CHAPTER

10

Knowledge-Based Simulation of DNA Metabolism: Prediction of Action and Envisionment of Pathways

Adam R. Galper, Douglas L. Brutlag & David H. Millis

1. Introduction

Our understanding of any process can be measured by the extent to which a simulation we create mimics the real behavior of that process. Deviations of a simulation indicate either limitations or errors in our knowledge. In addition, these observed differences often suggest verifiable experimental hypotheses to extend our knowledge.

The biochemical approach to understanding biological processes is essentially one of simulation. A biochemist typically prepares a cell-free extract that can mediate a well-described physiological process. The extract is then fractionated to purify the components that catalyze individual reactions. Fi-

nally, the physiological process is reconstituted *in vitro*. The success of the biochemical approach is usually measured by how closely the reconstituted process matches physiological observations.

An automated simulation of metabolism can play a role analogous to that of the biochemist in using and extending knowledge. By carefully representing the principles and logic used for reasoning in the laboratory, we can simulate faithfully, on a computer, known biochemical behavior. The simulation can also serve as an interactive modeling tool for reasoning about metabolism in the design of experiments, in discovery, and in education.

1.1 Simulation Methods

Simulation is a modeling technique that represents the behavior of individual components of a system over time. There are two predominant approaches [Rothenberg, 1989]. The *analytic* approach to simulation uses mathematical analysis to represent the temporal behaviors of components, often in closed form. Analytic simulations capture aggregate system behavior by modeling small and relatively similar entities. *Discrete-event*, or discrete-state, simulation is used when the system's overall behavior is not understood well enough to permit formal mathematical analysis; instead, the low-level, pairwise interactions of components are encoded, the simulation is "run," and higher-level patterns of interaction are revealed.

Until recently, most simulations of metabolism were analytic. By metabolism, we mean a set of reactions, the members of which participate in the synthesis (anabolism), degradation (catabolism), or general maintenance of a substance. A typical reaction in a metabolic pathway may involve numerous reactants, intermediates, and products, and may be catalyzed by an enzyme and cofactors. Furthermore, each reaction may be characterized kinetically in terms of metabolite concentrations and reaction rates. The analytic approach to metabolic simulation typically requires the determination of steady-state rate equations for constituent reactions, followed by numerical integration of a set of differential equations describing fluxes in the metabolism [Bierbicher, Eigen, and Gardiner, 1983; Franco and Canela, 1984; Kohn and Garfinkel, 1983a; Kohn and Garfinkel, 1983b; Thomas, *et al*., 1976; Waser, *et al*., 1983].

For example, Franco and Canela present an analytic simulation of purine metabolism, including the salvage pathway and interconversion of purine mononucleotides, using information from the literature about the kinetic behavior of 14 metabolic enzymes [Franco and Canela, 1984]. They then simulate an increase or decrease in the concentration of any enzyme to approximate the metabolic changes observed in inborn errors of purine metabolism.

The feasibility of the analytic approach is limited by the extent to which the metabolic processes of interest have been characterized. For most metabolic pathways, either we are unaware of all the steps involved or we

lack rate constants for each step. This lack of information precludes the use of the mathematical approach in describing the process. Even when reaction rates are known, differential equations incur great computational costs; numerical integration of Franco and Canela's set of 15 differential equations, implemented in FORTRAN 77, required almost 2 hours of CPU time on an IBM 4341 mainframe. Subsequent simulation of enzyme deficit and overproduction required an average of 275 seconds of CPU time per enzyme.

Although the closed-form solutions of analytic simulation are appealing, they are often cryptic and are difficult to use interactively. Differential equations model metabolites along only quantitative dimensions (e.g., concentrations, reaction rates); qualitative knowledge (e.g., structural properties of metabolites or enzymes) is often external to the simulation. If a reaction in a metabolic pathway is only partially characterized, a strict analytic approach to simulation may not work, for lack of quantitative data.

The discrete-event approach to simulation, on the other hand, can use all available data, both quantitative and qualitative, and can even incorporate analytic methods where applicable; semiquantitative models, which couple symbolic and numeric computing techniques, have been developed for a number of domains, including the human cardiovascular system and gene regulation in bacteria [Widman, 1989; Karp and Friedland, 1989; Meyer and Friedland, 1986].

The critical feature of discrete-event simulation is its natural support of qualitative representation and reasoning techniques, which offer explicit treatment of causality. Qualitative representations are thought to provide more insight into how physical systems function [deKleer and Brown, 1984]. The recent flurry of interest in qualitative reasoning has much to offer to both analytic and discrete-event simulations of physical systems [Bobrow, 1984].

Whereas the differential equation is the basic currency of analytic simulation, the rule is central to discrete-event simulation. Rule-based methods allow the representation of knowledge at multiple levels of detail [Buchanan and Shortliffe, 1984; Davis, Buchanan, and Shortliffe, 1977]. For example, in some instances, the actual catalytic mechanism and intermediates of a metabolic reaction are known and can be specified. In other instances, only substrates and products can be represented. Likewise, the regulation of a pathway can be represented at various levels of detail. For example, the feedback inhibition on the transcription process, which controls the overall level of activity of an enzyme, can be expressed in a few simple rules, without the entire process of gene expression being described. Other pathways may require a more detailed representation of all enzymes, activators, and inhibitors.

A rule-based, discrete-event simulation of metabolism can also be fast and highly interactive. Inference is commonly achieved through forward chaining, or deduction from an asserted fact, and through backward chaining, in

which specific facts are inferred to support a hypothesis. Truth-maintenance mechanisms, which deduce and retract conclusions automatically when the underlying fact base changes, make the reasoning processes involved in rule-based simulations more robust and efficient [deKleer, 1986].

Most important, a discrete-event simulation, implemented with rules, can explain its predictions based on the known facts and on the rules relating those facts. Explanation graphs show the flow of logic, the relationships among stated facts, and the deduced conclusions. In comparison, analytic simulations often obscure the understanding being sought.

1.2 A Simulation of DNA Metabolism

We have built a rule-based, discrete-event simulation of DNA metabolism. In particular, we have focused on the pathways of DNA replication and repair in *Escherichia coli* (E. coli). The simulation relies on a panoply of artificial-intelligence (AI) techniques for representation, inference, and explanation; we refer to the simulation as *knowledge-based*. We have chosen initially to represent all domain knowledge qualitatively, because most biochemists reason about DNA metabolism in qualitative terms [Schaffner, 1987].

Unlike intermediary metabolism, in which the flow of substrates and cyclical reactions are critical, DNA metabolism is characterized by discrete, temporally ordered events, in which the concentration of substrate is assumed to be sufficient to support metabolic reactions. For example, when a nucleotide is present, we assume that its concentration is greater than K_m, the substrate concentration at which an enzyme-catalyzed reaction proceeds at half-maximal velocity. Thus, the reactions with which we are concerned either occur or do not occur; there are no partial reactions in our system.

With this commitment, we have little need for the precise quantitative measures that characterize enzyme kinetics. We map all continuous variables, such as substrate concentration, pH value, and temperature, into discrete ranges, in which enzymes either show activity or do not show activity, and we refer to these ranges within rules.

Currently, the simulation can predict the action an enzyme will take under a large number of experimental conditions, and can envision a subset of the possible metabolic pathways followed by substrates. In qualitative reasoning, envisionment is the determination of all possible behavioral sequences from an initial structural description. Ultimately, we hope to envision all possible pathways from an initial description of an experimental situation.

This chapter recounts our experience thus far in developing a knowledge-based simulation of DNA metabolism. Section 2 provides background information for computer scientists and biologists. In Section 3, we present our techniques in detail. Section 4 provides sample interactions with the simulation. Finally, Section 5 compares our techniques to related work on metabol-

ic simulations, summarizes our conclusions, and discusses future research directions.

2. Background

We have begun a formal description, using AI techniques, of the replication and repair pathways of DNA metabolism in *E. coli*. In this section, we present, for computer scientists, a brief description of DNA metabolism, and, for biologists, an introduction to relevant symbolic processing techniques.

2.1. DNA Metabolism

The major mechanisms of DNA metabolism include replication, repair, transcription, and mutation. Genetic information is transferred from parent to progeny by the faithful replication of DNA, in which the nucleotide base sequence of the parent molecule is duplicated. Repair mechanisms preserve the integrity of DNA molecules by correcting occasional replication errors (mismatched base pairs) and eliminating damage caused by the environment (radiation, chemicals). The expression of genetic information begins with the transcription of DNA to RNA. Mutations of DNA molecules, which result in mutant phenotypes, can involve the substitution, addition, or deletion of one or more bases. These metabolic processes are not understood completely, but many of the implicated enzymes have been well characterized. In our simulation, we address the mechanisms of replication and repair in the common intestinal bacterium *E. Coli* by representing current knowledge about the critical enzymes.

DNA polymerase I from *E. coli* is one of the more complex enzymes of DNA metabolism, possessing at least five distinct enzymatic activities in a single polypeptide chain [Kornberg, 1980; Kornberg, 1982]. It is the central player in the major pathways of DNA replication and repair, and is one of the most highly characterized enzymes in DNA metabolism. The enzyme is able to synthesize DNA from the four precursor deoxynucleoside triphosphates—dATP, dGTP, dCTP, and dTTP—as long as a primer-template DNA molecule is present. The enzyme extends the 3'-hydroxyl terminus of a DNA primer, which is hydrogen-bonded to the template, by adding nucleotide residues one at a time, according to the Watson–Crick base-pairing rules—adenine with thymine and guanine with cytosine.

