Computational Simulations of Biological Systems

Adam R. Galper and Douglas L. Brutlag

I. Introduction

We seem to be at a point in the history of biology where new generalizations and higher order biological laws are being approached but may be obscured by the simple mass of data. (Morowitz and Smith, 1987)

The simple mass of data now available is causing a paradigm shift in biology (Gilbert, 1991). In the new paradigm, biologists will begin their investigations with nearly complete knowledge of an organism's genome; they will posit a theoretical conjecture and will first turn to computational experimentation to test the hypothesis. *In vitro* or *in vivo* experimentation will be conducted only after a computational model of the biological system—which draws on sequence data, structural data, and all derived knowledge—supports the possibility of the conjecture. Observation and analysis will remain the primary means for acquiring knowledge, but the simulation of a computational model will guide physical experimentation.

Simulation is well established as a fundamental mode of scientific exploration when a highly complex process is poorly understood. A simulation attempts to mimic the real behavior of a dynamic process. Deviations of a simulation from observed behavior indicate either limitations or errors in knowledge about the process. These observed
differences often suggest verifiable experimental hypotheses to extend knowledge.

The science of biology has always employed simulation techniques. For example, 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. She then fractionates the extract to purify the components that catalyze individual reactions. Finally, 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.

A simulation can thus serve as a diagnostic check of a model. Biological models, however, are difficult to characterize in a standard fashion; whereas models in physics and chemistry are rife with mathematical axiomatizations, biological models have historically defied rigorous, reductionist formalizations. One reason for this situation is that traditional biology is more data driven than it is theory or model driven; biologists seek data to synthesize rather than to confirm biological theories. Indeed, the theory-driven physical sciences have molded the notion of a scientific theory, whereas the theories of the nonphysical sciences have been forced to fit the mold.

The prospect of a theory-driven biology, as envisioned by Gilbert (1991) (a former physicist), is a compelling reason to work toward the formalization of biological knowledge. For decades, philosophers of science have struggled with the structure of biological theories; just as the mathematical structures underlying theories of the physical sciences are amenable to computational manipulation, biological-theory structures can serve as the basis for the computational representation of biomedical knowledge.

A. Biological-Theory Structures

The 1987 Biomatrix Workshop (Morowitz and Smith, 1987) proposed an overall organizational structure to biological theories that includes three major orderings:

1. The hierarchy of size or organizational complexity, from atoms to the global ecosystem, passing through molecules, polymers, organelles, cells, tissues, organs, organisms, populations, and biomes.
2. The hierarchy of temporal processes, from atomic to geochemical and evolutionary processes.
3. A bioenergetic ordering, from metabolic to geothermal processes.
In addition, the workshop suggested that the relationships between biological functions and processes could be formalized by a relational algebra, which would be amenable to graph-theoretic and other mathematical techniques.

Schaffner (1980, 1987, 1990) has suggested that the typical theory in the biomedical sciences is a structure of overlapping interlevel temporal models. The models are interlevel in that they contain component parts that are often specified in terms of "intermingled organ, cellular, and biochemical terms." Furthermore, he argues that biological theories fall into the "middle range," between the extremes of universal generalization and of data summarization; in addition, the entities of biological theories tend to be in the middle range of organizational complexity, whereas the time scales of biological phenomena tend to fall between the atomic and the evolutionary.

Many researchers have recently embraced object-oriented programming techniques for the computational representation of biological theories (Brutlag et al., 1991; Karp, 1989; Schaffner, 1987). Object-oriented principles (Stefik and Bobrow, 1986) permit the representation of the hierarchies and of interlevel, middle-range models suggested by biological-theory structures. In object-oriented designs, objects represent real entities that store local data and perform computations; an object may be a gene, aware of its sequence and other features, and able to transcribe itself in the presence of RNA polymerase. Objects perform cooperative computations by sending messages to one another; an RNA polymerase object may send the transcribe message to a gene object to generate a transcript object.

Object-oriented techniques are not required for biological simulation—indeed, for years, biological simulations have been performed using mathematical methods—but they encourage the development of highly modular models that can be explained easily. In addition, robust object-oriented representations of biological theories can facilitate the automated discovery of new biological theories (Karp, 1989). Artificial intelligence (AI) approaches for exploring hypothesis spaces have traditionally used frame-based and related object-oriented techniques. Simulation modelers in many domains have only recently embraced the principles of object-oriented design (Widman et al., 1989).

B. Simulation Models

The complexity of biological systems is in part due to the large number of interactions among components at different levels of the organizational hierarchies. For example, the presence of a toxic molecule in a cell may have direct effects on receptor molecules and indirect effects
on cell organelles, cellular function, and physiological behavior. This behavior is often counterintuitive, thus making it difficult to design and interpret experiments.

A simulation model can assist the scientist in the design and analysis of experiments by approximating an analytical model when no such model is available. In many biological systems, detailed quantitative knowledge is unavailable; instead, a modeler must combine qualitative knowledge of relationships and scarce quantitative data to construct a model of the system. In these cases, simulating such a model may be the only means for generating predictions. Simulations can assist at every step of the scientific method, from organizing data to testing and generating hypotheses. As a simulation embodies theories, educators can use an interactive simulation model to teach biological theories.

II. Levels of Biological Abstraction

Biological systems are physical systems; any collective behavior can, in theory, be explained in terms of physical and chemical interactions. The amount of computation required to model simple biological phenomena in terms of molecular structure is, however, daunting; molecular-dynamics simulations attempt to simulate the behavior of macromolecules using physical principles, often on an atom-by-atom basis (Levitt and Sharon, 1988). A molecular-dynamics simulation of significant biochemical behavior currently requires tremendous computational resources.

Biological science meets physicochemical science at the biochemical level, where metabolic events control living systems. Metabolism can be, and often is, explained at the atomic and molecular levels, but metabolic knowledge is expressed most succinctly in terms of quantified structural transformations, such as chemical reactions.

