A. GORDIUS proposes modifications to I_A, but not to T. PRE starts with $\{O_A, T\}$ and infers I_A (the state of the UNIX file system). Only the HYPGENE's design framework provides for modifying both the initial experimental conditions and the theory. For example, constraint propagation techniques by definition do not modify the set of constraints in the problem.

8. Limitations

The descriptions of I_A and T that we supply to HYPGENE and GENSIM must allow GENSIM to compute a predicted outcome for the experiment; for some problems we cannot predict an experimental outcome from a partial description of I_A.

HYPGENE has a limited ability to regress design goals through process effects to determine the relationship between the input objects and the output objects of a process. Some process effects call recursive LISP procedures, which are notoriously difficult for a program to reason about. AI researchers have not developed a general theory of goal regression, but HYPGENE needs one because process definitions could in principle be arbitrarily complex. The lack of such a theory has not been a problem for the test cases we have run, because HYPGENE contains heuristics (summarized in section 5.3.1) to solve a limited set of goal-regression problems.

Another assumption of HYPGENE's reasoning is that, to satisfy domain-specific goals such as (IS.PART X Y) (object X is part of object Y), HYPGENE depends on the existence of a LISP function that satisfies the IS.PART predicate—by making X a part of Y. Defining such functions is problematic for predicates that do not have a unique inverse. For example, if there were many different ways of making X as part of Y, HYPGENE would be forced to explore all of the possible ways, assuming that they could be enumerated. Although one possiblity would be to formulate an abstract hypothesis that does not say in exactly what way X is part of Y, since GENSIM must simulate every hypothesis proposed by HYPGENE to compute P'_A, the simulator must be able to execute such abstract hypotheses.

HYPGENE's design operators reflect the GENSIM property that process effects do not delete objects from the simulation (see section 4). If we extended GENSIM's qualitative chemistry such that it no longer incorporates this assumption, we would have to augment HYPGENE's operators. For example, we would create an initial-condition design operator that removes an assertion from a prediction by firing an existing process that removes the assertion.

9. Contributions

This article has shown that it is profitable to treat hypothesis formation as a design problem. Design provides both an interesting metaphor for formulating hypothesis-formation problems and specific methods for solving these problems. The design framework is very general in that it can be used to formulate hypotheses that modify both the initial conditions of an experiment and the theory for predicting experimental outcomes. Further, this framework provides a much more goal-oriented view of science than has previous AI research in scientific discovery. I believe this view will lead to more efficient hypothesis generation because the process of hypothesis formation is guided by the dependency structure of a prediction, rather than generating and testing hypotheses more or less at random.

I developed two approximate methods for reasoning about process effects. One involves simulating the effects of a process on typical inputs, and observing what outputs are produced. The second involves assuming that we can achieve a desired change in the output of a process by making the same change to the input of the process, because in this domain processes often copy and modify their inputs to produce their outputs.

Although I developed HYPGENE to solve hypothesis-formation problems in molecular biology, both the design framework and the specific methods discussed herein can be applied to other domains. The methods discussed in section 5 apply to experiments whose initial conditions and outcomes can be expressed as lists of predicate-calculus assertions. They use theories that can be expressed as sets of processes with predicate-calculus preconditions, and whose effects can be formulated as assertion add lists. The design operators described in section 5.3 manipulate the assertions in experiment descriptions, and the conditions and effects in process descriptions. Most design operators are not specific to the domain of molecular biology (the exceptions are the quantitative-hypothesis design operators, which incorporate assumptions about my qualitative biochemistry). Any domain that we can model using the GENSIM framework, such as chemistry and qualitative physics, should be a candidate domain for HYPGENE.

Although this proposition has not been proven, I proposed that HYPGENE's operators are not only general, they are also complete—in two senses. They are syntactically complete because they are capable of making all possible changes allowed within the GENSIM representation language. For example, the process-design operators proposed in Karp (1989) can create any process that can be expressed in GENSIM's process-description language.

The operators are semantically complete in that the hypotheses that they design encompass all types of causal change in this domain. For example, every causal mechanism that could increase the quantity of an object in a experiment is encoded as an operator.