DNA polymerase I occasionally adds a nucleotide that cannot hydrogen-bond to the corresponding base in the template strand. When this happens, polymerization stops, because the primer is no longer correctly hydrogen-bonded. However, DNA polymerase I can remove the unpaired base using an endogenous 3' exonuclease activity and resume polymerization. This 3' exonuclease activity is known as proofreading. DNA polymerase I can also remove base-paired nucleotides from the 5' terminus; when polymerization oc-

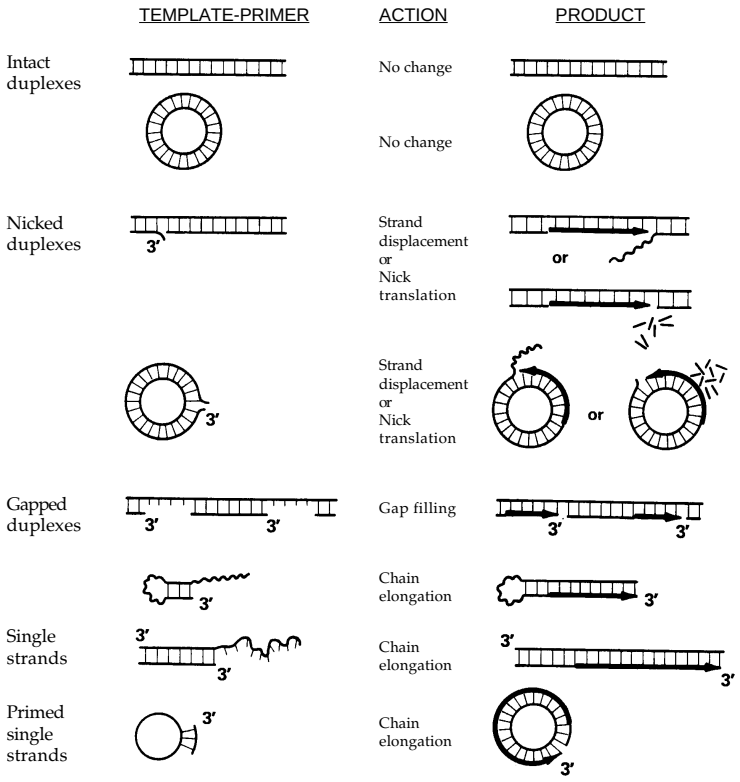

Figure 1. The activities of DNA polymerase I on various templates and primers. (Source: Adapted from Kornberg, 1980, with permission.)

curs simultaneously, nick translation may occur. Polymerization and exonucleolytic degradation are the primary activities of DNA polymerase I, as depicted in Figure 1.

E. coli DNA ligase performs an important function at the end of DNA repair, replication, and recombination—namely, sealing the remaining nicks. DNA ligase joins adjacent 3'-hydroxyl and 5'-phosphoryl termini in nicked duplex DNA by forming a phosphodiester bond. In *E. coli*, DNA ligase requires magnesium and nicotinamide adenine dinucleotide (NAD) as cofactors.

Phosphodiester bond synthesis occurs through three component reactions [Lehman, 1974], as depicted in Figure 2. First, the enzyme reacts with NAD to form ligase-adenylate, a complex in which an adenosine monophosphate (AMP) moiety is linked to a lysine residue of the enzyme through a phosphoamide bond. Nicotinamide mononucleotide (NMN) is released (see Figure

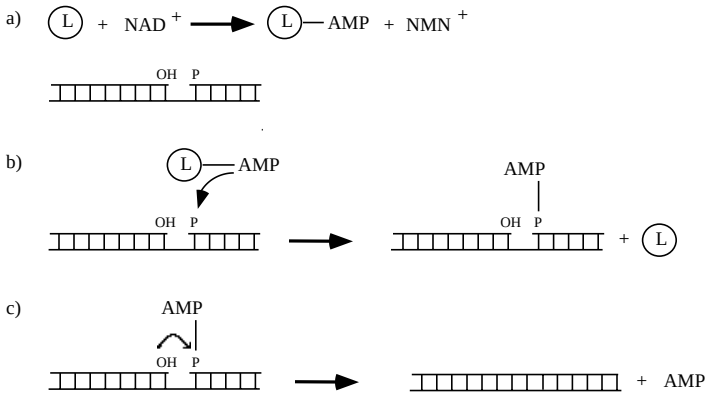

Figure 2. The activities mediated by DNA ligase. a) DNA ligase and nicotinamide adenine dinucleotide combine to form ligase adenylate. b) The adenyl group is transferred from the ligase-adenylate complex to the DNA at the site of the nick to generate a new pyrophosphate linkage. c) The 5' phosphate is attacked by the apposing 3'-hydroxyl group at the nick to form a phosphodiester bond, thus eliminating the AMP.

2a). Next, the adenyl group is transferred from the ligase-adenylate complex to the DNA at the site of the nick to generate a new pyrophosphate linkage, between the AMP group and the 5'-phosphoryl terminus at the nick (see Figure 2b). Finally, the 5' phosphate is attacked by the apposing 3'-hydroxyl group at the nick to form a phosphodiester bond, and AMP is eliminated (see Figure 2c).

Each of these component reactions is reversible; thus, DNA ligase is also able to catalyze an AMP-dependent endonuclease reaction. These nicking and sealing activities can be demonstrated through the AMP-dependent conversion of a closed superhelical circle via a nicked, adenylylated intermediate to a closed, relaxed circle [Modrich, Lehman, and Wang, 1972]. For a complete discussion of DNA metabolism, we refer the reader to any of several textbooks on molecular biology, including [Freifelder, 1985; Watson, 1988], and to reference texts on replication and repair [Kornberg, 1980; Friedberg, 1985].

The catalytic actions mediated by DNA polymerase I and DNA ligase depend on both the physiological conditions and the structure of the DNA substrate. For example, if conditions are not appropriate for binding free nucleotides, then polymerization by DNA polymerase I will not occur. Alternatively, if the 3' primer terminus of the DNA is not a hydroxyl group, then polymerase I will bind either too tightly or too loosely to the substrate, and synthesis of new DNA will be thwarted. If NAD is not present, then

DNA ligase will not seal a nick. Notice that these catalytic actions, as well as all those depicted in Figures 1 and 2, can be expressed succinctly as rules, with the appropriate descriptions of enzyme, substrate, and conditions.

2.2. Artificial Intelligence Methods

Artificial intelligence offers numerous methods for representing large amounts of knowledge and for reasoning with that knowledge to find solutions to problems. We use a common and very general framework known as a *production system*. A production system consists of a set of rules for drawing conclusions and performing actions, a working memory that structures the relevant information appropriately, and a control strategy for governing the use of the rule set on the working memory. Each of our production rules is expressed in an English-like if–then form. For example, to denote the requirement that a 3' terminus be paired and have a hydroxyl group for DNA polymerase I to extend a primer, we write

```
(IF (OR
     (AND (THE EXTERNAL-3P-GROUP OF DNA IS HYDROXYL)
          (THE EXTERNAL-3P-END OF DNA IS PAIRED))
     (AND (THE INTERNAL-3P-GROUP OF DNA IS HYDROXYL)
          (THE INTERNAL-3P-END OF DNA IS PAIRED)))
THEN
DEDUCE
     (A SPECIFICITY OF DNA-POLYMERASE-I IS PRIMER-EXTENSION))
```

We shall explain the representation of DNA and enzyme in Section 3.1; for now, notice that the premise and conclusion of this rule refer to objects in our simulated world (`DNA`, `DNA-POLYMERASE-I`). Each object, or *unit*, has various attributes, or *slots* (e.g., `INTERNAL-3P-END`, `SPECIFICITY`), each of which can take on a number of values (e.g., `PAIRED`, `PRIMER-EXTENSION`). Units correspond to real-world entities, and slots describe those entities; with these tools, we can build the second component of a production system—the working memory on which the rules act.

Unit representations, often referred to as *frame-based*, have several advantages over other representational methods [Fikes and Kehler, 1985; Minsky, 1975; Minsky, 1986; Stefik, 1979]. Frames can be organized into hierarchies, in which the most specific objects, called *instances*, inherit attributes from the more general objects, called *classes*. In addition, hierarchical frame-based representations are *object-oriented* and *modular* [Bobrow and Stefik, 1986; Brachman, Fikes, and Levesque, 1983; Levesque and Brachman, 1984; Stefik and Bobrow, 1986].

The final component of a production system is the control strategy, which is used to apply the production rules to the working memory. To determine the applicability of each rule, the production system can compare the premise

of the rule to working memory; if the premise is true, the rule is "fired," and the actions prescribed by the rule are taken. The control strategy specifies the order in which rules will be compared to working memory and resolves conflicts that arise when several rules match at the same time. Two common control strategies are *breadth-first* and *depth-first* search.

The production-system framework can be used to reason in both the forward and backward directions. In the forward direction, we reason from the data available currently; the premises of rules are matched against working memory, and any actions are taken on the working memory. Then, the rule set is compared to the new working memory. This approach is often called *data-directed reasoning* or *forward chaining*. In the backward direction, we reason from our desired goals; the conclusions of rules are matched against working memory, and the premises become new goals to be achieved. We continue until the initial goal is achieved. This approach is known as *goal-directed reasoning* or *backward chaining*. Of course, the same rule set can be used for both forward and backward chaining.

In addition to a production system, we use a technique known as *truth maintenance*. A truth-maintenance system (TMS) supports *nonmonotonic reasoning*, in which the number of facts known to be true is not strictly increasing over time. Thus, the addition of a new piece of information to working memory may force the deletion of another. A TMS manages the dependencies among facts. A particular fact becomes true when one or more supporting facts becomes true. The same fact may become false during the course of a run through the simulation if new information causes the supporting facts to become false.

