Knowledge-Based Simulation of DNA Metabolism: Prediction of Enzyme Action[1]

Douglas L. Brutlag, Adam R. Galper and David H. Millis
Department of Biochemistry and the **Section on Medical Informatics**
Stanford University School of Medicine
Stanford, CA 94305

Abstract

We have developed a knowledge-based simulation of DNA metabolism that accurately predicts the actions of enzymes on DNA under a large number of environmental conditions. Previous simulations of enzyme systems rely predominantly on mathematical models. We use a frame-based representation to model enzymes, substrates, and conditions. Interactions between these objects are expressed using production rules and an underlying truth maintenance system. The system performs rapid inference and can explain its reasoning. A graphical interface provides access to all elements of the simulation, including object representations and explanation graphs. Predicting enzyme action is the first step in the development of a large knowledge base to envision the metabolic pathways of DNA replication and repair.

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. 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.

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,

[1]Submitted to *Computer Applications in the Biosciences*.

on a computer, known biochemical behavior. In addition, the simulation can serve as an interactive modeling tool for reasoning about metabolism.

1.1 Simulation Methods

Until recently, most metabolic simulations were performed using mathematical models of the reactions involved. These models often require the solution of systems of differential equations (Bierbicher, Eigen and Gardiner, 1983; Franco and Canela, 1984; Kohn and Garfinkel, 1983a; Kohn and Garfinkel, 1983b; Waser *et al.*, 1983). However, in most metabolic pathways, either we are unaware of all the steps involved or we lack rate constants for each step. This precludes the use of differential equations to describe the process. Even when reaction rates are known, differential equations incur great computational costs. They are also difficult to use interactively and cannot represent knowledge at many levels simultaneously.

Rule-based methods, on the other hand, allow the representation of knowledge at several levels (Buchanan and Shortliffe, 1984). For example, in some instances, the actual catalytic mechanism and intermediates are known and can be specified. In others, only reactants 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 description of metabolism can also be extremely 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 automatically deduce and retract conclusions when the underlying fact base changes, make the reasoning processes involved in rule-based simulations more robust and efficient (de Kleer, 1986).

Most important, a rule-based simulation has the ability to explain its predictions based on the known facts and the rules relating those facts. Explanation graphs show the flow of logic, the relationships among stated facts, and the deduced conclusions.

1.2 A Symbolic Simulation of DNA Metabolism

We have built a rule-based simulation of DNA metabolism. In particular, we have focused on the pathways of DNA replication and repair. The simulation is *symbolic*, as opposed to numerical, and relies on *qualitative*, instead of quantitative, representations. We have chosen to avoid the inherent complexity of quantitative reasoning, 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.

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 of a DNA substrate be paired and have a hydroxyl group for DNA polymerase I to extend a primer, we write

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

The premises and conclusions of each rule refer to *units* in the simulation (e.g. DNA, DNA-POLYMERASE-I); each unit, or *object*, has various *attributes*, or *slots* (e.g. EXTERNAL-3P-GROUP, INTERNAL-3P-END), each of which can take on a number of *values* (e.g. HYDROXYL, PAIRED). Unit representations, often referred to as *frame-based*, offer numerous advantages over other representational methods (Fikes and Kehler, 1985; Minsky, 1986; Stefik, 1979). For one, frames can be organized into hierarchies, in which the most specific objects, called *instances*, can inherit attributes and attribute values from the more general objects, called *classes*; for example, in many frame systems, instances are used as multiple copies of a class frame, with each instance sharing the attributes of the class, but differing in attribute values. In addition, hierarchical frame-based representations are *object-oriented* and *modular*

(Bobrow and Stefik, 1986; Brachman, 1979; Brachman, Fikes and Levesque, 1983; Levesque and Brachman, 1984; Stefik and Bobrow, 1986).

This paper describes the methods we use to predict enzyme action in our knowledge-based simulation. The prediction of enzyme action is the first step in the development of a system that simulates entire metabolic pathways (Galper, et al., 1990). Section 2 briefly describes our domain — in particular, the mechanisms of replication and repair with DNA polymerase I and DNA ligase. In Section 3, we present in detail our techniques for representing all objects and behaviors in the system. Section 4 provides a sample interaction with the system. Finally, section 5 summarizes our conclusions and discusses future research directions.

2. Domain of Application

We have begun a formal description of the replication and repair pathways of DNA metabolism in *E. coli*. In this paper, we report on the representation of two bacterial enzymes: *E. coli* DNA polymerase I and *E. coli* DNA ligase.

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 occurs 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 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 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.

3. System and Methods

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.

3.1. Development Environment

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.

3.2. Knowledge Representation

In a metabolic simulation, we are concerned with the representation of two kinds of entities: metabolites and processes. Metabolites are the objects of the simulation; they include substrates, enzymes, cofactors, and products. Processes act on the simulation objects to effect change; processes include interactions between multiple objects (e.g., enzyme-substrate interactions) and behaviors of single objects (e.g., enzyme denaturation). For example, gap-filling is the process by which DNA polymerase I changes a gapped DNA molecule to a nicked DNA molecule. In our system, processes are only invoked after predictions of action are made; first, our system predicts a process, and then, the process occurs. We separate process prediction from process execution to maintain modularity, as discussed in Section 3.4. In this paper, we address only the first step of process representation: prediction of enzyme action. In Sections 3.3 through 3.6, we distinguish between the representation of metabolites and the representation of the predictive aspects of processes.

3.3. Representation of Metabolites

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 3.4. There are currently four major simulation 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 consistency 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. For example, the DNA-KB makes no reference to any knowledge contained in the CONDITIONS-KB, and vice versa; the enzyme knowledge bases, however, extensively refer to instances contained in DNA-KB and CONDITIONS-KB. Other designs are possible; for example, as our enzyme library grows and we begin to model enzyme-enzyme interactions, it may be worthwhile to collect all knowledge of interactions in a process knowledge base, using the techniques explored by Karp (Karp, 1989). Our current distributed design will eventually allow us to easily 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. Alternatively, the user can specify the values of component slots, and the STRUCTURE slot 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 symbolic ranges to facilitate purely qualitative reasoning and to reduce the number of rules that cannot be handled by the TMS (see Section 3.5). Currently, the TEMPERATURE-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, 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, *functional* 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-DISPLACEMENT), 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.

3.4. Representation of Process Predictions

The metabolite representations described in Section 3.3 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 predicting 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 likewise 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[2]

```
(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 1.2. 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. Our approach to this problem is described in detail in (Galper, et al., 1990). Here, we briefly explain why we separate the prediction of enzyme action from execution.

[2]The quasi-English syntax of KEE rules does not imply that the system "understands" English. KEE permits the expression of well-formed formulas using the natural language-like syntax, (a/an/the *slot* of *unit* is *value*).

One guiding principle in our paradigm for enzyme representation is modularity. The ACTIVITY, SPECIFICITY, and ACTION slots of enzymes have strictly-defined dependences: ACTIVITY is inferred from CONDITIONS attributes, SPECIFICITY is inferred from DNA attributes, and ACTION is inferred from the ACTIVITY and SPECIFICITY slots of the enzyme. This design results in a large number of succinct rules that are easily understood by humans and easily explained by the system (see Section 3.5). In following this principle, we have separated the knowledge of enzymatic processes into two steps: prediction and execution. Prediction simply involves the deduction of the values of the `ACTION` slot for a given enzyme, as depicted in the previous `POL-I-ACTION-RULES` example. An execution, or *transition*, rule typically examines the structural features of the substrate, the `STATE` slot of the enzyme, and the predicted `ACTION` of the enzyme; if the premise is satisfied, the transition rule will modify various attributes of the substrate and alter the `STATE` slot of the enzyme. Another reason we separate prediction from execution is that transition rules require different reasoning mechanisms. Details are described in (Galper, et al., 1990).

3.5. Inference

We use both forward-chaining (data-driven) and backward-chaining (hypothesis-driven) inference. We also require a truth-maintenance system (TMS) to support *nonmonotonic reasoning*, in which the number of facts known to be true is not strictly increasing over time (de Kleer, 1986; Charniak and McDermott, 1985; Rich, 1983). One of the reasons we have chosen KEE as our development environment is that it supports all of these reasoning mechanisms.

The TMS manages the dependencies between 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. The TMS is similar to a forward chainer in that both examine facts that are currently true in order 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 it.

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 of the knowledge system lead to a contradiction, the user is informed of the contradiction. The system can then display a complete explanation of the origin of the contradiction in terms of both the competing facts and the conclusions derived from them (see Section 3.6).

In our system, 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 retract the 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.

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