At the cellular level, metabolic changes affect cellular structure and behavior; a genomic mutation, through a series of metabolic steps, can affect the behavior of a single cell or of a whole cell line. In turn, tissues are composed of interacting cells, and organisms are composed of interacting tissues. Clearly, an atomic event, such as irradiation, has far-reaching effects at the metabolic, cellular, tissue, and organismal levels. The question facing the computational modeler is how best to represent and use knowledge at each level.

Biological abstraction continues beyond the organism. A collection of interacting organisms constitutes a population, populations exist in biomes, and ecosystems contain collections of populations and biomes. Simulations of biological systems at the ecological level should not be
concerned with representing change at the atomic level for several reasons. First, a model of one level of abstraction that relies on changes at a non-neighboring level is prone to intractability. Second, the user of an ecosystemic model typically does not want explanations of large-scale change at the atomic level. Third, a model builder should not have to represent coarse-grained knowledge, such as knowledge of change over great distances and large time intervals, in terms of infinitesimal spatial and temporal parameters.

A simulation of cellular behavior should be able to explain its predictions at the metabolic or molecular level. In general, most useful simulations employ interlevel models. Interlevel refers to “entities grouped together in a theory which are at different levels of aggregation. Roughly, an entity $e_2$ is at a higher level of aggregation than entity $e_1$ if $e_2$ has $e_1$ as (some of) its parts, and the defining properties of $e_2$ are not simple sums of $e_1$’s but require additional organizing relations” (Schaffner, 1990).

Interlevel models are most useful when knowledge of structural transformations is localized to a single level and the immediately neighboring levels. For example, when DNA is damaged, the effect of the damage should be represented by a change in the level of the encoded protein and not by a change in the physiology of the organism.

III. Simulation Methods

Simulation is a modeling technique that represents the behavior of individual components of a system over time. An analytic 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. A 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 modeler encodes the low-level, pairwise interactions of components, runs the simulation, and observes the higher-level patterns of interaction.

The analytic approach to metabolic simulation, for example, 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 (Biebricher et al., 1983; Franco and Canela, 1984; Kohn and Garfinkel, 1983; Waser et al., 1983). The feasibility of the analytic approach is, however, 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.

Analytic representations, such as differential equations, lack the robustness required to handle partial and uncertain knowledge. In addition, because analytic simulations model relatively similar structures over relatively similar temporal intervals, interlevel simulations are highly constrained.

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 (Widman et al., 1989).

Most importantly, discrete-event simulations provide natural support for qualitative representation and reasoning techniques, which offer explicit treatment of causality. The discrete-event approach can provide declarative representations for both the structures in the domain and the processes that act on these structures.

A. Structural Knowledge

Structural knowledge of a physical system is the foundation of a simulation. Most analytic and discrete-event simulations employ state-variable representations of physical entities. State variables describe the relevant qualitative or quantitative attributes of the system, but the structure of the system is expressed in terms of mathematical relationships among the state variables. For example, enzyme and substrate concentrations are state variables in a simulation of Michaelis–Menten enzyme kinetics.

An object-oriented representation is a type of state-variable representation in which variables are grouped together into objects that correspond to real-world entities. The structure of the system is represented in terms of the relationships between objects, as well as the relationships between the attributes (state variables) of objects. Object-oriented modeling techniques often provide the overall organization that is lacking in traditional state-variable representations of physical systems.

1. Object-Oriented Modeling

There are many excellent books on object-oriented design principles (Booch, 1991; Cox and Novobilski, 1991); here, we briefly review several principles relevant to biological modeling.
Object-oriented techniques provide a framework for representational generalization and specialization. For example, a modeler can define a class of objects known as MACROMOLECULES, of which there are several more specific subclasses, such as PROTEINS and NUCLEIC-ACIDS (Figure 1). An instance of PROTEINS, such as TRYPSIN, inherits attributes both from the PROTEINS class and from the MACROMOLECULES superclass.

Although the subtype relationship just described is most common, any object relationship can benefit from the inheritance mechanism. For example, many biological systems can be described as a collection of components; in some object-oriented programming environments, inheritance hierarchies can be designed for the part-of relationship. In this case, a component inherits some of its attributes and behaviors from the object of which it is a part (Section III.A.2).

Objects encapsulate both data and methods. Objects communicate with one another by passing messages, which execute the methods. Some methods alter the state variables contained within an object, whereas other methods create new instances of object classes. Inheritance, encapsulation, and message passing are powerful representational features that afford object-oriented designs a high degree of modularity and compactness.

2. Compositional Hierarchies

As described in Section I.A, biological theories rely on the compositionality of biological structures. A compositional hierarchy is an object-oriented class hierarchy based on the part-of relationship. At almost every level of abstraction, biological structures can be represented effectively as a collection of interacting parts. For example, Figure 2 depicts a compositional hierarchy for describing DNA structures in terms of strands, regions, nodes, sites, and termini.
B. Process Knowledge

Structural knowledge alone captures the state of a system at a fixed point in time, but does not capture the relationships and interactions among structural components over time. Process knowledge is functional knowledge of dynamic change. A declarative process representation is critical to the success of a simulation.

Process knowledge can be represented declaratively in several forms. A rule-based representation specifies the preconditions for change and the effects of the change in a unit known as a rule. For example, the effect of tetracycline on the mechanism of protein synthesis can be expressed in the following form:

(IF tetracycline is present
 THEN tetracycline will inhibit the binding of aminoacyl-tRNA to ribosome)

Rules are the predominant declarative representation of processes (see Sections IV,A and IV,B). Processes can also be represented with constraints. For example, a chemical reaction can be represented as a set of reactants, a set of products, and a set of stoichiometric constraints; the research described in Section IV,C employs a constraint-based approach to process representation.
C. Declarative Device Models

Karp proposes the term *declarative device modeling* for the subfield of AI that views electrical, mechanical, biological, and other systems as devices (Karp, 1989); the structure and function of these devices must be elucidated to predict future behavior, to explain past behavior, to diagnose current behavior, and to design new and improved devices. A declarative device model allows different computational agents to reason about the model by accessing its structural and functional components.