For example, a user may assert that DNA has a hydroxyl group at its 3' terminus, which is also paired. The system can conclude that DNA polymerase I could extend the primer from the 3' end, if the environmental conditions were appropriate (e.g., nucleotides are required). If the user now removes the fact that the 3' terminus is paired, the TMS retracts the earlier conclusion about DNA polymerase I's specificity for extending the primer. The TMS is similar to a forward chainer in that both examine facts that are currently true to determine whether new facts can become true. In addition, the TMS can withdraw a fact when there no longer are sufficient data to support that fact.

We refer the reader to any of several AI textbooks for a comprehensive introduction to the field [Nilsson, 1980; Rich, 1983; Charniak and McDermott, 1985; Schapiro, 1986].

3. Techniques

In this work, the domain-specific knowledge has been provided directly by the developers of the system and by readings from the literature

[Kornberg, 1980; Lehman, 1974]. In the future, we plan to simplify the process of knowledge acquisition, so that a biochemist will be able to enter new information without having to learn the details of the knowledge representation. We discuss some possibilities for knowledge-acquisition tools in Section 5.

The simulation currently resides in the Knowledge Engineering Environment (KEE), developed by Intellicorp, Inc. KEE provides a rich collection of knowledge-engineering tools in a Common LISP environment. A flexible and expressive frame system allows the representation of complex objects, relationships, and behaviors. KEE units can be organized logically into hierarchies to permit parsimonious representations. Rules are themselves units, and can be used for both forward and backward chaining. An assumption-based truth-maintenance system [deKleer, 1986] manages the dependencies among facts, as expressed by rules. Facts can thus be concluded automatically whenever existing evidence supports their inference; when justifications are retracted, all dependent facts are retracted as well. KEE's ActiveImages package provides a number of graphic displays for both viewing and modifying attribute values in KEE objects. The KeePictures package will permit us to develop graphic representations of metabolic objects, including intermediates and products.

In Sections 3.1 through 3.4, we distinguish between the representation of simulation objects and the representation of the interactions between these objects.

3.1. Representation of Objects

We have developed a modular and robust representation for the metabolites we wish to simulate. There are currently three major classes of objects: `DNAS`, `ENVIRONMENTAL-CONDITIONS`, and `ENZYMES`. An instance of each class requires specification of all possible attribute values; the rules describing an instance's behavior are described in Section 4.2. There are currently four major objects: `DNA`, an instance of the `DNAS` class; `CONDITIONS`, an instance of the `ENVIRONMENTAL-CONDITIONS` class; and two `ENZYMES` instances, `DNA-POLYMERASE-I` and `DNA-LIGASE`. All information regarding `DNA`–including class information, instances, active images, and rules–is contained in the `DNA-KB` knowledge base. Likewise, all information regarding the environmental conditions is contained in the `CONDITIONS-KB`. All information regarding each enzyme instance is contained in the dedicated knowledge bases, `POL-I-KB` and `LIGASE-KB`. This design allows us to load units for selective testing, without the interference of knowledge from other objects in the simulation. This use of distributed knowledge bases also will eventually allow us to examine specific subsets of enzymes, much as a biochemist would mix reagents in the laboratory.

The descriptive information in the `DNAS` class is intentionally redundant.

Our goal is to provide methods for specifying the properties of DNA in as many ways as is natural for a scientist. For example, the biochemist can declare that the STRUCTURE of DNA is a NICKED-CIRCLE, or that the TOPOLOGY is CIRCULAR and the STRANDS are NICKED-DUPLEX. Either description will infer the other. The rules for reasoning about DNA are instances of the DNA-RULES class and refer only to attributes of a DNA unit. All 96 DNA-RULES instances are organized by the attribute of DNA referenced in the conclusion of the rule; thus, all rules that determine the TOPOLOGY of DNA are members of the TOPOLOGY-RULES subclass of DNA-RULES.

We use a hierarchy of four levels to describe DNA. At the lowest level, we describe a DNA molecule by characterizing the 5' and 3' termini at both external and internal positions. For example, a gapped, linear DNA molecule will have 3'-internal and 5'-internal termini at either end of the gap; in addition, there are 3' and 5' external termini at the ends of the molecule. We characterize each terminus by specifying the chemical group present (e.g., HYDROXYL, PHOSPHATE, DIDEOXY, ADENYL) and the nature of the terminus (e.g., PAIRED, UNPAIRED, RECESSED, PROTRUDING). At the next level, we summarize the information about the termini by filling the ENDS slot with values such as FLUSH and 3'-PROTRUDING. These values can be specified by the user or inferred by rules that consider the status of the component termini. At the next level, the user can fill slots that specify components of the overall structure of the molecule: The NICKS slot qualitatively describes the nicks present (NONE, SOME, ONE, MULTIPLE), the TOPOLOGY slot specifies the possible shapes (e.g., LINEAR, Y-FORM, CIRCULAR), the STRANDEDNESS slot can take on the values SINGLE-STRANDED and DOUBLE-STRANDED, and the STRANDS slot describes the strands independent of topology (e.g., INTACT-DUPLEX, NICKED-DUPLEX, PRIMED-SINGLE-STRAND). Finally, at the highest descriptive level, the overall STRUCTURE slot offers a list of common DNA structures (e.g., PRIMED-CIRCLE, NICKED-LINEAR, COVALENTLY-CLOSED-CIRCLE), from which the value of component slots can be inferred. The active image associated with the DNA unit is shown in Figure 3. We can conceive of multiple, independent DNA units in a simulation; if a reaction causes the generation of a new, independent DNA molecule (e.g., strand displacement followed by cleavage of the displaced strand), the simulation will contain two DNA instances, a duplex and a single-stranded molecule, each of which will interact differently with the enzymes present.

The CONDITIONS unit contains three quantitative attributes that describe the physical environment: TEMPERATURE, PH, and IONIC-STRENGTH (see Figure 4). Values of these slots have been mapped into discrete ranges to facilitate purely qualitative reasoning and to reduce the number of rules that cannot be handled by the TMS (see Section 4.3). Currently, the TEMPERA-

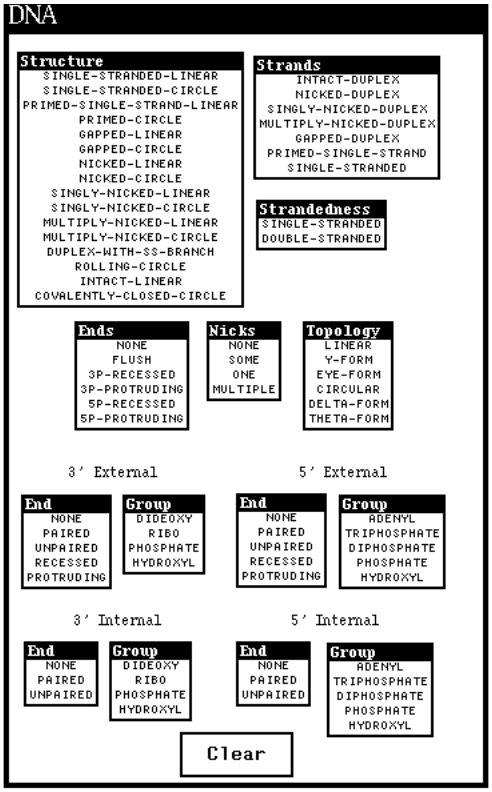

Figure 3. Display of the DNA unit. DNA can be described at several levels of detail. At the most detailed level, DNA can be characterized by the 5' and 3' termini at both external and internal positions; at the most abstract level, the substrate DNA can be one of 16 common structures. The goal is to provide methods for specifying the properties of DNA in as many ways as is natural for a scientist.

TURE-RANGE slot can take on values from among `0-TO-5`, `5-TO-20`, `20-TO-45`, `30-TO-37`, and `45-TO-100`.

The significant `PH-RANGE` values are `6.0-TO-9.5` and `7.5-TO-8.0`; the `PH-RANGE` slot value is unknown if the `PH` slot value is not within these ranges. The `IONIC-STRENGTH-RANGE` slot is handled in a similar fashion, with range values `0.001-TO-0.003` and `0.001-TO-0.3`.

The `CONDITIONS-RULES` class manages the mapping of all quantitative variables into qualitative ranges, which are then referenced in the premises of rules that represent interactions between enzymes, substrates, and the environment. The other attributes in the `CONDITIONS` unit, including `NUCLEOTIDES`, `MONOVALENT-CATIONS`, `DIVALENT-CATIONS`,

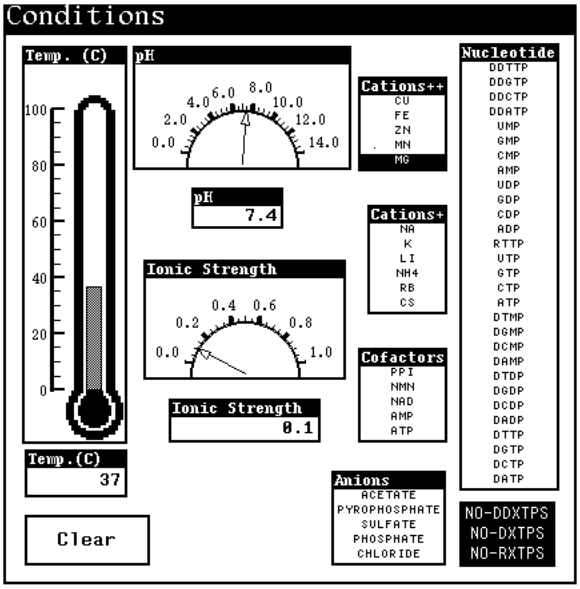

Figure 4. Display of the CONDITIONS unit. The quantitative attributes are mapped into range attributes (not shown). For example, when the TEMPERATURE is 37.5 degrees, the TEMPERATURE-RANGE attribute is 20-TO-45. All enzyme-activity rules that depend on temperature use this attribute to determine temperature.