```
(IF (OR (THE EXTERNAL-5′-ENDS OF DNA ARE PAIRED)
        (THE INTERNAL-5′-ENDS OF DNA ARE 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-SYNTHESIS))))
```

KEE cannot generate justifications for this rule, because the rule expresses explicit nonmonotonic reasoning — its conclusion forces the deletion of an existing fact. Likewise, rules with certain operators as premises, including LISP expressions, do not generate TMS justifications. These rules are referred to as *same-world-action* rules within KEE; we also refer to these rules as *non-TMS* rules.

Since justifications are not generated for non-TMS rules, these rules are not automatically invoked when their premises become true, as is the case for TMS rules. 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 daemons (or KEE active values) are attached to the attributes mentioned in the antecedents of non-TMS rules. These daemons 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 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 unit describing the rule 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 its action and to change its behavior.

3.6. Explanation

Our knowledge system has a mechanism by which it can explain its reasoning. 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. 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 a 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

The user engages the prediction mode of the system by asserting known facts about an experimental system; this typically involves 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. We present a brief example, using DNA polymerase I in isolation.

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 polymerase 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).

5. Discussion

This qualitative simulation of DNA metabolism has several advantages over classical quantitative simulation methods. First, the representation of enzymes and substrates is extremely 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, rather than in differential equations. Differential equations capture the functional relationships between variables, but do not