We should acknowledge that biological devices are less well understood than are manufactured devices; consequently, biological simulations often yield highly uncertain results. An ongoing goal of simulation researchers is to develop robust methods for quantifying the uncertainty in device models and in simulation predictions.

IV. Review of Research on Biological Simulation

This section is concerned exclusively with simulations at the metabolic level. For a review of higher-level simulations, see Robertson and colleagues (1991). We review in detail three recent projects to simulate biological phenomena using the methods described in Section III.

A. Knowledge-Based Simulation of DNA Metabolism

We have built a rule-based, discrete-event simulation of DNA metabolism (Brutlag *et al.*, 1991). In particular, we have focused on the pathways of DNA replication and repair in *Escherichia coli*. The simulation relies on a panoply of AI techniques for representation, inference, and explanation; this type of simulation is often referred to 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 assumption, 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.

Developed in KEE (the Knowledge Engineering Environment, by IntelliCorp), 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.

1. DNA Metabolism

The major mechanisms of DNA metabolism include replication, repair, transcription, and mutation. 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, the most important of which is DNA polymerase I.

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, 1992). 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 deoxy-nucleoside 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 situation occurs, 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 can then 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 occurs simultaneously, nick translation may occur. Polymerization and exo-
nucleolytic degradation are the primary activities of DNA polymerase I, as depicted in Figure 3.

The catalytic actions mediated by DNA polymerase I and other DNA enzymes, such as 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 to the substrate either too tightly or too loosely, and synthesis of new DNA will be thwarted. If NAD is not present, then DNA ligase will not seal a nick. These catalytic actions, as well as all those depicted

<table>
<thead>
<tr>
<th>Intact duplexes</th>
<th>Template-Primer</th>
<th>Action</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No change</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>No change</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nicked duplexes</th>
<th>Template-Primer</th>
<th>Action</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3'</td>
<td>Strand displacement or Nick translation</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Strand displacement or Nick translation</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gapped duplexes</th>
<th>Template-Primer</th>
<th>Action</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3'</td>
<td>Gap-filling</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Single strands</th>
<th>Template-Primer</th>
<th>Action</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3'</td>
<td>Chain elongation</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Primed single strands</th>
<th>Template-Primer</th>
<th>Action</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3'</td>
<td>Chain elongation</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3. Activity of DNA polymerase I on various templates and primers. Adapted, with permission, from Kornberg (1992).
in Figure 3, can be expressed succinctly as rules, with the appropriate representations of enzymes, substrates, and conditions.

2. Representation of Simulation Objects

We define three classes of simulation objects: DNAs, ENVIRONMENTAL-CONDITIONS, and ENZYMES. In a typical simulation, we would introduce at least three major objects: DNA, an instance of the DNAs class; CONDITIONS, an instance of the ENVIRONMENTAL-CONDITIONS class; and at least one instance of the ENZYMES class, for example, the DNA-POLYMERASE-I and DNA-LIGASE objects.

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. There are 96 DNA-RULES instances, each of which infers the value of a single DNAs instance.

Our substrate representation can describe DNA at four levels. 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'-external 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 attribute 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 attributes that specify components of the overall structure of the molecule: The NICKS attribute describes qualitatively the nicks present (NONE, SOME, ONE, MULTIPLE), the TOPOLOGY attribute specifies the possible shapes (e.g., LINEAR, Y-FORM, CIRCULAR), the STRANDEDNESS attribute can take on the values SINGLE-STRANDED and DOUBLE-STRANDED, and the STRANDS attribute describes the strands independent of topology (e.g., INTACT-DUPLEX, NICKED-DUPLEX, PRIMED-SINGLE-STRAND). Finally, at the highest descriptive level, the overall STRUCTURE attribute offers a list of common DNA structures (e.g., PRIMED-CIRCLE,
NICKED-LINEAR, COVALENTLY-CLOSED-CIRCLE), from which the value of component attributes can be inferred. We can conceive of multiple, independent DNAs instances 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 DNAs instances, a duplex and a single-stranded molecule, each of which will interact differently with the enzymes present.

The CONDITIONS object contains three quantitative attributes that describe the physical environment: TEMPERATURE, PH, and IONIC-STRENGTH. Values of these attributes have been mapped into discrete ranges to facilitate purely qualitative reasoning. 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 among enzymes, substrates, and the environment. The other attributes in the CONDITIONS object, including NUCLEOTIDES, MONOVALENT-CATIONS, DIVALENT-CATIONS, ANIONS, and COFACTORs, represent physical objects, and could be modeled as objects in the simulation. We have chosen not to model them thus, because we are interested in these objects only by virtue of their presence or absence. 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. The active image for the CONDITIONS object is displayed in Figure 4; the simulation user can observe and modify this object, as well as all other simulation objects, with a mouse-based graphical interface provided by KEE.

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 STATE of an enzyme object influences the object's ACTIVITY and SPECIFICITY. Depending on the enzyme, the ACTIVITY, SPECIFICITY, ACTION, and STATE attributes can take on different values. For example, DNA polymerase I can display binding activities (e.g., XMP-BINDING, XTP-BINDING, DNA-BINDING), synthetic activities (DIDEOXY-CHAIN-TERMINATION, STRAND-DISPLACEMENT), or degradative activities (3P-EXONUCLEASE, 5P-EXONUCLEASE). Similarly, DNA ligase can bind (DNA-ADENYLYLATION, SELF-ADENYLYLATION), synthesize (SEALING-ACTIVITY), or degrade (ENDONUCLEASE-AC-
TIVITY). Recall that the ACTIVITY value depends on the environmental conditions; the SPECIFICITY attribute for each enzyme has similar types of values, but depends on the substrate description. Attributes 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.