ANIONS, and COFACTORS, represent physical objects, and could be modeled as units in the simulation. We have chosen not to do this, because we are interested in these objects only by virtue of their presence or absence; we have no use for structural descriptions of these objects. We thus consider these substances as attributes of the environment and assume that they are present in quantities that support the reactions simulated, if they are present at all.

We propose a general model for the qualitative representation of enzymes, embodied in the ENZYMES class. The ACTIVITY of an enzyme is determined by the environmental conditions; likewise, the SPECIFICITY of an enzyme depends solely on the substrate. In turn, the ACTION of the enzyme depends on the enzyme's specificity and activity. In many cases, an enzyme may exist in different STATEs—for example, free or bound to a substrate. The DNA-POLYMERASE-I and DNA-LIGASE units contain different lists of potential values for each of the ACTIVITY, SPECIFICITY, ACTION, and STATE slots. For example, DNA polymerase I can display binding activities (e.g., XMP-BINDING, XTP-BINDING, DNA-BINDING), synthetic activities (DIDEOXY-CHAIN-TERMINATION, STRAND-DISPLACE-

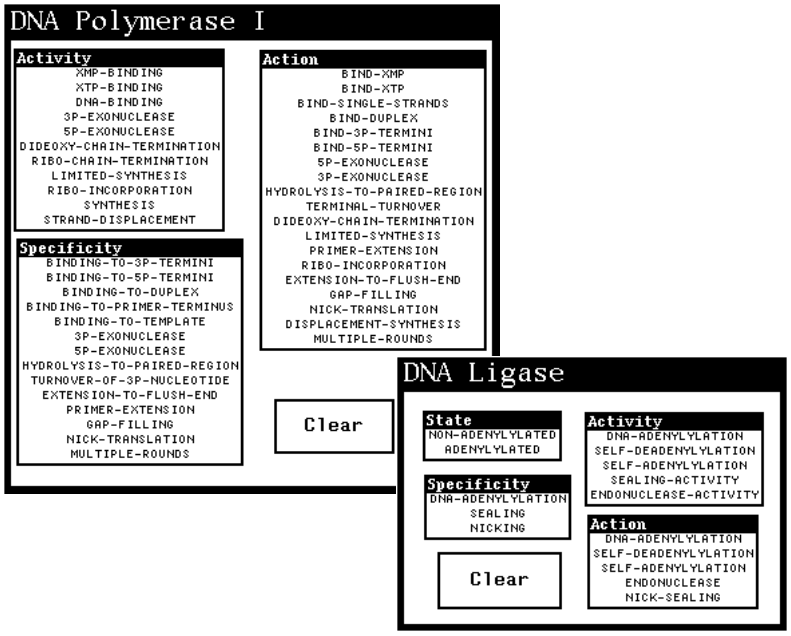

Figure 5. The DNA-POLYMERASE-I and DNA-LIGASE representations. Each subpanel represents an enzyme attribute and contains all possible values of that attribute.

MENT), or degradative activities (3P-EXONUCLEASE, 5P-EXONUCLEASE). Similarly, ligase can bind (DNA-ADENYLYLATION, SELF-ADENYLYLATION), synthesize (SEALING-ACTIVITY), or degrade (ENDONUCLEASE-ACTIVITY). Recall that the ACTIVITY value depends solely on the environmental conditions; the SPECIFICITY slot for each enzyme has similar types of values, but depends on the substrate description. Slots describing an enzyme can take on multiple values at the same time; for example, in nick translation, the polymerization and 5' exonuclease activities of polymerase I are possible simultaneously. The active images for DNA-POLYMERASE-I and DNA-LIGASE, depicting all possible values for each slot, are displayed in Figure 5.

3.2. Representation of Interactions and Behaviors

The object representations described in Section 3.1 correspond to the working memory of a production system; the rule set, which operates on working memory, captures knowledge of the potential interactions between and behaviors of the simulation objects. Rules for simulating DNA-POLYMERASE-I action are all instances of the DNA-POL-I-RULES class, contained in the POL-I-KB knowledge base. The LIGASE-RULE class is like-

wise independently contained in the LIGASE-KB knowledge base.

We structure enzyme rule hierarchies along the same lines as the representation of the enzyme; for example, there are POL-I-SPECIFICITY-RULES, POL-I-ACTIVITY-RULES, and POL-I-ACTION-RULES subclasses of the DNA-POL-I-RULES class. A typical instance of POL-I-ACTIVITY-RULES, describing the effect of the environment on the activity of an enzyme, is

```
(IF (OR   (A TEMPERATURE-RANGE OF CONDITIONS IS 0-TO-5)
          (A TEMPERATURE-RANGE OF CONDITIONS IS 5-TO-20)
          (A TEMPERATURE-RANGE OF CONDITIONS IS 20-TO-45))
     (A IONIC-STRENGTH-RANGE OF CONDITIONS IS .001-TO-.3)
     (A PH-RANGE OF CONDITIONS IS 6.0-TO-9.5)
THEN
DEDUCE
     (AN ACTIVITY OF DNA-POLYMERASE-I IS DNA-BINDING))
```

Another POL-I-ACTIVITY-RULES instance may reference previously deduced activities, in addition to other slots of the CONDITIONS unit.

To predict the action an enzyme mediates, we combine knowledge about the specificity and activity of the enzyme. If the ACTIVITY of DNA-POLYMERASE-I is DNA-BINDING, but there is no DNA present, then we cannot predict that DNA polymerase I actually will bind. A POL-I-SPECIFICITY-RULES instance asserts the readiness of the DNA substrate for action by an enzyme; an example can be found in Section 2.2. A POL-I-ACTION-RULES example follows:

```
(IF (AN ACTIVITY OF DNA-POLYMERASE-I IS SYNTHESIS)
    (A SPECIFICITY OF DNA-POLYMERASE-I IS PRIMER-EXTENSION)
THEN
DEDUCE
    (AN ACTION OF DNA-POLYMERASE-I IS PRIMER-EXTENSION))
```

Most rules for predicting enzyme action are fairly simple. However, there may be 15 to 20 underlying facts necessary to infer the required specificity and activity of the enzyme.

Prediction of enzyme action is only the first step in metabolic simulation; we also want to predict a sequence of different reactions that the enzymes may mediate as the substrate is altered by the actions of the enzyme. The KEEworlds facility is used to this end. The KEEworlds facility allows us to represent steps in a metabolic pathway as changes in the substrate or enzyme; rules can define new worlds (steps in a pathway) in which all information about metabolic objects is inherited from a parent world and only changes to these objects are stored explicitly in the child world. The KEEworlds facility is tightly coupled to the TMS; the TMS is used to predict

what actions the enzyme would take in the altered environment. Multiple worlds can be linked together in a highly branched fashion typical of known pathways of DNA metabolism.

When an enzyme action is predicted, the simulation creates a new world in which the structure of the substrate DNA in the original world is modified by the enzyme's action; the new world inherits all information about the DNA from the original world, but modifies slot values accordingly. For example, if the ACTION of DNA-POLYMERASE-I is NICK-TRANSLATION on a nicked-linear structure, in a new world, the substrate DNA will now be an intact, duplex molecule. In addition, the enzyme structure may change as a result of its action. In the preceding example, the enzyme may begin bound to the nick; we describe this situation by filling the STATE slot of DNA-POLYMERASE-I with the value NICK-BOUND. In the new world, there is no longer a nick, and the enzyme is bound to a flush end.

Rules that generate new worlds are called new-world-action rules in KEE. The new-world-action rule for the example in the previous paragraph is

```
(IF  (THE STRUCTURE OF DNA IS NICKED-LINEAR)
     (THE STATE OF DNA-POLYMERASE-I IS NICK-BOUND)
     (THE ACTION OF DNA-POLYMERASE-I IS NICK-TRANSLATION)
     (THE INTERNAL-5P-ENDS OF DNA ARE ?Z)
THEN
IN.NEW.AND.WORLD
     (CHANGE.TO (THE STRUCTURE OF DNA IS INTACT-LINEAR))
     (DELETE (THE INTERNAL-3P-GROUP OF DNA IS HYDROXYL))
     (DELETE (THE INTERNAL-3P-ENDS OF DNA ARE PAIRED))
     (DELETE (THE INTERNAL-5P-ENDS OF DNA IS ?Z))
     (CHANGE.TO (THE STATE OF DNA-POLYMERASE-I IS FLUSH-BOUND))
```

This rule represents the *process* of nick translation in a nicked-linear molecule. The generated world has modified the DNA molecule—there are no longer internal 3' or 5' termini—and has changed the state of the enzyme. Nick translation is actually a process composed of similar, repeated steps; we lump these steps into one for this process. Other processes may require a finer granularity of representation.

3.3. Inference

We use an assortment of inference techniques in our simulation. The prediction of enzyme action involves a combination of forward chaining, backward chaining, and truth maintenance. In addition, the simulation of steps in a metabolic pathway requires forward chaining on new-world-action rules. Each of these rules generates a new world in a pathway, asserts new facts, and possibly retracts existing facts; the TMS then predicts enzyme actions in the newly generated world. Next, the new-world-action rules are fired in the

new world, and the process repeats, until no new worlds can be generated.

Within each world, the TMS distinguishes between two types of facts. Primitive facts are added directly to working memory by the user. These facts do not depend on the truth of any other fact. The truth of deduced facts depends entirely on the truth of one or more other facts. Thus, only deduced facts can lose their support and be withdrawn by the TMS during a simulation. Primitive facts can be withdrawn by only the user.