represent structural relationships nor the assumptions underlying both functional and structural relationships. Our qualitative representation also provides explanation and didactic capabilities that can be readily used by biochemists not involved in the development of the knowledge system. The modular and object-oriented nature of frame systems allows the user to rapidly shift focus or attention from object to object. The speed with which our rule system operates permits rapid, interactive analyses of different experimental conditions and substrates. Knowledge of a metabolic step can be either detailed or sketchy and still be readily represented by rule-based methods.

Rule systems permit a few rules (currently less than 100 rules for DNA polymerase I) to describe the action of a complex enzyme under many different experimental conditions with varying substrates. We can calculate an estimate of the total number of distinct experimental conditions that our rule system can handle by multiplying the number of independent experimental states represented in the premises of the rules. This estimate indicates that the action of DNA polymerase I alone can be predicted from over 10^{15} qualitatively different experimental situations (2^{41} states determined by 41 independent values of premises in less than 100 DNA polymerase I rules). Similarly, an upper bound on the number of qualitatively distinct final predictions of DNA polymerase I action is given by $2^{|\text{distinct specificities}|} \times 2^{|\text{distinct activities}|}$. We estimate that there are on the order of 10^7 qualitatively different predictions that our system can make for the action of DNA polymerase I. With a qualitative representation, we can characterize the relevant information processing capabilities of the model. For example, we can conclude that each prediction for DNA polymerase I can arise from over $10^{15} / 10^7 = 10^8$ qualitatively different initial conditions, on average.