3. Representation of Processes

The metabolite representations described in Section IV.A.2 correspond to the working memory of a production system; the rule set, which operates on working memory, captures knowledge of the potential interactions among and the behaviors of the metabolites. Rules
for simulating DNA-POLYMERASE-I action are all instances of the DNA-POL-I-RULES class.

We structure enzyme rule hierarchies along the same lines as we do 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 the POL-I-ACTIVITY-RULES class, describing the effect of the environment on the activity of an enzyme, is

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

Another POL-I-ACTIVITY-RULES instance may reference previously deduced activities, in addition to other attributes 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. 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
    (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. We 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. 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 attribute 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 attribute 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. This rule would be expressed as follows:

(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-3P-ENDS OF DNA ARE ?Z)
THEN
IN.NEW.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 many steps into one for this process. Other processes require a finer granularity of representation.

4. Example: Prediction of Enzyme Action

To illustrate the predictive power of the system, we describe briefly a sequence of assertions and conclusions; see Galper et al. (1993) for details. We begin with an experimental environment at 37°C and a

![conditions table]

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Nucleotide</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temp. (°C)</td>
<td>DDTTP</td>
</tr>
<tr>
<td>pK</td>
<td>DDGTP</td>
</tr>
<tr>
<td>Cations2+</td>
<td>DDCTP</td>
</tr>
<tr>
<td></td>
<td>DDATP</td>
</tr>
<tr>
<td></td>
<td>UMP</td>
</tr>
<tr>
<td></td>
<td>GMP</td>
</tr>
<tr>
<td></td>
<td>CMP</td>
</tr>
<tr>
<td></td>
<td>AMP</td>
</tr>
<tr>
<td></td>
<td>UDP</td>
</tr>
<tr>
<td></td>
<td>GDP</td>
</tr>
<tr>
<td></td>
<td>CDP</td>
</tr>
<tr>
<td></td>
<td>ADP</td>
</tr>
<tr>
<td></td>
<td>RTP</td>
</tr>
<tr>
<td></td>
<td>UTP</td>
</tr>
<tr>
<td></td>
<td>GTP</td>
</tr>
<tr>
<td></td>
<td>CTP</td>
</tr>
<tr>
<td></td>
<td>ATP</td>
</tr>
<tr>
<td></td>
<td>DTMG</td>
</tr>
<tr>
<td></td>
<td>DGMP</td>
</tr>
<tr>
<td></td>
<td>DCMG</td>
</tr>
<tr>
<td></td>
<td>DAMP</td>
</tr>
<tr>
<td></td>
<td>DTDP</td>
</tr>
<tr>
<td></td>
<td>DGDG</td>
</tr>
<tr>
<td></td>
<td>DCDP</td>
</tr>
<tr>
<td></td>
<td>DADP</td>
</tr>
<tr>
<td></td>
<td>DTTP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ionic Strength</th>
<th>DNA polymerase I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temp. (°C)</td>
<td>Activity</td>
</tr>
<tr>
<td></td>
<td>XMP-BINDING</td>
</tr>
<tr>
<td></td>
<td>XTP-BINDING</td>
</tr>
<tr>
<td></td>
<td>DNA-BINDING</td>
</tr>
<tr>
<td></td>
<td>3P-EXONUCLEASE</td>
</tr>
<tr>
<td></td>
<td>3P-EXONUCLEASE</td>
</tr>
<tr>
<td></td>
<td>DIDEOXY-CHAIN-TERMINATION</td>
</tr>
<tr>
<td></td>
<td>RIBO-CHAIN-TERMINATION</td>
</tr>
<tr>
<td></td>
<td>LIMITED-SYNTHESIS</td>
</tr>
<tr>
<td></td>
<td>RIBO-INCORPORATION</td>
</tr>
<tr>
<td></td>
<td>SYNTHESIS</td>
</tr>
<tr>
<td></td>
<td>STRAND-DISPLACEMENT</td>
</tr>
</tbody>
</table>

Figure 5. 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.
pH value of 7.4; under these conditions, DNA polymerase I displays no activity. If we increase the ionic strength from 0.0 to 0.1, as depicted in the active images of Figure 5, the system concludes that DNA polymerase I is able to bind to DNA. With the subsequent addition of the divalent cation Mg\(^{2+}\), the system concludes that DNA polymerase I can now show 3'- and 5'-exonuclease activities. After the addition of four nucleoside triphosphates—ribo-ATP, dTTP, dGTP, and dCTP—DNA polymerase I can now bind to these triphosphates, and can incorporate some of them; the system concludes a limited synthetic activity due to the lack of dATP. With the addition of Mn\(^{2+}\), however, DNA polymerase I can incorporate ribo-ATP into a growing strand; the activities RIBO-INCORPORATION, SYNTHESIS, and STRAND-DISPLACEMENT can now be concluded.

Next, 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 system assigns multiple values to the SPECIFICITY attribute of DNA-POLYMERASE-I, indicating 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. The simulation predicts seven actions for DNA polymerase I, the final active image of which is displayed in Figure 6. Each action can be explained with rule traces.

5. Example: Envisionment of Metabolic Pathways

Figure 7 depicts a partial envisionment of the metabolic pathways that originate with the gapped linear DNA molecule described in Section IV,A,4. KEE generates a graph of worlds; we have enhanced this graph to depict diagrammatically 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 IV,A,4. Each of these worlds is the result of a binding process. In the first world (W1), 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; the simulation generates a new world 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 (W2) contains the enzyme bound to the external
5' terminus of the gapped molecule. In this world, the enzyme can
hydrolyze the 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 labeled "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. As 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 (W3), the enzyme is bound to the internal 5' terminus and 5'exonuclease is a predicted action. In the world labeled "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. The simulation continues with the extension of the primer in one pathway and the hydrolysis of the primer in another (not shown in Figure 7).