The operation of the TMS is analogous to the activity of readjusting our belief in certain propositions based on a set containing contradictory evidence. Facts can become true through a cascading of evidence in which the consequent of one justification serves as one of the antecedents of another. If the facts asserted by the user lead to a contradiction, this contradiction is displayed to the user in a special window called a *worlds browser* and in the KEE message window. A menu item provides a complete explanation of the origin of the contradiction in terms of both the competing facts and the conclusions derived from those facts (see Section 3.4).

In our simulation, users can assert or retract facts via the graphical interface, or programmatically via a LISP expression. Using the mouse to point to a fact will assert that fact if it is unknown, or will retract that fact if it is known. Known facts are highlighted in inverse video. After a new primitive fact is asserted or retracted, the TMS adds facts that can now be deduced, and removes any deduced facts that are no longer true. All user-initiated assertions take place in the *background* world, from which all new worlds are spawned.

KEE restricts TMS justifications to purely monotonic rules; these rules are called *deduction* rules. For example, the following rule is monotonic; facts are added to only the current environment:

```
(IF  (OR   (THE EXTERNAL-5'-END OF DNA IS PAIRED)
           (THE INTERNAL-5'-END OF DNA IS PAIRED))
THEN
DEDUCE
     (A SPECIFICITY OF DNA-POLYMERASE-I IS 5'-EXONUCLEASE))
```

Assume, however, that we want to retract a fact explicitly, as in the following rule:

```
(IF  (AN ACTIVITY OF DNA-POLYMERASE-I IS DNA-BINDING)
     (OR   (A DIVALENT-CATIONS OF CONDITIONS IS MG)
           (A DIVALENT-CATIONS OF CONDITIONS IS MN))
     (A NUCLEOTIDES OF CONDITIONS IS DATP)
     (A NUCLEOTIDES OF CONDITIONS IS DTTP)
     (A NUCLEOTIDES OF CONDITIONS IS DGTP)
     (A NUCLEOTIDES OF CONDITIONS IS DCTP)
```

```
     (A NUCLEOTIDE-RANGE OF CONDITIONS IS NO-DDXTPS)
THEN
DO
     (AN ACTIVITY OF DNA-POLYMERASE-I IS SYNTHESIS)
     (DELETE (AN ACTIVITY OF DNA-POLYMERASE-I IS LIMITED-SYNTHE-
              SIS))))
```

KEE cannot generate justifications for this rule, because the rule expresses explicit nonmonotonic reasoning. Likewise, rules with certain operators as premises, including LISP expressions, do not generate TMS justifications. These rules are expressed as *same-world-action* rules within KEE; we also refer to these same-world-action rules as *non-TMS* rules.

Since justifications are not generated for non-TMS rules, these rules are not invoked automatically when their premises become true, whereas TMS rules are. In addition, non-TMS rules cannot be included in explanation graphs. We group all non-TMS rules into a single class, and forward chain on this class whenever the value of a unit referenced in the antecedent of a non-TMS rule changes. Special functions called demons (or KEE *active values*) are attached to the attributes mentioned in the antecedents of non-TMS rules. These demons permit nonmonotonic reasoning with non-TMS rules.

To accommodate both standard production rules and the TMS representation of the same knowledge, we have modified the KEE rule parser. Whenever a new rule is entered into the knowledge system (whether via the standard user interface or via the KEE rule editor), the rule is parsed by KEE and the type, the premises, and the conclusions of the rule are determined. We have added a further rule-parsing function to the normal KEE rule parser that examines the rule type. If the rule is nonmonotonic, then the rule unit is added to the non-TMS rules class so that it will be invoked automatically whenever one of its premises changes, as described. If the rule is monotonic, then its premises are asserted into the TMS as justifiers for the conclusions of the rule. Thus, all monotonic rules have a double representation in the knowledge system. The fact that KEE is itself implemented as a series of knowledge bases allows us to modify KEE's action and to change its behavior.

3.4. Explanation

Our knowledge system has a mechanism by which it can explain its predictions for each step of a pathway. For any fact deduced by the TMS, an explanation graph displays the sequence of TMS justifications that were used to derive that fact. In Figure 6, the simulation has ascertained that, given the current state of the substrate, DNA polymerase I could translate a nick. If asked to explain this fact, the TMS would construct the explanation graph shown, based on the currently justifiable facts.

The explanation graph displays the following information. The user has stated that the DNA has some nicks and that it has an internal 3'-OH group.

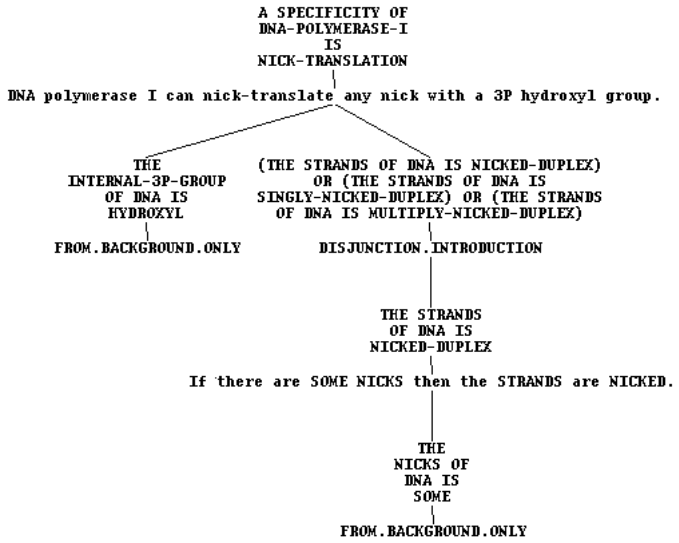

Figure 6. An explanation graph depicting why (A SPECIFICITY OF DNA-POLYMERASE-I IS NICK-TRANSLATION). The explanation graph uses TMS justifications to explain the system's reasoning.

These facts are labeled "FROM.BACKGROUND.ONLY." The TMS has invoked a rule by which it is able to deduce that any DNA with some nicks must be a nicked duplex. The firing of another rule allows the TMS to deduce, from the presence of the internal 3'-OH group in a nicked duplex DNA, that DNA polymerase I could perform nick translation on this DNA molecule.

The user can also ask the system about facts that have not been determined to be true by using the QUERY function. This function invokes the backward chainer and engages in a brief dialogue with the user, searching through the set of rules for ways to establish the given fact and asking the user for additional information that could serve to support this fact.

4. Sample Interactions

There are two modes of interaction with the simulation: prediction and envisionment. In the prediction mode, the user asserts known facts about an experimental system, by describing the DNA and environmental conditions via the corresponding active images; the TMS will conclude other facts automatically. The user can also reason backward from a desired enzyme action. In the envisionment mode, the user chooses to generate all possible metabolic pathways from initial conditions. We present brief examples of each mode, using DNA polymerase I in isolation.

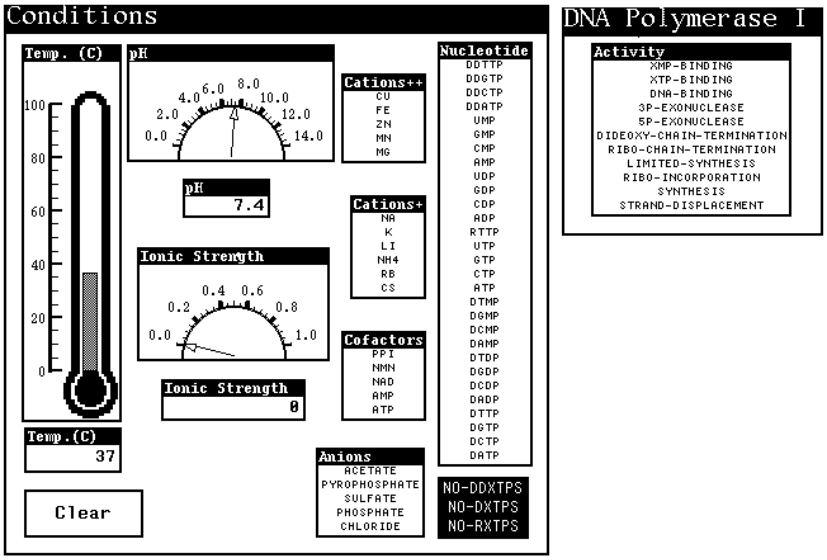

Figure 7. An initial experimental environment. The temperature is 37 degrees Celsius and the pH value is 7.4. No DNA polymerase I activity is possible.

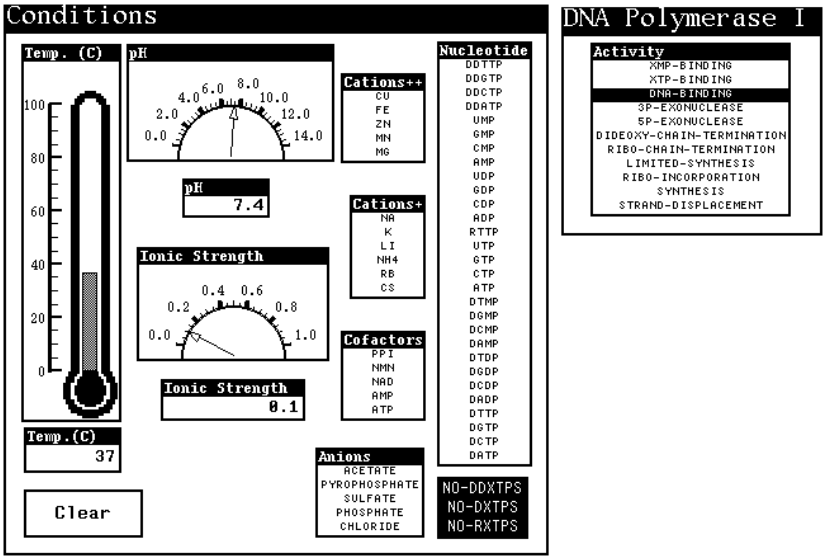

Figure 8. An increase in the ionic strength. DNA polymerase I is now able to bind to DNA. The display for the ACTIVITY of DNA-POLYMERASE-I now shows DNA-BINDING.