Another important characteristic of rule systems is that the predictions are independent of the order in which the rules are acquired in the knowledge system. The order in which rules are fired is determined by the order in which facts are asserted or concluded and the natural linkage of rule premises and conclusions during the forward and backward chaining processes. Thus, it is simple to add to the rule-based system and to increase its knowledge incrementally without invalidating or revising existing knowledge. This property of rule systems allows new rules and facts to be added to the system as they are discovered, and facilitates the development of a simulation by many people independently.

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 3.3. 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 (Karp, 1989). 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.

Our paradigm for enzyme representation guarantees one form of completeness, in that every action of the enzyme can be concluded from at least one set of experimental conditions. Earlier, we estimated that each enzyme action may arise from 10^8 qualitatively different starting conditions. The terseness and modularity of the rules permits 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, the TMS checks the consistency of the rule set and constantly monitors contradictions or violations of cardinality in the frame system.

One limitation of this simulation is that it 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 has one activity inhibited (mutant enzymes, chemically modified enzyme, or the presence of a specific enzyme inhibitor). Although it is possible to do this manually by duplicating the enzyme and removing or altering specific rules, we plan to develop an automatic method for inhibiting any single activity of an enzyme. This 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.

In summary, we have described a novel approach to the representation and prediction of enzyme action in DNA metabolism. Previous simulations of enzyme systems rely predominantly on quantitative models and typically require the integration of a set of differential equations describing fluxes in the metabolism (Franco and Canela, 1984). Our qualitative representations capture the relevant structural features of substrates and the important functional features of enzymes. In a companion paper, we describe preliminary results on the envisionment of metabolic pathways using this approach (Galper, 1990).