The final world (W4) contains a situation encountered previously in the pathways originating from W1, W2, and W3; namely, the enzyme has bound free nucleotides, and no further activity is observed.

B. Qualitative Simulation of the Trp Operon

Intrigued by the prospect of automated scientific discovery, Karp (1989; 1993) sought to develop representations for scientific theories and experiments, to use these theories to predict the outcomes of experiments, and to formulate hypotheses to explain unexpected observations. His study focuses on the discovery of the attenuation mechanism of bacterial gene regulation in the trp operon.

Karp develops three increasingly expressive and complex declarative device models for the representation of the structures and processes crucial to the attenuation mechanism. The first model uses frames to represent objects, and production rules to represent interactions among objects. The second model uses a fixed-state-variable network to address quantitative aspects of the attenuation mechanism. The third model, called GENSIM (genetics simulator), uses compositional hierarchies to describe biological objects and processes. To reason with the GENSIM model, Karp presents two computer programs: one to predict the outcomes of GENSIM experiments, and another to generate hypotheses to explain differences between observed and expected outcomes. Karp developed the models and programs in the LISP language, using KEE. After a brief review of the trp operon, we discuss the models and programs in moderate detail.

1. The Trp Operon

Yanofsky and colleagues discovered the attenuation mechanism by studying the regulation of the biosynthesis of tryptophan (Yanofsky, 1981). The trp operon contains five structural genes, which encode three enzymes crucial to the synthesis of trp (Figure 8). Before the discovery of attenuation, the transcription of the trp operon was
thought to be regulated through Jacob–Monod repression, a mechanism by which an activated repressor protein binds to the operator, a region within the operon's promoter region; the repressor protein prevents the binding of RNA polymerase to the promoter and subsequent transcription. The trp repressor protein binds to the operator only in the presence of trp. Hence, the transcription of the trp operon varies inversely with the concentration of tryptophan in the cell; the cell produces more of the trp enzymes when tryptophan is scarce.

In the early 1970s, experiments indicated that Jacob–Monod repression could not fully explain the regulation of the trp operon. The attenuation mechanism explains how the trp operon can be regulated over a range wider than the one that simple Jacob-Monod repression allows. During transcription, the attenuator region of the operon signals RNA polymerase to terminate transcription prematurely. For this mechanism, the frequency of termination is directly proportional to the concentration of tryptophan in the cell. Thus, when there is less tryptophan available in the cell, both repression and attenuation direct the cell to manufacture the trp biosynthetic enzymes.

The discovery and elucidation of the attenuation mechanism of regulation spanned nearly 20 years of research. Karp's primary goal was to develop computational models and computer programs that could reproduce the discoveries and theory modifications made by Yanofsky's laboratory. Using techniques from knowledge engineering and information-processing psychology, Karp first studied the knowledge and reasoning processes employed by the biologists, so that he could develop a conceptual reconstruction of the discovery of attenua-
tion. Karp then analyzed this reconstruction for patterns (see Karp, 1989, for extensive details of the historical study). These patterns provided a basis for the design of declarative device models of the trp operon and for the design of a computer program for automated hypothesis formation.

2. Declarative Device Models of the Trp Operon

Karp presents three computational models for representing the evolution of scientific theories of attenuation. According to Karp, a scientific theory is a device for prediction, which takes as input a description of a system at time t and produces as output a description of the system at a later time t'. One class of theories generates predictions of experimental outcomes; another class predicts the structural arrangements of objects found in physical systems. Both types of theories make predictions by describing how objects interact over time.

Using frames to represent biological objects and production rules to represent interactions between objects, Karp’s first model focuses on only the initiation of transcription. This model is adequate for simulating the reactions involved in transcription initiation, but has severe limitations. For example, the model lacks any quantitative representations and thus cannot predict rates or amounts. The model is also unable to represent complex compositional hierarchies that exist in the trp system. Finally, the production-rule language lacks operators for negation, disjunction, and quantification, making the description of many complex processes difficult.

a. The Fixed-State-Variable Model To address several of the shortcomings of his initial model, Karp developed a second model to focus on quantitative aspects of the attenuation mechanism. This model contains knowledge of quantitative state variables, such as the concentration of trp or the rate of transcription initiation, but does not contain knowledge of the part-whole structure of the repressor-operator complex. The state variables are organized into a network that expresses the functional dependences among the variables. Figure 9 shows the fixed-state-variable network for the trp system. The network is fixed because it represents a single experimental system and cannot be used to represent, for example, a mutant bacterium growing in a different medium. State variables can be expressed with respect to quantitative values ([trp = 0.001], [trp > 0.005]) or with respect to other state variables ([trp = trp.maximum − 0.0001], [trp > trp.equilibrium]).

Karp represents interactions among state variables using a hierarchical framework that permits the approximate expression of mathematical relationships. Function frames describe how a set of input variables
Figure 9. The fixed-state-variable model. Karp's state-variable networks capture the dependencies among model variables in the trp operon. Variables representing concentrations and rates can be expressed in both qualitative and quantitative terms, and their effects can be propagated through the functional relationships specified by the network; however, each network represents only a single experimental configuration. Adapted, with permission, from Karp (1989).

combine—arithmetically, multiplicatively, or in an unknown fashion—to affect a single output variable. If a function frame is specified fully, the relationship constitutes a quantitative algebraic constraint; if the values of dependent state variables are unknown, the relationship imposes a qualitative constraint. Mapping frames describe observed values of a function and can be referenced by function frames. When the precise mathematical form of a function of several variables is unknown, mappings can be used to interpolate the function.
To simulate the trp system over time, a modeler first specifies the values of exogenous state variables, such as the RNA polymerase and trp concentrations. The model derives the values of all state variables at the next time point by propagating the previous state variables through the functional relationships specified by the network. The model predicts subsequent values of state variables by referring to values at the previous time point. The propagation algorithm is described in detail in Karp and Friedland (1989).