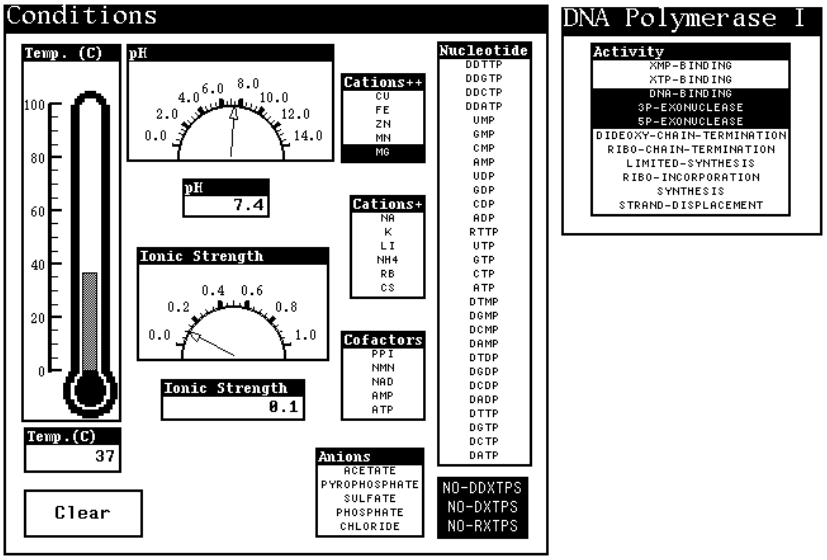

Figure 9. The addition of Mg++. The divalent cation Mg++ is required for exonuclease activities. 3P-EXONUCLEASE and 5P-EXONUCLEASE activities now appear in the ACTIVITY slot.

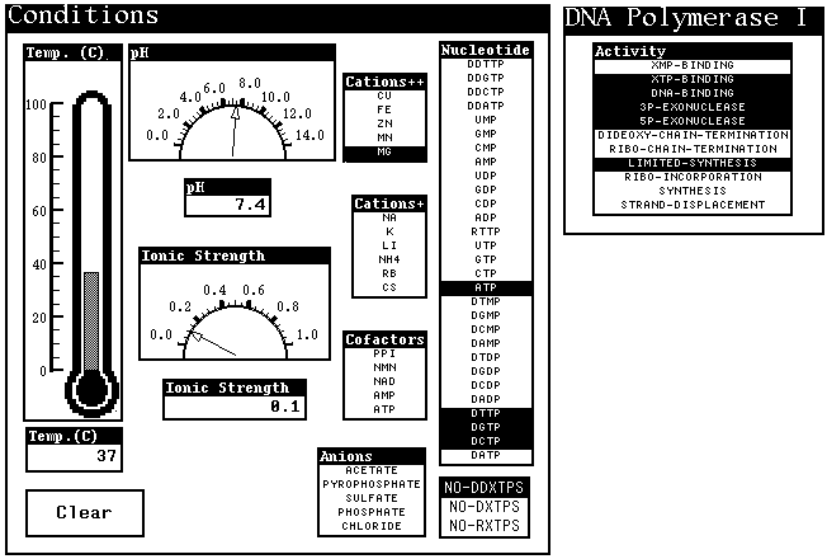

Figure 10. The addition of nucleotides. With the introduction of four nucleotides — ribo ATP, dTTP, dGTP, and dCTP — DNA shows limited synthetic activity due to the lack of dATP.

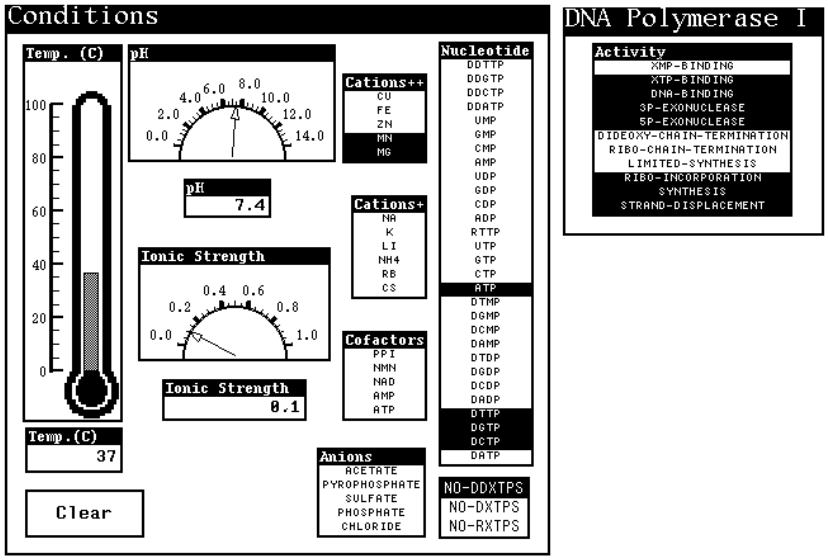

Figure 11. The addition of Mn++. Ribo ATP is incorporated into a growing strand, in the presence of Mn++. Three new activities are displayed: RIBO-INCORPORATION, SYNTHESIS, and STRAND-DISPLACEMENT.

4.1. Prediction of Enzyme Action

We begin with an experimental environment at 37 degrees Celsius and a pH value of 7.4 (Figure 7); notice that DNA polymerase I displays no activity. If we increase the ionic strength (Figure 8), polymerase I is able to bind to DNA. With the subsequent addition of the divalent cation Mg^{++} (Figure 9), DNA polymerase I now shows 3' and 5' exonuclease activities. In Figure 10, we add four nucleoside triphosphates—ribo ATP, dTTP, dGTP, and dCTP. Notice that DNA polymerase I can now bind to these triphosphates and incorporate some of them; the limited synthetic activity is due to the lack of dATP. In the presence of Mn^{++} (Figure 11), however, DNA polymerase I can incorporate ribo ATP into a growing strand; the activities `RIBO-INCORPORATION`, `SYNTHESIS`, and `STRAND-DISPLACEMENT` can now be concluded.

In Figure 12, we assert the presence of a `GAPPED-LINEAR` molecule, with paired 3' and 5' internal ends and a hydroxyl group at the 3'-internal terminus. The `SPECIFICITY` slot of `DNA-POLYMERASE-I` now indicates that DNA polymerase I can bind to three locations on the substrate (the 3' termini, the 5' termini, and the primer terminus), hydrolyze the molecule from either a 3' or a 5' terminus, or extend the primer, and, in doing so, fill the gap. In Figure 13, the simulation predicts seven actions for DNA poly-

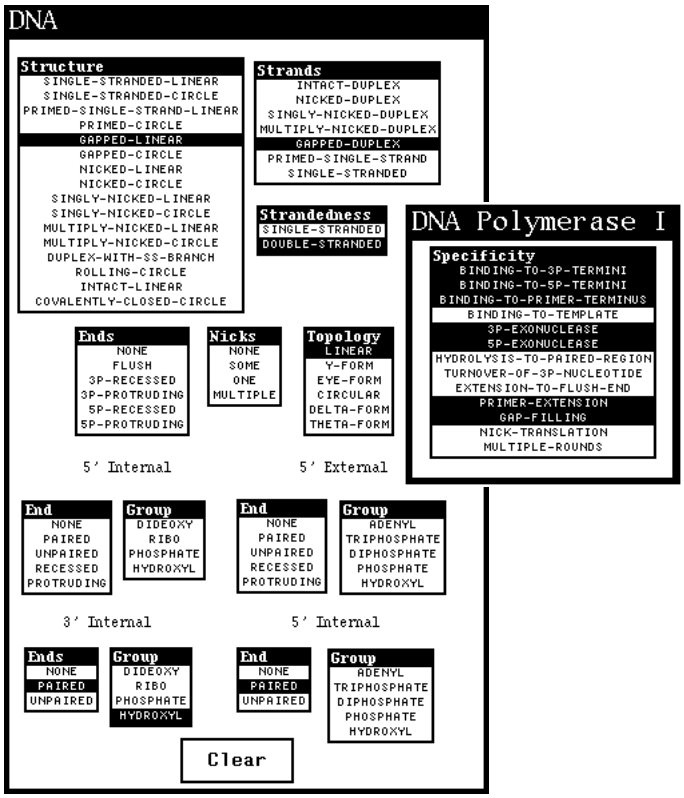

Figure 12. A description of DNA and the specificities of DNA polymerase I. A GAPPED-LINEAR structure is asserted; from this fact, the system concludes that the STRANDS are GAPPED-DUPLEX, the STRANDEDNESS is DOUBLE-STRANDED, the TOPOLOGY is LINEAR, and the 3' and 5' internal ENDs are PAIRED. The SPECIFICITY slot of DNA-POLYMERASE-I now indicates that DNA polymerase I can bind to three locations on the substrate (BINDING-TO-3P-TERMINI, BINDING-TO-5P-TERMINI, and BINDING-TO-PRIMER-TERMINUS), hydrolyze the molecule from either a 3' or a 5' terminus (3P-EXONUCLEASE and 5P-EXONUCLEASE), extend the primer (PRIMER-EXTENSION), and fill the gap (GAP-FILLING).

merase I, each of which can be explained graphically using TMS justifications. Other examples of the use of the system to predict enzyme action have been published elsewhere [Brutlag, 1988].