Availability

The knowledge base described above is available, by anonymous FTP, from the Internet host DNA.STANFORD.EDU. The current knowledge base has been tested under KEE 3.1 on the following platforms: SUN Sparcstation 1 (OS 4.0.3 with Lucid Common LISP), IBM RT-135 (AIX OS with Lucid Common LISP), Macintosh-TI Microexplorer, Xerox 1109 and Xerox 1186 (Lyric Release of Xerox Common LISP). Address all inquiries and comments about the knowledge base to BRUTLAG@DNA.STANFORD.EDU.

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 and IntelliGenetics, Inc., for providing DLB with support for a sabbatical during which this work was initiated.

References

Bierbicher, C. K., Eigen, M. and Gardiner, W. C. J. (1983) The kinetics of RNA replication. *Biochemistry*, **22**, 2544–2559.

Bobrow, D. G. and Stefik, M. J. (1986) Perspectives on artificial intelligence programming. *Science*, **1986**, 951–956.

Brachman, R. J. (1979) On the epistemological status of semantic networks. In *Associative Networks: Representation and Use of Knowledge by Computers*. Academic Press, New York, pp. 3–50.

Brachman, R. J., Fikes, R. E. and Levesque, H. J. (1983) KRYPTON: A functional approach to knowledge representation. *IEEE Computer*, **16**, 67–73.

Brutlag, D. L. (1988) Expert system simulations as active learning environments. In Colwell, R. R. (ed), *Biomolecular Data: A Resource in Transition*. Oxford University Press, Oxford, pp. 185–188.

Buchanan, B. G. and Shortliffe, E. H. (1984) *Rule-based Expert Systems: The MYCIN experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA.

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

de Kleer, J. (1986) An assumption-based TMS. *Artificial Intelligence*, **28**, 127–162.

Fikes, R. and Kehler, T. (1985) Control of reasoning in frame-based representation systems. *Communications of the ACM*, **28**, 904–920.

Franco, R. and Canela, E. I. (1984) Computer simulation of purine metabolism. *Eur. J. Biochem.*, **144**, 305–315.