By incorporating imprecise specifications of both qualitative and quantitative knowledge, Karp’s fixed-state-variable model extends the work of several other researchers in the field of qualitative simulation; however, because each network is valid for only a single experimental configuration, Karp found the approach too inflexible for experiments in hypothesis formation. His final model addresses this limitation.

b. The GENSIM Model The GENSIM model combines techniques from the first two models. A class knowledge base (CKB) provides a library of object classes that may be present in an experiment on the trp system; classes may describe the decomposition of objects into their component parts. The CKB defines general classes, such as enzymes, operons, promoters, and amino acids, as well as classes specific to the trp system, including each gene and gene product within the trp operon. Objects are instantiated from these classes and are stored in the simulation knowledge base (SKB); each chemical object

Figure 10. Simulation knowledge-based objects in a transcription experiment. The operon and RNA polymerase objects are components of the transcription experiment (XCription.Expt) object and are composed of other objects. Adapted, with permission, from Karp (1989).
represents a population of molecules. Figure 10 displays some objects in a typical transcription experiment SKB. The process knowledge base (PKB) uses frames to describe chemical reactions, such as binding, rearrangement, and dissociation, that can occur among the objects in an experiment; processes can modify the properties of existing objects and can create new objects. GENSIM processes are based on the work of Forbus (1984). Because reactions are probabilistic events on populations of molecules, processes split each reacting population of molecules into two subpopulations: one that does react and one that does not react.

Unlike the fixed-state-variable model, the GENSIM model does not reason about quantitative state variables such as concentrations and rates, but instead attempts to predict what objects are produced in an experiment and what the configurations of those objects are. The model assumes that a population of molecules is never fully consumed, so that objects are never deleted from the simulation. Thus, during the course of a simulation, the number of objects increases monotonically. This assumption simplifies the implementation of GENSIM significantly.

The PKB arranges processes in an inheritance hierarchy, so that processes can inherit parts of their behaviors from more general process classes. A portion of the process hierarchy is depicted in Figure 11. Process frames have attributes to specify preconditions and actions; if the preconditions hold, the actions are taken. Preconditions and actions are represented in a process-description language developed especially for chemical reactions, and are executed by the GENSIM process interpreter. Several attributes from a process frame are displayed in Figure 12.

3. Simulation and Hypothesis Formation

The GENSIM process interpreter is similar to a production system whose working memory is the SKB. The interpreter detects interactions between objects, and computes the effects of these interactions. The process interpreter must activate processes whose parameter objects are present in the SKB, execute those processes whose preconditions are met, and manage the proliferation of new objects generated by process executions. Details of these methods are given in Karp's (1989) dissertation; Figure 13 shows a simulation of the reactions in the normal trp operon system.

When the observed outcome of an experiment does not match the outcome predicted by GENSIM, the HYPGENE (hypothesis generator) program suggests changes both to the theory embodied in the GENSIM model and to the presumed initial conditions of the experiment. HYPGENE takes as input a tuple \((I_A, P_A, Error_A, PKB, CKB)\), where \(I_A\) is
Figure 11. Part of the GENSIM process inheritance hierarchy. Solid lines indicate subclass relations; dashed lines indicate member relations. Subclasses of Processes that are not shown include dissociation and mutation processes. Adapted, with permission, from Karp (1989).
Parameter Object Classes: Trp-ApoRepressor
Parameter Objects: $A$ $B$
Bindings $M$: ($Complex\ Class\ Trp-Repressor$)
                  ($A_{obj}$ $A$)
                  ($B_{obj}$ $B$)
                  ($A_{site}$ $A_{site}$)
Preconditions $M$:
Preconditions $A$:

("* Check that $A_{obj}$ contains an active site which interacts with objects
  of $B_{obj}$'s type\)

([EXISTS $A_{site}$
  (AND (IS\ PART\ R $A_{site}$ $A_{obj}$)
        (OBJECT\ EXISTS\ $A_{site}$ 'Active\ Sites'))
  (EXISTS $A_{site}$ interaction\ class
    (AND (MEMB $A_{site}$ interaction\ class
          (W/GET\ VALUES $A_{site}$
            'Potential\ Interacting\ Objects'))
    (OBJECT\ EXISTS\ $B_{obj}$ $A_{site}$ interaction\ class))]

("* Check that $A_{site}$ isn't occupied)
[NOT (EXISTS $ob$]
  (AND (MEMB $ob$ (W/GET\ VALUE
    $A_{site}$ 'Object\ Interacting\ With\ Site'))
    (OBJECT\ EXISTS\ $ob$ (W/GET\ VALUE $A_{site}$
      'Potential\ Interacting\ Objects'))]

("* Mutation Check 1)
[NOT (EXISTS $mutation$ (AND (IS\ PART\ $mutation$ $A_{site}$)
    (OBJECT\ EXISTS\ $mutation$ 'Mutation'))
    (MEMB $ob$ Current\ Process
      (W/GET\ VALUES $mutation$
        'Processes\ Disabled'))]

[NOT (EXISTS $ob$ (IS\ PART\ $ob$ $A_{site}$
    (OBJECT\ EXISTS\ $ob$ 'Physical\ Entities'))]

Efficiency Preconditions $A$:
Effects $M$:
  (BINDV $A$ ($COPY\ STRUCTURE\ A$))
  (BINDV $B$ ($COPY\ STRUCTURE\ B$))
  (BINDV $Complex$ ($CREATE\ COMPLEX:\ Complex\ Class$
    (LIST $A_{site}$ $B_{site}$ 'RBOUND')))
Effects $A$:
  (W/PUT\ VALUE (OBJECT\ COPIED\ TO $A_{site}$
    Object\ Interacting\ With\ Site
    (OBJECT\ COPIED\ TO $B_{obj}$))]

Figure 12. Attributes from the process frame for Trp-ApoRepressor.Binds.Trp. The Preconditions and Effects attributes are expressed in a process-description language that is executed by the GENSIM process interpreter. Adapted, with permission, from Karp (1989).

the presumed initial conditions of experiment $A$, $P_A$ is the predicted outcome, and $Error_A$ is the difference between the predicted and observed outcomes. It generates as output a set of hypotheses, where each hypothesis is a tuple ($I_A$, PKB). Thus, HYPGENE can compute modifications to $I_A$ and PKB, such that the predicted outcome of the modified experiment matches the observed outcome. HYPGENE reasons backward through the dependency graph generated by a GENSIM simulation.