HYPGENE advances the state of the art of hypothesis formation because it has solved two complex scientific problems from the real-world domain of molecular biology (see sections 6.1 and 6.2). The realism of the problems that HYPGENE has solved was documented in Karp's (1989) historical study of attenuation. Each of the two problems involved biological experiments and hypotheses that were worthy of publication in the molecular biology literature, and that contributed to the discovery of an important new mechanism of bacterial gene regulation. HYPGENE produced correct solutions to hypothesis-formation problems involving the repression of the trp operon, matching the hypotheses that were originally formulated by the biologist Hiraga (1969). When applied to a more significant pair of experiments from Jackson and Yanofsky (1973), HYPGENE produced all but one of the hypotheses proposed by Jackson and Yanofsky, and formulated several other hypotheses that Jackson and Yanofsky did not propose. To formulate the hypothesis that it missed, HYPGENE would require knowledge of laboratory techniques that is beyond the scope of this work. The biologists did not propose the extra hypotheses that HYPGENE generated because they possessed quantitative knowledge that HYPGENE did not have.

Acknowledgments

This work benefited greatly from the joint guidance of Bruce Buchanan, Edward Feigenbaum, Peter Friedland, and Charles Yanofsky. Jan Żytkow contributed very valuable comments on two drafts of this article. This work was supported by funding from NSF grant MCS83-10236, NIH grant RR-00785, DRPA contract N00039-83-0136, and by the National Library of Medicine, where the author was a postdoctoral fellow.

Notes

1. The MOLGEN work of Friedland and of Stefik addressed the experiment-planning problem (Friedland, 1979; Stefik, 1980); my work evolved from theirs, but addresses different problems.
2. Unfortunately, scientists use the word *artifact* to refer to interesting experimental observations that were caused by technical errors in an experiment, rather than by the properties of a system under study; I use the word to mean a designed entity.
3. An additional hypothesis-formation example is presented in Karp (1989). The present article corrects errors in the figure references in the trials chapter of Karp (1989).
4. What altering the intrinsic rate of a process means in physical terms depends on the process in question. If the process described is an enzymatic reaction, we could alter its rate by changing the physical properties of the enzyme, such as by introducing mutations in the enzyme.

Appendix

This appendix provides more detailed descriptions of the quantity-hypothesis design operators that were introduced in section 5.3.2.

Figure 10 shows a hypothetical reaction network that will be used as an example in this discussion. Imagine that we wish to generate hypotheses about how to increase the amount of the object G that is present in the hypothetical prediction shown in figure 10.

Figure 11 shows HYPGENE's quantity-hypothesis operators. They have been arranged in classes according to whether they are directed at increasing or decreasing a quantity, and according to which of the three types of influences on a quantity the operator attempts to alter. The operator "Modify Initial Conditions To Increase Quantity" implements the simplest way of increasing the final concentrations of G, namely, to postulate that the initial concentration of G in I_A was higher than was originally thought.

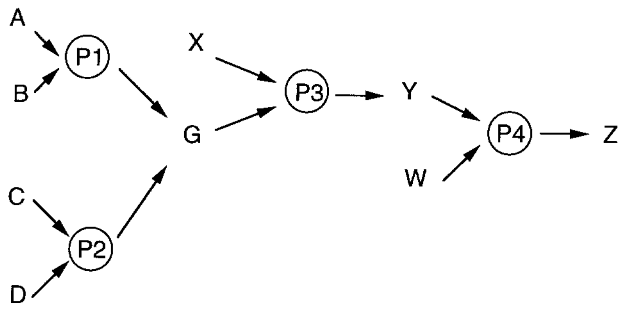

Figure 10. A sample reaction network. Processes are circled and objects are not circled. Here the process $P1$ specifies that A and B react to form G.