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

Galper, A., Millis, D., and Brutlag, D. (1990) Knowledge-based simulation of DNA metabolism: Prediction of action and envisionment of pathways. KSL Technical Report 90-??, Medical Computer Science Group, Stanford University, Stanford, CA.

Karp, P. (1989) *Hypothesis Formation and Qualitative Reasoning in Molecular Biology*, Ph.D. Thesis, Stanford University, Stanford, CA.

Kohn, M. C. and Garfinkel, D. (1983a) Computer simulation of metabolism in palmitate-perfused rat heart. I. Palmitate oxidation. *Ann. Biomed. Eng.*, **11**, 361–384.

Kohn, M. C. and Garfinkel, D. (1983b) Computer simulation of metabolism in palmitate-perfused rat heart. II. Behavior of complete model. *Ann. Biomed. Eng.*, **11**, 511–531.

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

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

Lehman, I. R. (1974) DNA Ligase: Structure, mechanism and function. *Science* , **186**, 790–797.

Levesque, H. J. and Brachman, R. J. (1984) A fundamental tradeoff in knowledge representation and reasoning. *Proceedings of the CSCI/SCEIO Conference 1984,* CSCI, London, Ontario.

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

Modrich, P., Lehman, I. R. and Wang, J. C. (1972) Enzymatic joining of polynucleotides. XI. Reversal of *Escherichia coli* deoxyribonucleic acid ligase reaction. *Journal of Biological Chemistry,* **247**, 6370–6372.

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

Schaffner, K. (1986) 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.

Stefik, M. (1979) An examination of a frame-structured representation system. *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*. IJCAI, pp. 845–852.

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

Waser, M. R., L., G., C., K. M. and D., G. (1983) Computer modeling of muscle phosphofructokinase kinetics. *Journal of Theoretical Biology,* **103**, 295–312.

Figure Legends

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

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.

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.