4.2. Envisionment of Metabolic Pathways

Figure 14 depicts a partial envisionment of the metabolic pathways that

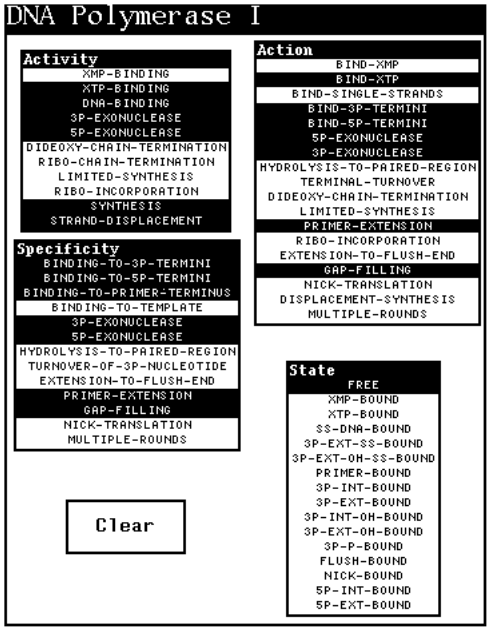

Figure 13. The predictions of ACTIVITY, SPECIFICITY, and ACTION for DNA-POLYMERASE-I.

originate with the gapped linear DNA molecule described in Section 4.1. KEE generates a graph of worlds; we have enhanced this graph to diagramatically depict changes in the structure and states of objects. In this example, DNA polymerase I is the only enzyme present. Each world is named for the action most recently taken by the enzyme; if the enzyme is free, the world is named after the structure of the DNA present in that world.

The system generates four new worlds based on the predicted actions of DNA polymerase I, as described in Section 4.1. Each of these worlds is the result of a binding process. In the first world (W_1), the enzyme is bound to the 3'-internal, or primer, terminus. Here, primer extension, or gap filling, is a predicted action; thus, a new world is generated in which the primer is extended, until the gap has become a nick. In this world, DNA polymerase I can nick translate; the system generates another new world in which the enzyme is now bound to the 3'-external terminus of an intact linear molecule. The only action possible in this world is the dissociation of DNA polymerase I from the substrate; a new world is generated to describe the result of this process. Finally, as depicted in the final world of this pathway, the free enzyme can bind to free nucleotides, but no further activity is observed.

The second world (W_2) contains the enzyme bound to the external 5' terminus of the gapped molecule. In this world, the enzyme can hydrolyze the

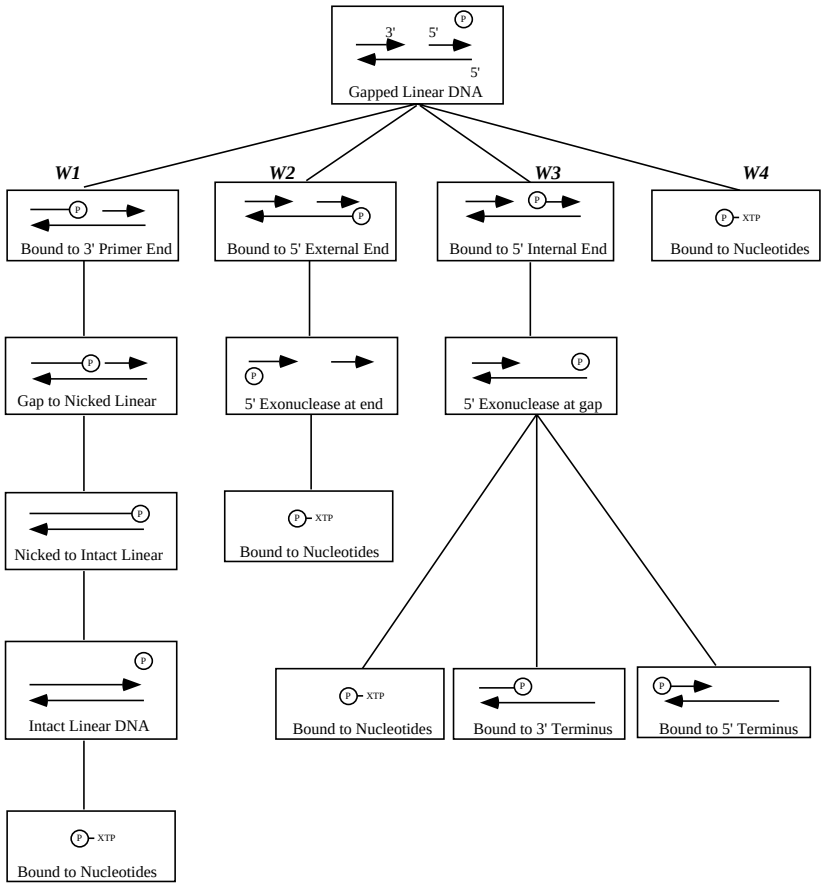

Figure 14. A partial envisionment of the metabolic pathways mediated by DNA polymerase I, beginning with a gapped, linear DNA molecule.

DNA from the external 5' position. The system generates a new world, corresponding to the result of exonuclease activity at the external end. In the world labelled, "5' exonuclease at end," DNA polymerase I has hydrolyzed the intact single strand of the substrate; as a result, the `STATE` of `DNA-POLYMERASE-I` has changed from `5P-EXT-BOUND` to `FREE`, and the `STRUCTURE` of `DNA` has changed from `GAPPED-LINEAR` to `SINGLE-STRANDED-LINEAR`. Since the substrate no longer supports any enzymatic actions, the enzyme binds to free nucleotides in a new world, and the pathway is terminated.

In the third world (W_3), the enzyme is bound to the internal 5' terminus and 5' exonuclease is a predicated action. In the world labelled, "5' exonuclease at gap," the enzyme has hydrolyzed a segment of the gapped strand of

the substrate; as a result, the STATE of DNA-POLYMERASE-I has changed from 5P-INT-BOUND to FREE, and the STRUCTURE of DNA has changed from GAPPED-LINEAR to PRIMED-LINEAR. The enzyme can now bind the primer and the 5' end of the molecule, as well as the free nucleotides present in the environment. New worlds are generated for each of these possibilities. Although not shown, the simulation continues with the extension of the primer in one pathway and the hydrolysis of the primer in another.

The final world (W_4) contains a situation encountered previously in the pathways originating From W_1,W_2, and W_3—namely, the enzyme has bound free nucleotides, and no further activity is observed.

5. Discussion

In recent years, several researchers have developed qualitative models of metabolic processes [Weld, 1986; Karp, this volume; Mavrovonioutis, this volume]. Weld's PEPTIDE system serves as the basis for his theory of aggregation, which detects repeating cycles of processes and creates a continuous process description of the cycle's behavior. We avoid many of the problems his theory addresses by representing continuous processes explicitly. For example, we do not model the polymerization process as a sequence of steps, each of which adds a nucleotide to a growing strand of DNA; instead, polymerization extends a primer as far as possible in one step, and then stops. Likewise, the exonuclease activities of DNA polymerase I are represented not as a sequence of discrete processes, but rather as one continuous process. Binding processes are modeled in a stepwise fashion; in discrete steps, polymerase I may bind to the end of a DNA substrate, then bind to a group at the end of the substrate, and then catalyze a polymerization or hydrolysis reaction. If an event can interrupt a continuous process, with qualitatively similar results at any point in time, we anticipate the event before the process has begun and generate two new worlds: one for the continuous process and one for the interrupted, continuous process. However, we do foresee that we will need aggregation and cycle detection techniques in the near future; the lack of cycle detection currently limits the envisionment capabilities of our system to those pathways that have no reversible reactions.

Karp's model of the regulation of the trp operon serves as the basis for a hypothesis-formation program, which can reproduce the discoveries made by biologists studying gene regulation over a 20-year period. Karp represents biochemical objects with KEE frames; these objects correspond to homogeneous populations of molecules. In our system, instances of the DNAS and ENZYMES classes represent single molecules. To illustrate the implications, suppose a DNA molecule has several spatially separated locations at which an enzyme can bind; in a real experiment, one DNA polymerase I molecule may bind at an internal nick, while another may bind at the external end of

the same DNA molecule. Since we represent only a single enzyme molecule, our simulation generates pathways for only one enzyme action at a time. Thus, our system cannot envision all the pathways that may occur under experimental conditions; simultaneous combinations of actions may produce paths that are missing from our system. For this reason, we refer to our pathway generation technique as *partial* envisionment.

Karp's reactions are stored in KEE frames, and are organized into process knowledge bases. These processes are arranged in an inheritance hierarchy and can inherit attributes from more general reaction classes. Karp constructed a process interpreter to detect and permit interactions between objects. We store our reactions using KEE rule units, and use standard KEE reasoning mechanisms (forward chaining and truth maintenance) to generate pathways. Currently, none of our reactions dynamically generates simulation objects distinct from the original substrate; our new-world-action rules simply change the character of the original DNA molecule. Karp's system dynamically instantiates new objects; we foresee the need for a similar function, as we augment our representation of enzyme functions and add more enzymes to the system.

The goals of our system are different from those of Weld, Karp, and Mavrovonioutis. PEPTIDE's domain of DNA transcription was meant to test Weld's process aggregation methods; Karp's system contains a comprehensive model of bacterial gene regulation, but was developed to test theories of scientific discovery. Mavrovonioutis has developed a system for the computer-aided design of biochemical pathways. Our simulation has been developed for use by biochemists as an interactive reasoning tool. We believe that our representation of enzymes, substrates, and processes is natural and intuitive. The attributes both of the substrates and of the enzymes, as well as the rules relating them, are expressed in biochemical terms and phrases. Representing DNA metabolism in this way provides explanation and didactic capabilities that can be used readily by biochemists not involved in the development of the knowledge system. Knowledge of a metabolic step can be either detailed or sketchy and still can be represented by the rule-based methods we employ.