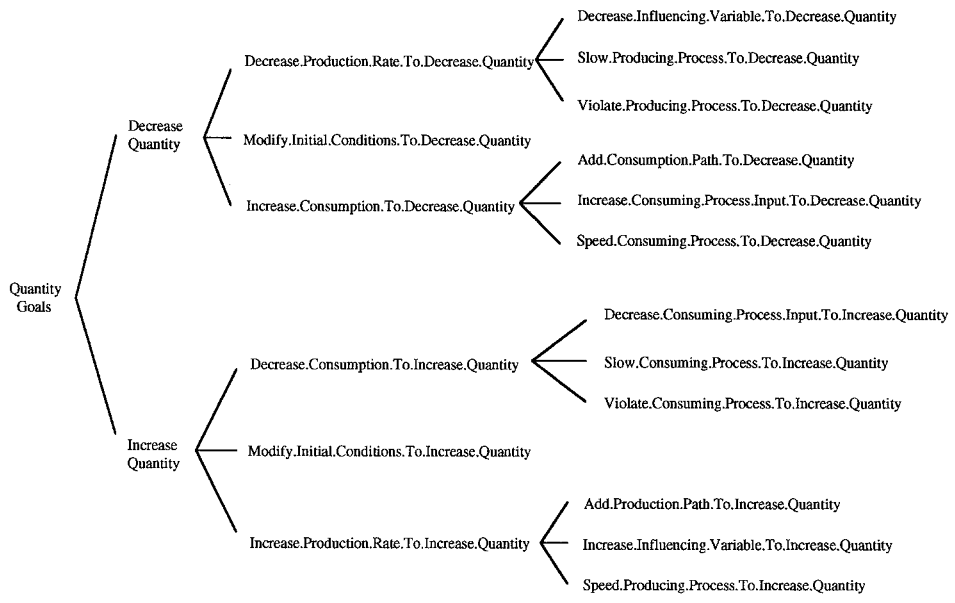
Quantity Goals
Decrease Quantity
Decrease.Production.Rate.To.Decrease.Quantity
Decrease.Influencing.Variable.To.Decrease.Quantity
Slow.Producing.Process.To.Decrease.Quantity
Violate.Producing.Process.To.Decrease.Quantity
Modify.Initial.Conditions.To.Decrease.Quantity
Increase.Consumption.To.Decrease.Quantity
Add.Consumption.Path.To.Decrease.Quantity
Increase.Consuming.Process.Input.To.Decrease.Quantity
Speed.Consuming.Process.To.Decrease.Quantity
Increase Quantity
Decrease.Consumption.To.Increase.Quantity
Decrease.Consuming.Process.Input.To.Increase.Quantity
Slow.Consuming.Process.To.Increase.Quantity
Violate.Consuming.Process.To.Increase.Quantity
Modify.Initial.Conditions.To.Increase.Quantity
Increase.Production.Rate.To.Increase.Quantity
Add.Production.Path.To.Increase.Quantity
Increase.Influencing.Variable.To.Increase.Quantity
Speed.Producing.Process.To.Increase.Quantity

Figure 11. The hierarchy of quantity-hypothesis design operators. The leaves in the tree are executable operators that achieve the goals to their left.

The operator class "Increase Production Rate To Increase Quantity" in figure 11 includes several refinements of the general principle of increasing G by increasing the rate at which processes produce G. The operator "Increase Influencing Variable" alters the rate at which an existing bioreaction R produces G by postulating an increase in the concentration of a reactant in R. As noted above, the rate of a bioreaction is proportional to the concentrations of its precursors (in this example, to $\{A, B, C, D\}$). The next operator, "Speed Producing Process," depends on the fact that every process has an *intrinsic rate* (the rate constant for its bioreaction). Increasing the intrinsic rate of either P_1 or P_2 will increase the production of G^4. The operator "Add Production Path" postulates that G is produced through an additional reaction that is not currently believed to be occurring. Such a reaction may not consume any of $\{A, B, C, D\}$—if it did, then its net effect on G would be unclear.

Members of the operator class "Decrease Consumption To Increase Quantity" in figure 11 are analogous to those in the previous paragraph. We can decrease the consumption of G by existing processes (operator "Decrease Consuming Process Input"), by decreasing the concentration of the objects that G reacts with (such as object X), or by decreasing the intrinsic rates of processes that consume G, such as process P_3 ("Slow Consuming Process"). In addition, the amount of G will be increased if P_3 no longer fires at all, which would occur if X did not exist or if a precondition of P_3 were violated ("Violate Consuming Process").

An important caveat for all of these operators is the fact that G is a parameter object for P_3 does not necessarily imply that P_3 consumes G. For example, enzymes participate in many reactions, but are not consumed by the reactions they catalize. For HYPGENE to know whether a given object is actually consumed by a process, it must either be told this information explicity, or determine it by comparing the chemical composition of the process parameter objects with the objects created by the process. In addition, this analysis cannot be local to a single process, but must be a global analysis that is applied to all objects that result from reactions involving G (in this example, Y and Z are the relevant objects). The processes that are activated by G must consume G, and none of the downstream processes may produce G, if we are to be certain that this network is a net consumer of G. HYPGENE does not currently perform this type of chemical analysis, but it would be straightforward to implement.