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.

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

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.

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.

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).

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

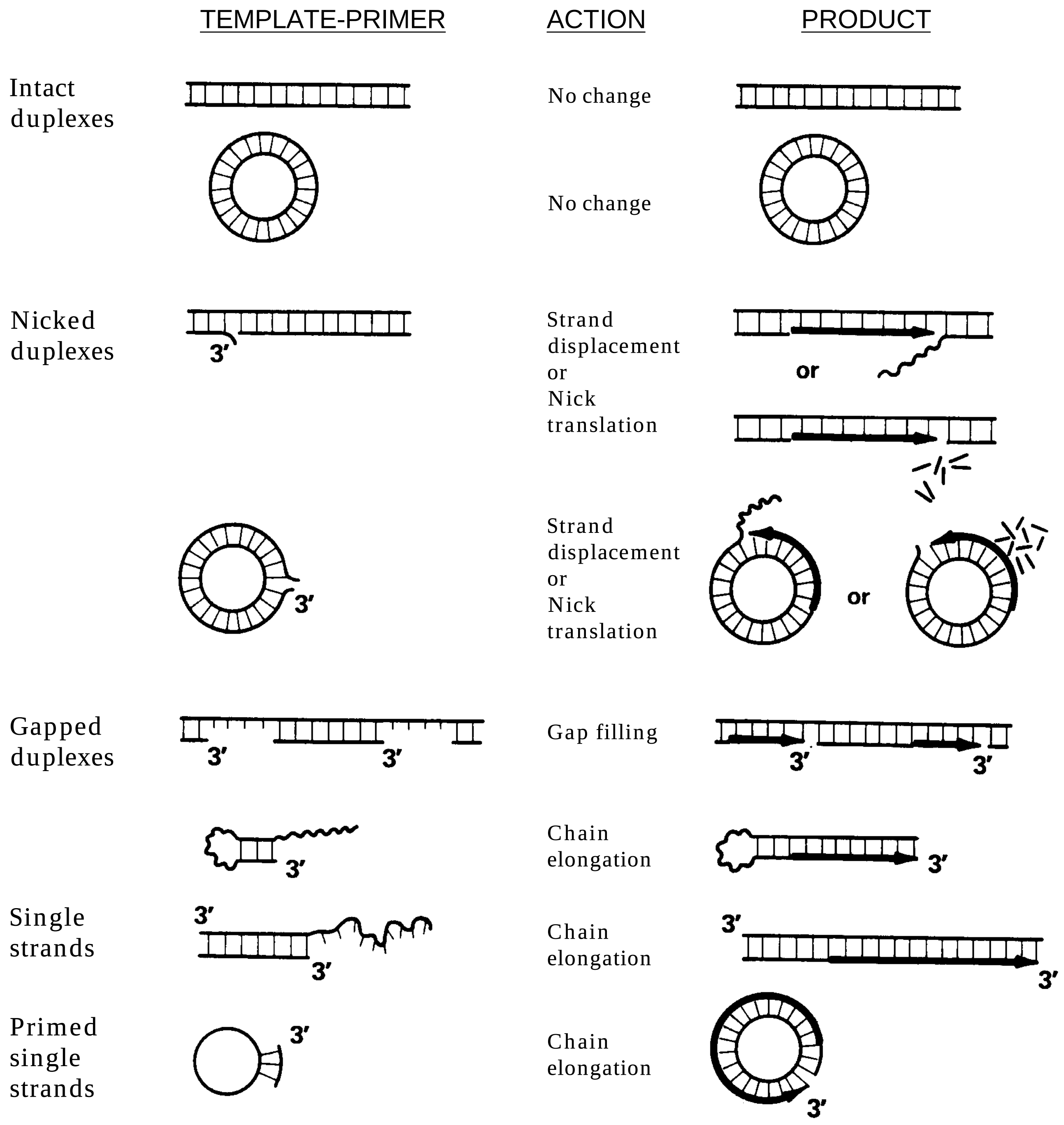

TEMPLATE-PRIMER
ACTION
PRODUCT
Intact duplexes
No change
No change
Nicked duplexes
3′
Strand displacement or Nick translation
or
3′
Strand displacement or Nick translation
or
Gapped duplexes
3′
3′
Gap filling
3′
3′
3′
Chain elongation
3′
Single strands
3′
3′
Chain elongation
3′
3′
Primed single strands
3′
Chain elongation
3′

Figure 1

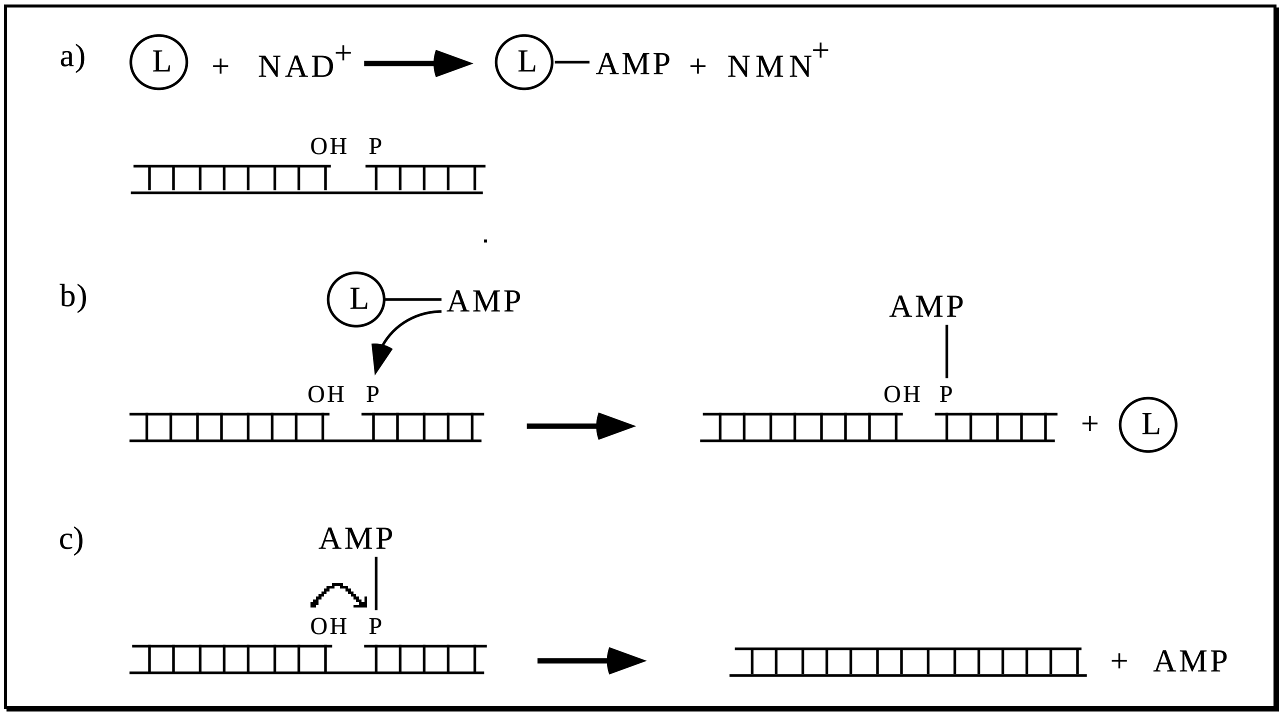


Figure 2

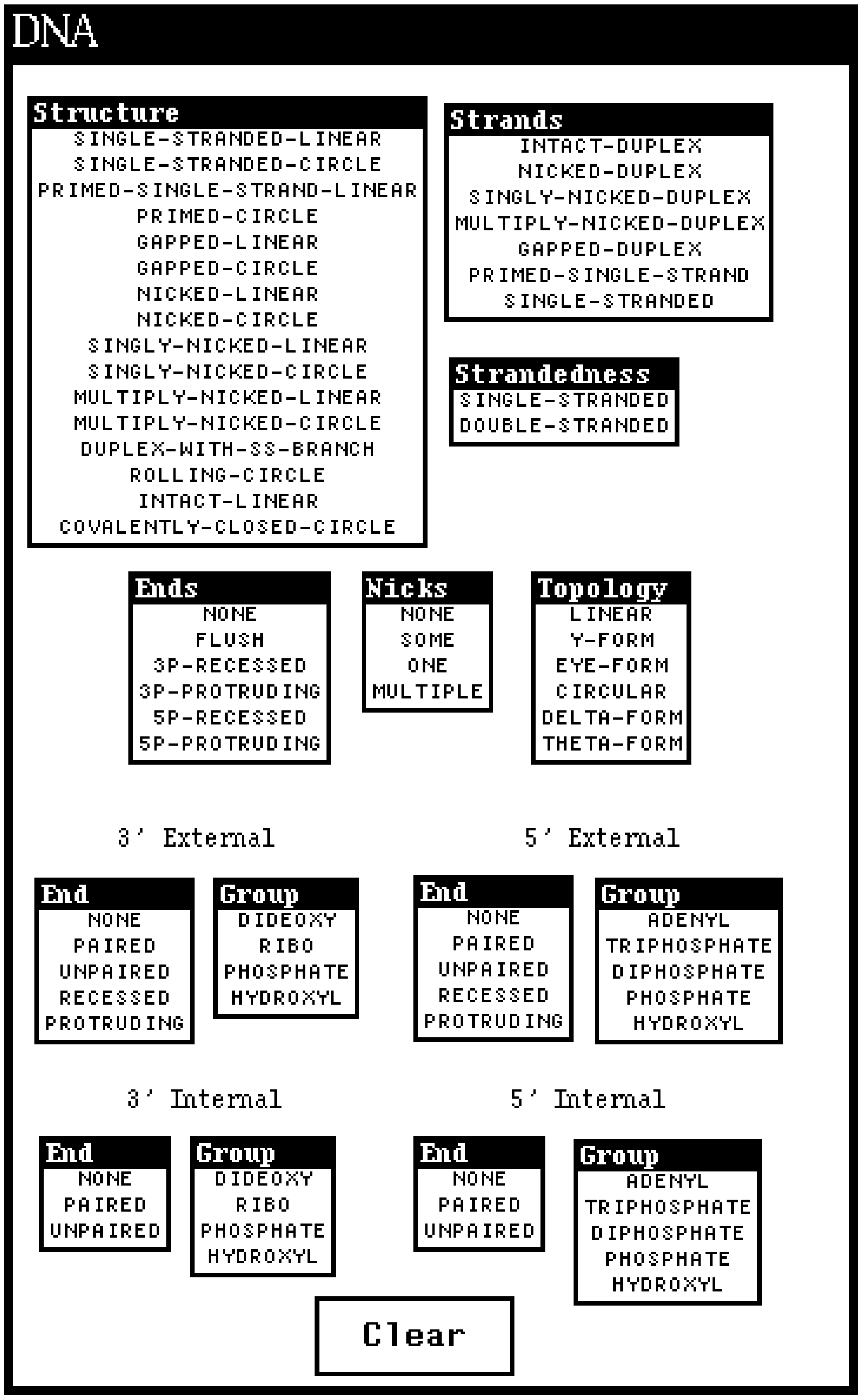
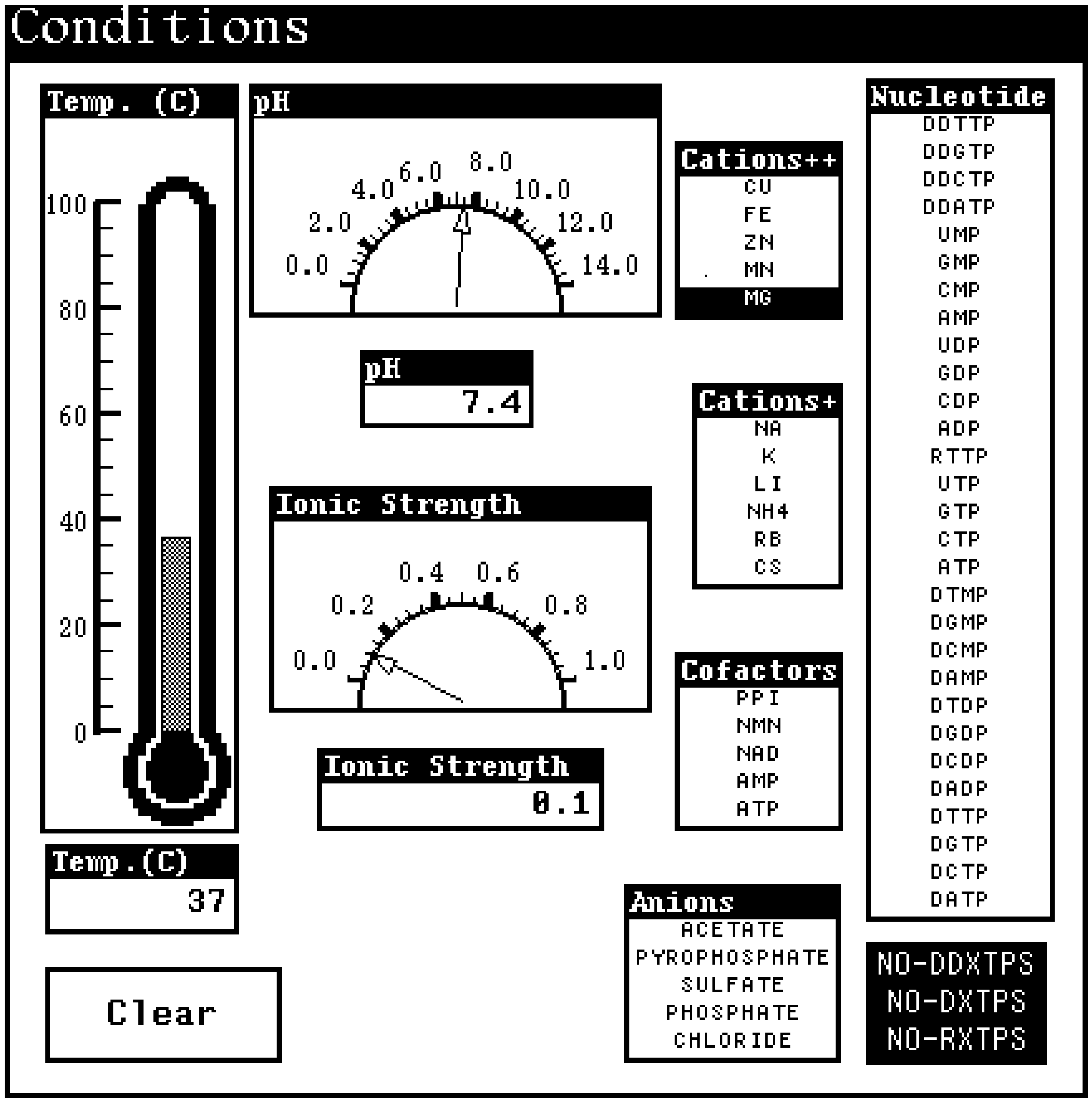

DNA
Structure
SINGLE-STRANDED-LINEAR
SINGLE-STRANDED-CIRCLE
PRIMED-SINGLE-STRAND-LINEAR
PRIMED-CIRCLE
GAPPED-LINEAR
GAPPED-CIRCLE
NICKED-LINEAR
NICKED-CIRCLE
SINGLY-NICKED-LINEAR
SINGLY-NICKED-CIRCLE
MULTIPLY-NICKED-LINEAR
MULTIPLY-NICKED-CIRCLE
DUPLEX-WITH-SS-BRANCH
ROLLING-CIRCLE
INTACT-LINEAR
COVALENTLY-CLOSED-CIRCLE
Strands
INTACT-DUPLEX
NICKED-DUPLEX
SINGLY-NICKED-DUPLEX
MULTIPLY-NICKED-DUPLEX
GAPPED-DUPLEX
PRIMED-SINGLE-STRAND
SINGLE-STRANDED
Strandedness
SINGLE-STRANDED
DOUBLE-STRANDED
Ends
NONE
FLUSH
3P-RECESSED
3P-PROTRUDING
5P-RECESSED
5P-PROTRUDING
Nicks
NONE
SOME
ONE
MULTIPLE
Topology
LINEAR
Y-FORM
EYE-FORM
CIRCULAR
DELTA-FORM
THETA-FORM
3' External
End
NONE
PAIRED
UNPAIRED
RECESSED
PROTRUDING
Group
DIDEOXY
RIBO
PHOSPHATE
HYDROXYL
5' External
End
NONE
PAIRED
UNPAIRED
RECESSED
PROTRUDING
Group
ADENYL
TRIPHOSPHATE
DIPHOSPHATE
PHOSPHATE
HYDROXYL
3' Internal
End
NONE
PAIRED
UNPAIRED
Group
DIDEOXY
RIBO
PHOSPHATE
HYDROXYL
5' Internal
End
NONE
PAIRED
UNPAIRED
Group
ADENYL
TRIPHOSPHATE
DIPHOSPHATE
PHOSPHATE
HYDROXYL
Clear

Figure 3



pH 7.4

Ionic Strength 0.1

Temp.(C) 37

Figure 4

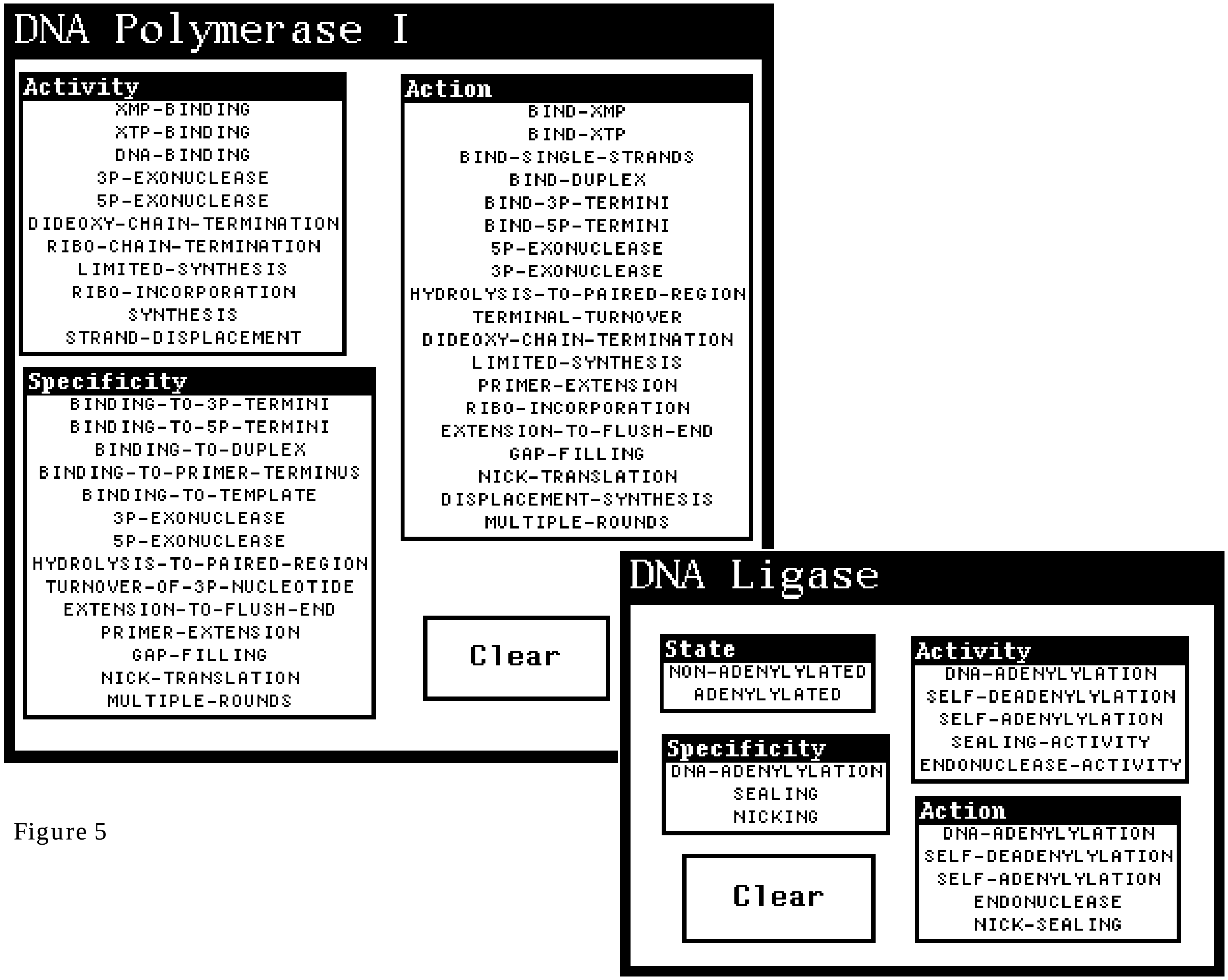

DNA Polymerase I
Activity
XMP-BINDING
XTP-BINDING
DNA-BINDING
3P-EXONUCLEASE
5P-EXONUCLEASE
DIDEOXY-CHAIN-TERMINATION
RIBO-CHAIN-TERMINATION