Currently, our representation paradigm requires that a user have an understanding of knowledge-representation techniques to change or add new information. For instance, to represent a new enzyme, the user must first create a new instance of the `ENZYMES` class. Then, he must specify all possible values of `SPECIFICITY`, `ACTIVITY`, `ACTION`, and `STATE` for the `ENZYME`, and write at least one rule to conclude each value of each attribute, following the general paradigm for enzyme representation described in section 3.1. To perform these operations, the user must know how to create a unit, how to specify the values allowed for attributes, what the syntax for writing a new rule is, and which semantics are allowed in the premises of those rules. KEE allows new units to be generated by simple menu selection and rules to be

built in a context-sensitive text editor. In a future version of the knowledge system, we intend to provide a programmatically driven enzyme-acquisition function that, in conjunction with a tutorial, will greatly facilitate these steps.

Specifically, we would like to automate as much of the enzyme-representation process as possible. It is clear to us that many enzymes will share many of their rules with other enzymes of their class (e.g., all exonucleases hydrolyze DNA from the ends), and only a few of the rules are needed to specify uniquely any instance of an enzyme class. Hence, one method for automating the knowledge-acquisition process would be to write prototypical rules describing an enzyme class that refer to object types; these rules could then be inherited by specific instances of the enzymes, with references to object types replaced by specific objects. This approach is similar to Karp's use of process hierarchies. For example, the class of 3' exonucleases would have a set of general rules describing the binding and hydrolysis of 3' termini of DNA. Instances of 3' exonucleases would be represented by instantiated rules from the class level and by additional rules describing the specific behavior of the enzyme.

One advantage of this paradigm for enzyme representation is the ease with which knowledge can be validated. The steps we have outlined guarantee one form of completeness, in that every action of the enzyme can be concluded from at least one set of experimental conditions. Because of the natural modularity of the rules, it is easy for an expert to examine the premises of every rule, either manually or programmatically, to determine whether they cover every situation leading to the conclusion. In addition, consistency of the rule set is checked at the same time by the TMS, which constantly monitors contradictions or violations of cardinality in the frame system.

We are addressing three major limitations of our current representation of enzymes. First, there is no provision in our model for rules concerning interactions among enzymes. One enzyme influences the activity of the other enzyme only through its effects on substrates or cofactors. For example, DNA ligase inhibits nick translation by DNA polymerase I by sealing nicks in the substrate. There are no rules indicating that DNA ligase can limit the extent of nick translation by displacing the DNA polymerase I in a competitive fashion based on the processivity of the polymerase reaction. We are developing a framework for enzyme-interaction rules.

Second, the simulation predicts the action of only a normal, intact, and uninhibited enzyme. We might want to study an enzyme that was missing one of its activities or that had one activity inhibited (e.g., a mutant enzyme, a chemically modified enzyme, or a specific enzyme inhibitor). Although it is possible to do this manually by duplicating the enzyme and removing or altering specific rules, we would like to develop an automatic method for inhibiting any single activity of an enzyme. This method would allow the system to analyze, via backward chaining, experimental situations in which one

or more activities may be missing. The system could then conclude from the results of an experiment which activities may be present or absent. Karp's system can simulate mutant enzymes.

The third limitation is the current lack of a communicative visual representation of the objects in the simulation. We would like to represent DNA molecules and enzymes with object-oriented graphics; we believe that we could summarize in small diagrams most of the information currently presented in lists of attribute values. A pathway could then be represented by a sequence of diagrams showing enzyme–substrate interactions; animation would be possible, as well.

Acknowledgments

We thank Lyn Dupré for her editorial comments. This work was supported by grant LM04957 from the National Library of Medicine. ARG and DHM are Training Fellows of the National Library of Medicine. The KEE Software was provided under the University Grant Program from IntelliCorp, Inc. We would also like to thank both IntelliCorp, Inc., and IntelliGenetics, Inc., for providing DLB with support for a sabbatical during which this work was initiated.

References

C. Bierbicher, M. Eigen, and W. Gardiner, "The Kinetics of RNA Replication," *Biochemistry* 22, 2544–2559 (1983).

D. Bobrow and M. Stefik, "Perspectives on Artificial Intelligence Programming," *Science* 231, 951–956 (1986).

D. Bobrow, ed., *Qualitative Reasoning About Physical Systems*, MIT Press, Cambridge, MA, 1988.

R. Brachman, R. Fikes, and H. Levesque, "KRYPTON: A Functional Approach to Knowledge Representation," *IEEE Computer* 16, 67–73 (1983).

D. Brutlag, "Expert System Simulations as Active Learning Environments," in R. Colwell, ed., *Biomolecular Data: A Resource in Transition*, Oxford University Press, Oxford, England, 1988.

B. Buchanan and E. Shortliffe, eds., *Rule-Based Expert Systems*, Addison-Wesley, Reading, MA, 1984.

E. Charniak and D. McDermott, *Introduction to Artificial Intelligence*, Addison-Wesley, Reading, MA, 1985.

R. Davis, B. Buchanan, and E. Shortliffe, "Production Rules as a Representation for a Knowledge-based Consultation Program," *Artificial Intelligence* 8, 15–45 (1977).

J. deKleer, "An Assumption-based TMS," *Artificial Intelligence* 28, 127–162 (1986).

J. deKleer and J. Brown, "A Qualitative Physics Based on Confluences," in D. Bobrow, ed., *Qualitative Reasoning About Physical Systems*, The MIT Press, Cambridge, MA, 1985.

R. Fikes and T. Kehler, "Control of Reasoning in Frame-based Representation Systems,"

Communications of the ACM 28, 904–920 (1985).

R. Franco and E. Canela, "Computer Simulation of Purine Metabolism," *European Journal of Biochemistry* 144, 305–315 (1985).

D. Freifelder, *Essentials of Molecular Biology*, Jones and Bartlett, Boston, 1985.

E. Friedberg, *DNA Repair*, W.H. Freeman, New York, 1985.

P. Karp and P Friedland, "Coordinating the Use of Qualitative and Quantitative Knowledge," in L. Widman, *et al*., eds., *Artificial Intelligence, Simulation, and Modeling*, Wiley, New York, 1989.

P. Karp, "A Qualitative Biochemistry and its Application to the Regulation of the Tryptophan Operon," this volume.

M. Kohn and D. Garfinkel, "Computer Simulation of Metabolism in Palmitate-perfused Rat Heart. I. Palmitate oxidation," *Annals of Biomedical Engineering* 11, 361–384 (1983a).

M. Kohn and D. Garfinkel, "Computer Simulation of Metabolism in Palmitate-Perfused Rat Heart. II. Behavior of Complete Model," *Annals of Biomedical Enineering* 11, 511–532 (1983b).

A. Kornberg, *DNA Replication*, W. H. Freeman, New York, 1980.

A. Kornberg, *1982 Supplement to DNA Replication*, W. H. Freeman, New York, 1982.

I. Lehman, "DNA ligase: Structure, Mechanism and Function," *Science* 186, 790–797 (1974).

H. Levesque and R. Brachman, "A Fundamental Tradeoff in Knowledge Representation and Reasoning," in *Proceedings of the CSCI/SCEIO Conference 1984*, CSCI, London, Ontario, 1984.

M. Mavrovouniotis, "The Identification of Qualitatively Feasible Metabolic Pathways, this volume.

S. Meyers and P. Friedland, "Knowledge-based Aimulation of Genetic Regulation in Bacteriophage Lambda," *Nucleic Acid Research* 12, 1–9 (1984).

M. Minsky, "A Framework for Representing Knowledge," in P. Winston, ed., *The Psychology of Computer Vision*, McGraw-Hill, New York, NY, 1975.

M. Minsky, *The Society of Mind*, Simon and Schuster, New York, 1986.

P. Modrich, I. Lehman, and J. Wang, "Enzymatic Joining of Polynucleotides. XI. Reversal of Escherichia Coli Deoxyribonucleic Acid Ligase Reaction," *Journal of Biological Chemistry* 247, 6370–6372 (1972).

N. Nilsson, *Principles of Artificial Intelligence*, Tioga, Palo Alto, CA, 1980.

E. Rich, *Artificial Intelligence*, McGraw-Hill, New York, 1983.

J. Rothenberg, "The Nature of Modeling," in L. Widman *et al*., eds., *Artificial Intelligence, Simulation, and Modeling*, Wiley, New York, 1989.

K. Schaffner, "Exemplar reasoning about biological models and diseases: A relationship between the philosophy of medicine and philosophy of science," *Journal of Medicine and Philosophy* 11, 63–80 (1986).

S. Schapiro, *The Encylcopedia of Artificial Intelligence*, Wiley, New York, 1986.

M. Stefik, "An Wxamination of a Frame-Structured Representation System," *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, 845–852 (1979).

M. Stefik and D. Bobrow, "Object-oriented Programming: Themes and variations," *Science* 6, 40–62 (1986).

R. Thomas, *et al*., "A Complex Control Circuit: Regulation of Immunity in Temperate Bac-

teriophages," *European Journal of Biochemistry* 71, 211–227 (1976).

M. Waser, *et al*., "Computer Modeling of Muscle Phosphofructokinase Kinetics,"*Journal of Theoretical Biology* 103, 295–312 (1983).

D. Weld, "The Use of Aggregation in Causal Simulation," *Artificial Intelligence* 30, 1–34, 1986.

J. Watson, *et al*., *The Molecular Biology of the Gene*, Benjamin/Cummings, Menlo Park, CA, 1987.

L. Widman, "Semi-quantitative 'Close-enough' Systems Dynamics Models: An Alternative to Qualitative Simulation," in L. Widman, *et al*., eds., *Artificial Intelligence, Simulation, and Modeling*, Wiley, New York, 1989.