HYPGENE designs hypotheses by satisfying constraints and goals. HYPGENE's initial goal is to eliminate the prediction error $Error_A$. An agenda-based, best-first search algorithm controls HYPGENE's goal stack and searches for operators that can satisfy outstanding
Figure 13. Simulation of the normal trp system. This diagram shows the transcription of the trp operon, the translation of the mRNA, the reactions catalyzed by the trp biosynthetic enzymes, the reactions involved in the repression of the operon, and the degradation of the trp operon mRNA. The nodes represent objects in the simulation; solid directed arcs connect the reactants and products of each process. The dashed undirected arcs indicate ownership; that is, the Messenger.RNA.18 object contains the five ribosome binding site objects. Adapted, with permission, from Karp (1989).

goals. Operators may either remove satisfied goals from the stack or add new goals to the stack. Using theory-modification operators elucidated in the historical study, Karp proposes four types of design operators that alter $I_A$ and PKB to satisfy the goals on HYPGENE's
goal stack: initial condition modification operators, class-modification operators, process-modification operators, and quantity-hypothesis design operators. HYPGENE contains implementations of initial-condition modification operators and quantity-hypothesis design operators, but does not contain implementations of the class-modification and process-modification operators. Nevertheless, HYPGENE performs admirably on hypothesis-formation problems faced by scientists in the 1970s, in one case formulating plausible hypotheses that the scientists did not consider.

C. Design of Metabolic Pathways

Mavrovouniotis’ (1988) dissertation on the computer-aided design of biochemical pathways presents a framework for the systematic construction of biochemical pathways that satisfy a set of constraints. Knowledge of all possible pathways is important in the design of a biological process, when a pathway must be chosen to yield a desired product, and in the selection of microbial strains. The examination of features common to multiple pathways can help in the identification of fundamental constraints in the process. For example, several pathways may depend on the presence of a common intermediate or on the yield of a specific reaction.

To identify a mutant strain that lacks a particular enzyme, scientists must identify those sets of substrates that permit the growth of the mutant cell, as well as those that prevent the cell’s growth. The generation of biochemical pathways can assist scientists in predicting the ability or inability of mutant strains that lack certain bioreactions to grow on specified sets of substrates. Conversely, to block the growth of a strain on a specific set of substrates, an experimentalist must select the set of enzymes that should be eliminated to block all the pathways for the catabolism of the substrates.

The approach taken in Mavrovouniotis’ research is to represent metabolites and biochemical reactions as constrained objects and constrained processes, respectively. The constraints restrict the participation of the metabolites and reactions in the synthesized pathways. A reaction- and metabolite-processing algorithm, implemented in LISP, then yields all possible pathways by satisfying the constraints one at a time. Unlike the research described in Sections IV,A,B, Mavrovouniotis’ research does not rely on simulation methods to generate hypotheses, but rather relies on design and planning methods. In the following sections, we describe the representation, outline the algorithm, and present an example of the algorithm’s usefulness.
1. Constraints on Metabolites and Bioreactions

Mavrovouniotis' system is built on a database of metabolic intermediates and biochemical reactions. Metabolites are simple objects—that is, the chemical or compositional structure of metabolites are not represented—that are characterized by a name (e.g., fructose 6-phosphate) and by a set of labels that represent the constraints applicable to the metabolite's participation in the pathway. For a given pathway-design problem, Mavrovouniotis constrains the metabolites of intermediary metabolism along two dimensions. First, metabolites are constrained to be any one of net reactants, net products, or intermediates of the pathway. Second, metabolites can be marked as required for, allowed in, or excluded from the synthesized pathway. For example, a required reactant must be consumed by the synthesized pathway, whereas an excluded reactant must not be consumed. In realistic problems, most metabolites are excluded reactants. The constraints on a metabolite are not independent; for example, a required product must also be an excluded reactant.

The reaction database consists of a set of stoichiometric reactions among metabolites. For each reaction, the database stores the names of the reactant metabolites, the names of the product metabolites, and the stoichiometric coefficients for each reactant and product. Reactions are constrained to be in either the forward or backward direction. Like metabolites, reactions are specified as required, allowed, or excluded. A reaction required in the forward direction must be excluded in the reverse direction.

Mavrovouniotis refers to these constraints as stoichiometric for the following reason: if \( a_i \) is the stoichiometric coefficient of metabolite \( i \), then a required metabolite implies either \( a_i < 0 \) (a required reactant) or \( a_i > 0 \) (a required product). In theory, the specification of constraints could be extended to refer to the quantitative stoichiometries of reactions and pathways (e.g., \( a_i > 2 \)). The required and excluded constraints are strict, whereas the allowed constraint is loose. This distinction is important to the constraint-satisfaction algorithm.

2. The Constraint-Satisfaction Algorithm

The constraint-satisfaction algorithm consists of three phases: reaction processing, metabolite processing, and pathway marking. In the reaction-processing phase, excluded reactions are removed from the active reaction database, which contains all reactions in both the forward and reverse directions. Required reaction constraints are not processed until the pathway-marking phase. The reaction-processing
phase generates a set of one-step pathways, the members of which satisfy the excluded reaction constraints, the allowed reaction constraints, and the loose form of the required constraints.