In the preceding discussion we considered how to generate hypotheses to account for increased quantities of G; accounting for a decrease in G is analogous, and such hypotheses are generated by operators in the class "Decrease Quantity" in figure 10.

References

Altman, R. (1989). *Exclusion methods for the determination of protein structure from experimental data*. Ph.D. thesis, Stanford University Medical Information Sciences, Stanford, CA.

Bertrand, K., & Yanofsky, C. (1976). Regulation of transcription termination in the leader region of the tryptophan operon of *E. coli* involves tryptophan or its metabolic product. *Journal of Molecular Biology, 103*, 339-349.

Buchanan, B., & Mitchell, T. (1978). Model-directed learning of production rules. In *Pattern-Directed Inference Systems*. New York: Academic Press.

Chapman, D. (1987). Planning for conjunctive goals. *Artificial Intelligence, 32*, 333–377.

Dietterich, T. (1984). *Constraint propagation techniques for theory-driven data interpretation*. Ph.D. thesis, Stanford University, Stanford, CA. Also STAN-CS-84-1030 and HPP Report 84-46.

Dietterich, T., & Bennett, J. (1986). The test incorporation theory of problem-solving (preliminary report). In *Proceedings of the Workshop on Knowledge Compilation*.

Fikes, R., & Nilsson, N. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence, 2*, 189–208.

Forbus, K. (1984). Qualitative process theory (Technical Report TR-789). Cambridge, MA: Massachusetts Institute of Technology AI Laboratory.

Friedland, P. (1979). *Knowledge-based hierarchical planning in molecular genetics*. Ph.D. thesis, Stanford University Computer Science Department, Stanford, CA. Report CS-79-760.

Hiraga, S. (1969). Operator mutants of the tryptophan operon in *E. coli*. *Journal of Molecular Biology, 39*, 159–179.

Jackson, E., & Yanofsky, C. (1973). The region between the operator and first structural gene of the tryptophan operan of *E. coli* may have a regulatory function. *Journal of Molecular Biology, 76*, 89–101.

Karp, P. (1989). *Hypothesis formation and qualitative reasoning in molecular biology*. Ph.D. thesis, Stanford University Computer Science Department, Stanford, CA. Technical reports STAN-CS-89-1263, KSL-89-52.

Karp, P. (1990). Hypothesis formation as design. In *Computational models of discovery and theory formation*. Morgan Kaufmann. See also Stanford University Knowledge Systems Laboratory report KSL-89-11.

Karp, P., & Friedland, P. (1989). Coordinating the use of qualitative and quantitative knowledge in declarative device modeling. In *Artificial Intelligence, Modeling and Simulation*. New York: John Wiley and Sons. See also Stanford University Knowledge Systems Laboratory report KSL-87-09.

Karp, P.D. (in press). A qualitative biochemistry and its application to the regulation of the tryptophan operon. In L. Hunter (Ed.), *Artificial intelligence and molecular biology*. Menlo Park, CA: AAAI Press.

Kulkarni, D. (1988). *The processes of scientific research: The strategy of experimentation*. Ph.D. thesis, Carnegie Mellon University Computer Science Department. CMU School of Computer Science Technical report 88-207.

Langley, P., Simon, H., Bradshaw, G., & Żytkow, J. (1987). *Scientific discovery: Computational explorations of the creative process*. Cambridge, MA: MIT Press.

Lindsay, R., Buchanan, B.G., A., F.E., & Lederberg, J. (1980). *Applications of artificial intelligence for organic chemistry: The DENDRAL project*. New York: McGraw-Hill.

Pednault, E. (1988). Extending conventional planning techniques to handle actions with context-dependent effects. In *Proceedings of the 1988 National Conference on Artificial Intelligence*, (pp. 55–59). Morgan Kaufmann.

Rajamoney, S. (1988). *Explanation-based theory revision: An approach to the problems of incomplete and incorrect theories*. Ph.D. thesis, University of Illinois Department of Computer Science.

Simmons, R. (1988). *Combining associational and causal reasoning to solve interpretation and planning problems*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

Stefik, M. (1980). *Planning with constraints*. Ph.D. thesis, Stanford University Computer Science Department, Stanford, CA. Technical reports HPP-89-2, STAN-CS-80-784.

Tong, C. (1988). *Knowledge-based circuit design*. Ph.D. thesis, Stanford University Computer Science Department, Stanford, CA.

Yanofsky, C. (1989). Personal communication.

Received November 28, 1990
Final Manuscript May 26, 1992