The metabolite-processing phase iteratively composes this set of one-step pathways into larger and larger multireaction pathways, until all the imposed metabolite constraints are satisfied. At each step, the metabolite that participates in the fewest active pathways is selected. The algorithm then modifies the set of active pathways to satisfy the constraint on the metabolite. The program constructs new pathways as linear combinations of existing ones and deletes pathways that violate the current constraint. An important property of the algorithm is that once a constraint is satisfied, further linear combinations of an active pathway will never violate the constraint. At the completion of the metabolite-processing phase, only the loose forms of the constraints are guaranteed to have been satisfied in the active pathways.

The pathway-marking phase combines the loose-form pathways such that the resultant pathways each have at least one constituent pathway that satisfies each of the strict constraints. Again, linear combinations of the active pathways satisfy the union of the constraints satisfied by the component pathways. The output of this phase is a set of pathways, each of which includes at least one pathway that consumes each required reactant, at least one pathway that produces each required product, at least one pathway containing each required intermediate, and at least one pathway in which each required reaction participates.

A detailed description of the algorithm is presented in Mavrovouniotis (1988, 1990). In the worst case, the algorithm exhibits exponential time complexity with respect to the size of the reaction database; for pathways of fixed maximum length, however, the algorithm is polynomial. The algorithm is provably correct and complete and can generate partial results, that is, if the algorithm cannot run to completion, it can return a complete and correct list of pathways that satisfy a subset of the specified constraints.

3. Lysine Synthesis from Glucose and Ammonia

Mavrovouniotis demonstrates the utility of his method by examining the synthesis of the amino acid lysine from glucose and ammonia in bacteria. The thick arrows in Figure 14 depict the normal pathway for lysine synthesis. If a pathway is desired that bypasses the malate dehydrogenase reaction (because it is thermodynamically unlikely), the following constraints are used: glucose is a required reactant, lysine is a required product, and the malate dehydrogenase reaction is a
reaction excluded in the forward direction. The thick arrows in Figure 15 depict one alternative pathway (of several hundred generated) that bypasses the entire trichloroacetic acid (TCA) cycle through the direct carboxylation of pyruvate into oxaloacetate. From the stoichiometries of the reactions in the pathway, it can be shown that the pathway yields 1 mole of lysine for every 1 mole of glucose, whereas the pathway in Figure 14 yields 3 mole of lysine for every 1 mole of glucose.

The algorithm can also answer such questions as, "Can lysine be synthesized from glucose without the involvement of oxaloacetate?" When oxaloacetate is an excluded intermediate, reactant, and product, no pathways are possible, given the current database. The current database contains over 200 reactions and over 400 metabolites.

Although Mavrovouniotis' algorithm is correct and complete, it
Figure 15. A pathway that bypasses malate dehydrogenase. This pathway, one of many generated, relies on pyruvate decarboxylase and oxaloacetate decarboxylase to bypass the entire trichloroacetic acid cycle. Adapted, with permission, from Mavrovouniotis (1990).

does not distinguish among the plausibilities of the generated pathways. A well-formulated problem may generate 5000 alternate pathways. Early research on how to order these pathways according to their likelihood of occurrence employs thermodynamic group contribution methods (Mavrovouniotis, 1991). To estimate the Gibbs energy of a compound, Mavrovouniotis decomposes each compound into functional groups, for example, amine and carbonyl groups, and takes the sum of the thermodynamic contribution of each group. The Gibbs energy of a pathway is a linear combination of the Gibbs energies of that pathway’s component reactions, each of which is a linear combination of the Gibbs energies of formation of reactants and products. The Gibbs energy of each proposed pathway provides a means for ordering the alternative pathways.
V. Conclusion

The early attempts at biological simulation described in Section IV address specific problems, and serve more as demonstrations of general principles than as generalizable systems. There is no grand unifying technique for biological simulation; the method of choice may depend very much on the level of abstraction of the model and on the desired output of the simulation. A recurrent theme in these metabolic simulations is that the desired output—a simulated pathway, a likely hypothesis, or a reaction network—has a qualitative structure that must be selected from a vast discrete space using fundamentally qualitative techniques. A simulation can focus the search using quantitative data, if physical properties of the processes are known.

Much of the power of the qualitative simulation framework lies in the ability to represent and reason with compositional structures explicitly. The constraint-based and rule-based process representations have similar, but limited, functionality on state-variable structural representations. Karp turned from his constraint-based fixed-state-variable network to his rule-based compositional model (GENSIM) in large part because of the benefits provided by compositional hierarchies. Likewise, Mavrovouniotis (1991) has embraced the compositionality of metabolite structures in his thermodynamic group contribution research. Figure 2 reflects the compositional approach to our simulation research.

Although the reviewed biological simulation frameworks have produced useful and interesting results, they all lack explicit representations for uncertainty and time. Methods for producing estimates of the likelihood of a simulated pathway, of a generated hypothesis, or of a designed pathway and for differentiating between processes that are temporally dissimilar are active areas of current research. Analytical simulation methods offer several approaches to the representation of uncertainty and time in biological processes, but researchers are still struggling to integrate quantitative and qualitative reasoning techniques in a single paradigm for simulation (Forbus, 1984; Williams and de Kleer, 1991).

Widespread acceptance of biological representation and simulation methods awaits the development of effective tools for knowledge acquisition. Until biologists can easily manipulate and augment biological knowledge representations, the power of the simulation, hypothesis-generation, and design methods described in Section IV will remain largely untapped. One hope of the simulation modeling community is that biological knowledge bases will grow as methods emerge for extracting knowledge automatically from existing sequence, structure, mapping, and taxonomic databases (Koile and Overton, 1989).
References