LIMITED-SYNTHESIS
RIBO-INCORPORATION
SYNTHESIS
STRAND-DISPLACEMENT
Specificity
BINDING-TO-3P-TERMINI
BINDING-TO-5P-TERMINI
BINDING-TO-DUPLEX
BINDING-TO-PRIMER-TERMINUS
BINDING-TO-TEMPLATE
3P-EXONUCLEASE
5P-EXONUCLEASE
HYDROLYSIS-TO-PAIRED-REGION
TURNOVER-OF-3P-NUCLEOTIDE
EXTENSION-TO-FLUSH-END
PRIMER-EXTENSION
GAP-FILLING
NICK-TRANSLATION
MULTIPLE-ROUNDS
Action
BIND-XMP
BIND-XTP
BIND-SINGLE-STRANDS
BIND-DUPLEX
BIND-3P-TERMINI
BIND-5P-TERMINI
5P-EXONUCLEASE
3P-EXONUCLEASE
HYDROLYSIS-TO-PAIRED-REGION
TERMINAL-TURNOVER
DIDEOXY-CHAIN-TERMINATION
LIMITED-SYNTHESIS
PRIMER-EXTENSION
RIBO-INCORPORATION
EXTENSION-TO-FLUSH-END
GAP-FILLING
NICK-TRANSLATION
DISPLACEMENT-SYNTHESIS
MULTIPLE-ROUNDS
Clear
DNA Ligase
State
NON-ADENYLYLATED
ADENYLYLATED
Specificity
DNA-ADENYLYLATION
SEALING
NICKING
Clear
Activity
DNA-ADENYLYLATION
SELF-DEADENYLYLATION
SELF-ADENYLYLATION
SEALING-ACTIVITY
ENDONUCLEASE-ACTIVITY
Action
DNA-ADENYLYLATION
SELF-DEADENYLYLATION
SELF-ADENYLYLATION
ENDONUCLEASE
NICK-SEALING

Figure 5

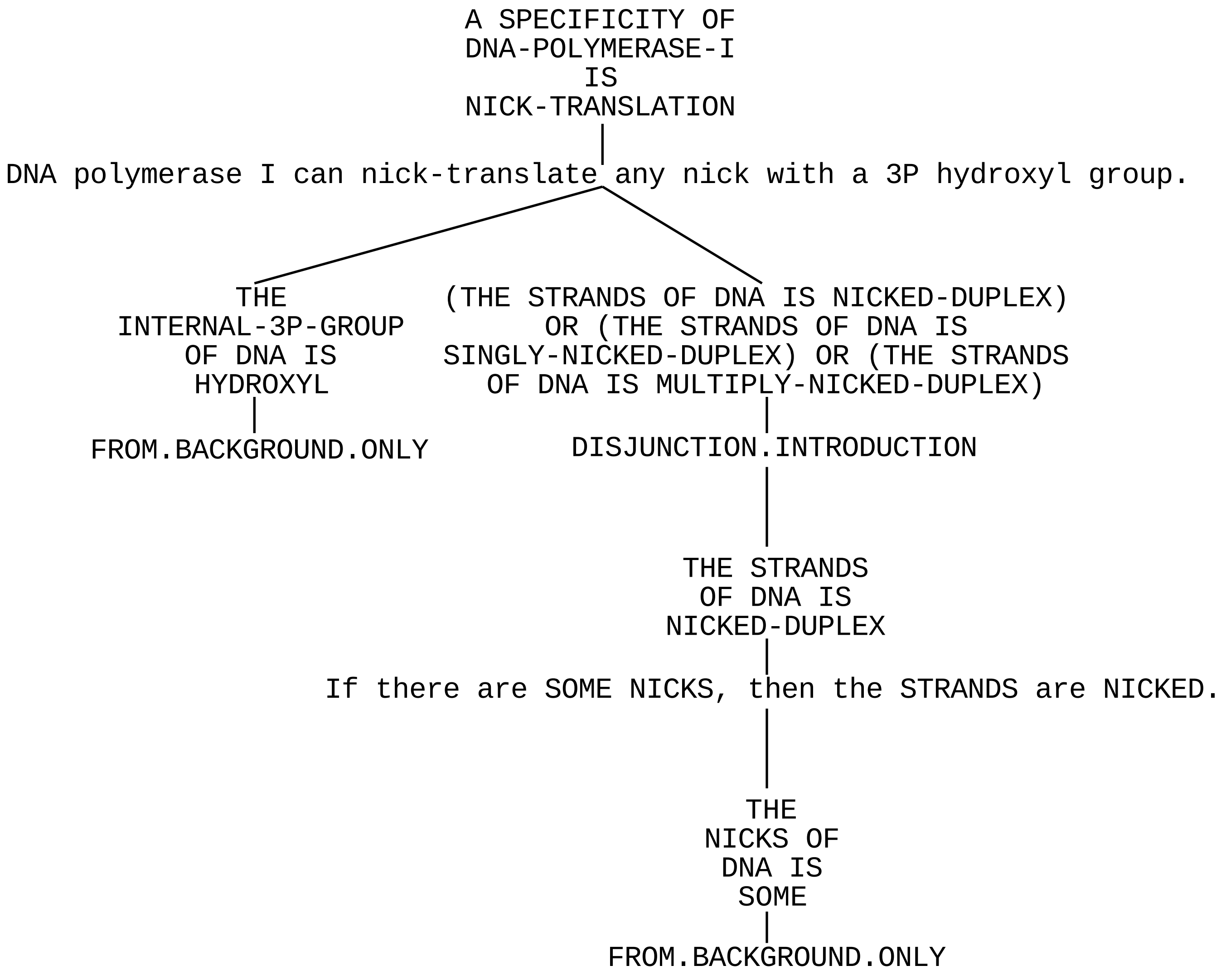
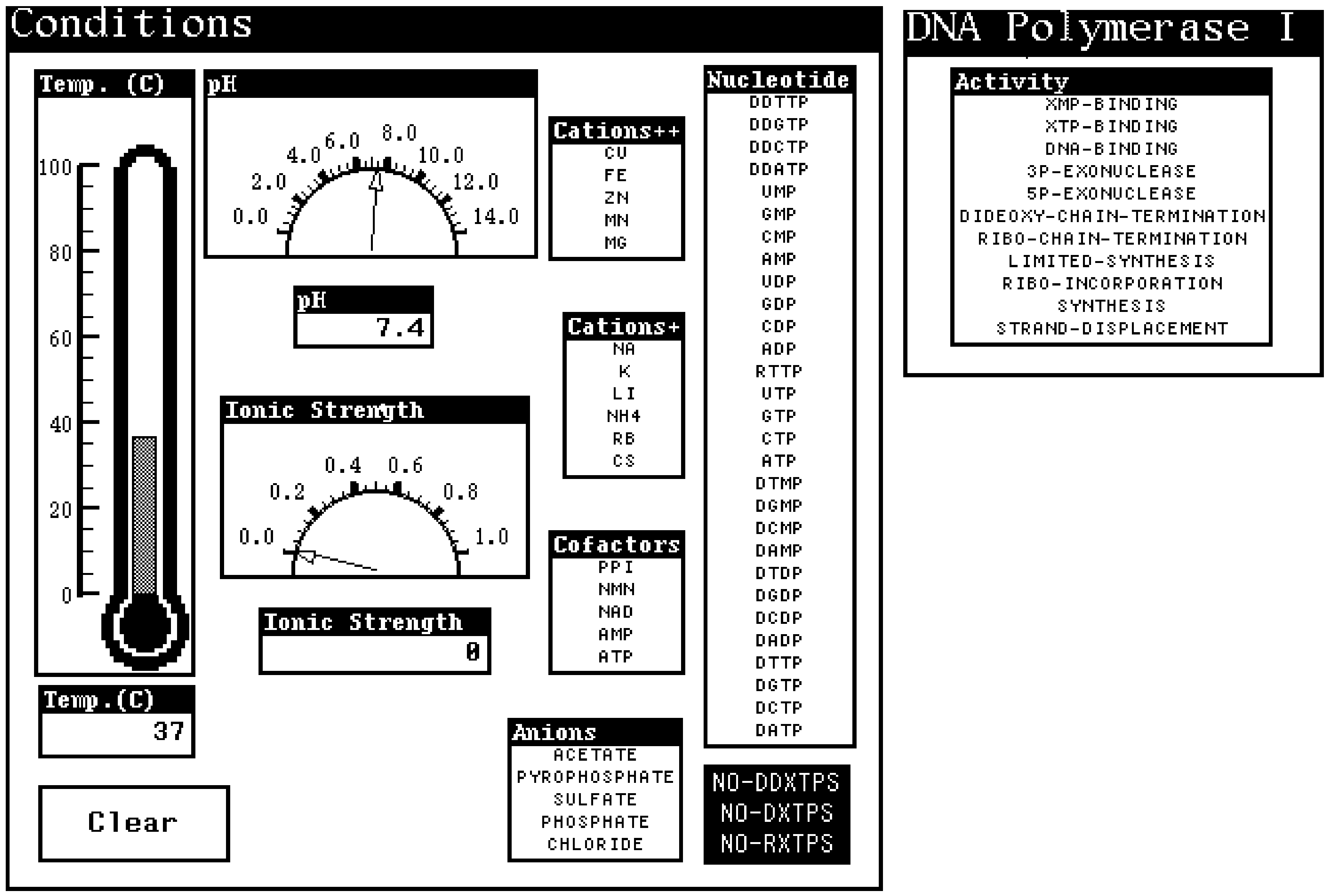

A SPECIFICITY OF DNA-POLYMERASE-I IS NICK-TRANSLATION
DNA polymerase I can nick-translate any nick with a 3P hydroxyl group.
THE INTERNAL-3P-GROUP OF DNA IS HYDROXYL
FROM.BACKGROUND.ONLY
(THE STRANDS OF DNA IS NICKED-DUPLEX) OR (THE STRANDS OF DNA IS SINGLY-NICKED-DUPLEX) OR (THE STRANDS OF DNA IS MULTIPLY-NICKED-DUPLEX)
DISJUNCTION.INTRODUCTION
THE STRANDS OF DNA IS NICKED-DUPLEX
If there are SOME NICKS, then the STRANDS are NICKED.
THE NICKS OF DNA IS SOME
FROM.BACKGROUND.ONLY

Figure 6

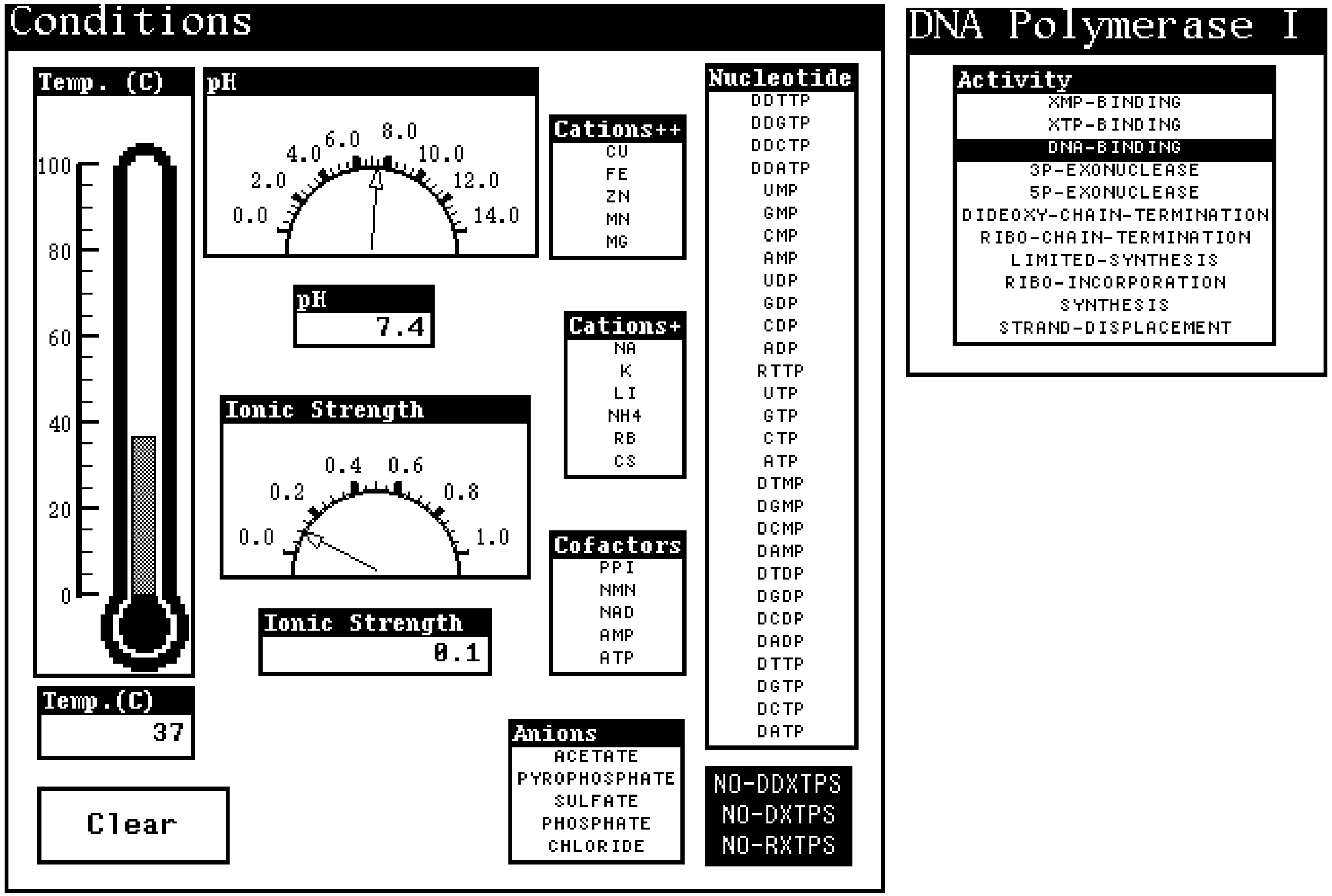


pH 7.4

Ionic Strength 0

Temp.(C) 37

Figure 7



pH 7.4

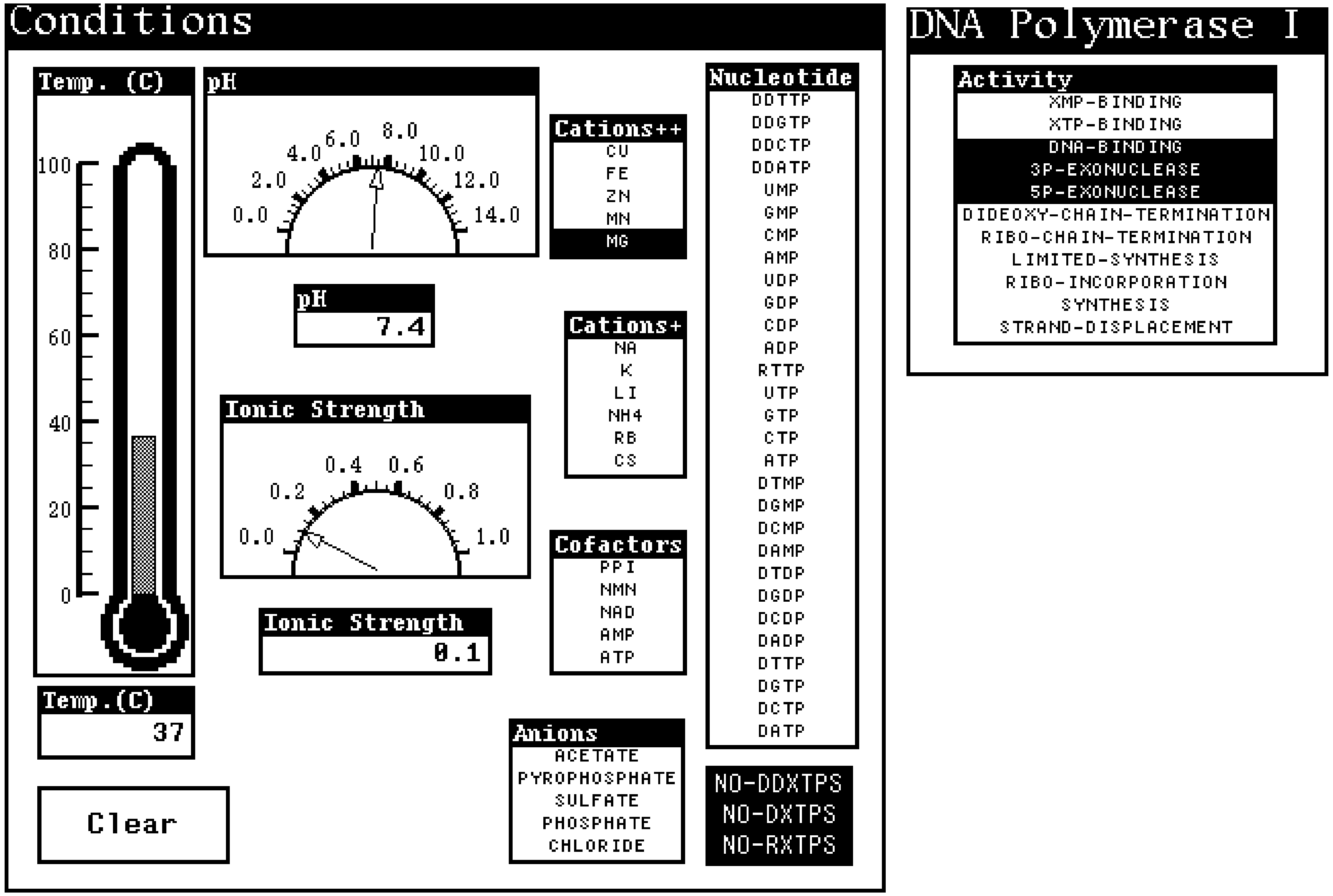
Ionic Strength 0.1

Temp.(C) 37

NO-DDXTPS
NO-DXTPS
NO-RXTPS

DNA-BINDING

Figure 8

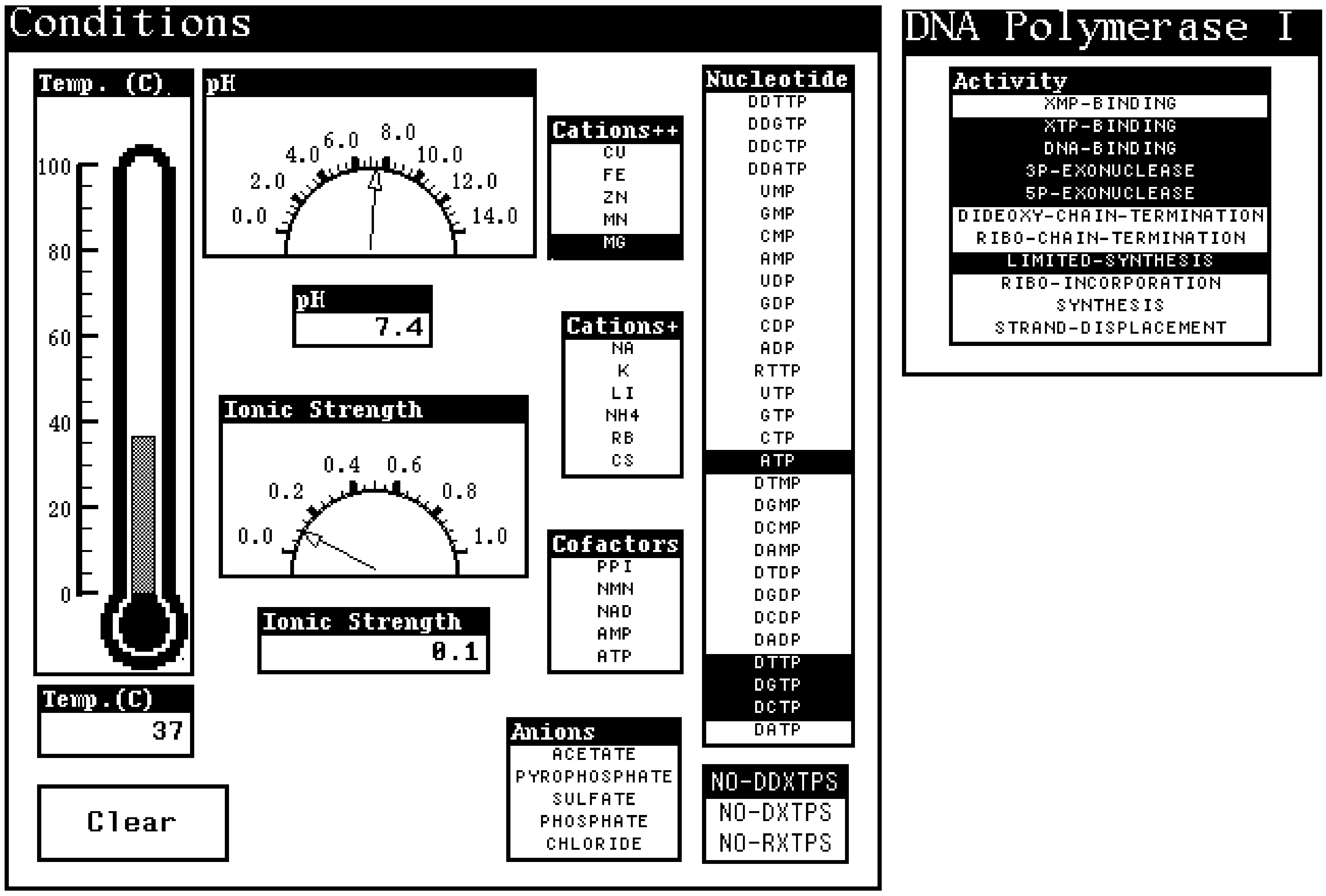


pH 7.4

Ionic Strength 0.1

Temp.(C) 37

Figure 9



Conditions

Temp. (C)

pH

pH 7.4

Ionic Strength

Ionic Strength 0.1

Temp.(C) 37

Clear

Cations++
CU
FE
ZN
MN
MG

Cations+
NA
K
LI
NH4
RB
CS

Cofactors
PPI
NMN
NAD
AMP
ATP

Anions
ACETATE
PYROPHOSPHATE
SULFATE
PHOSPHATE
CHLORIDE

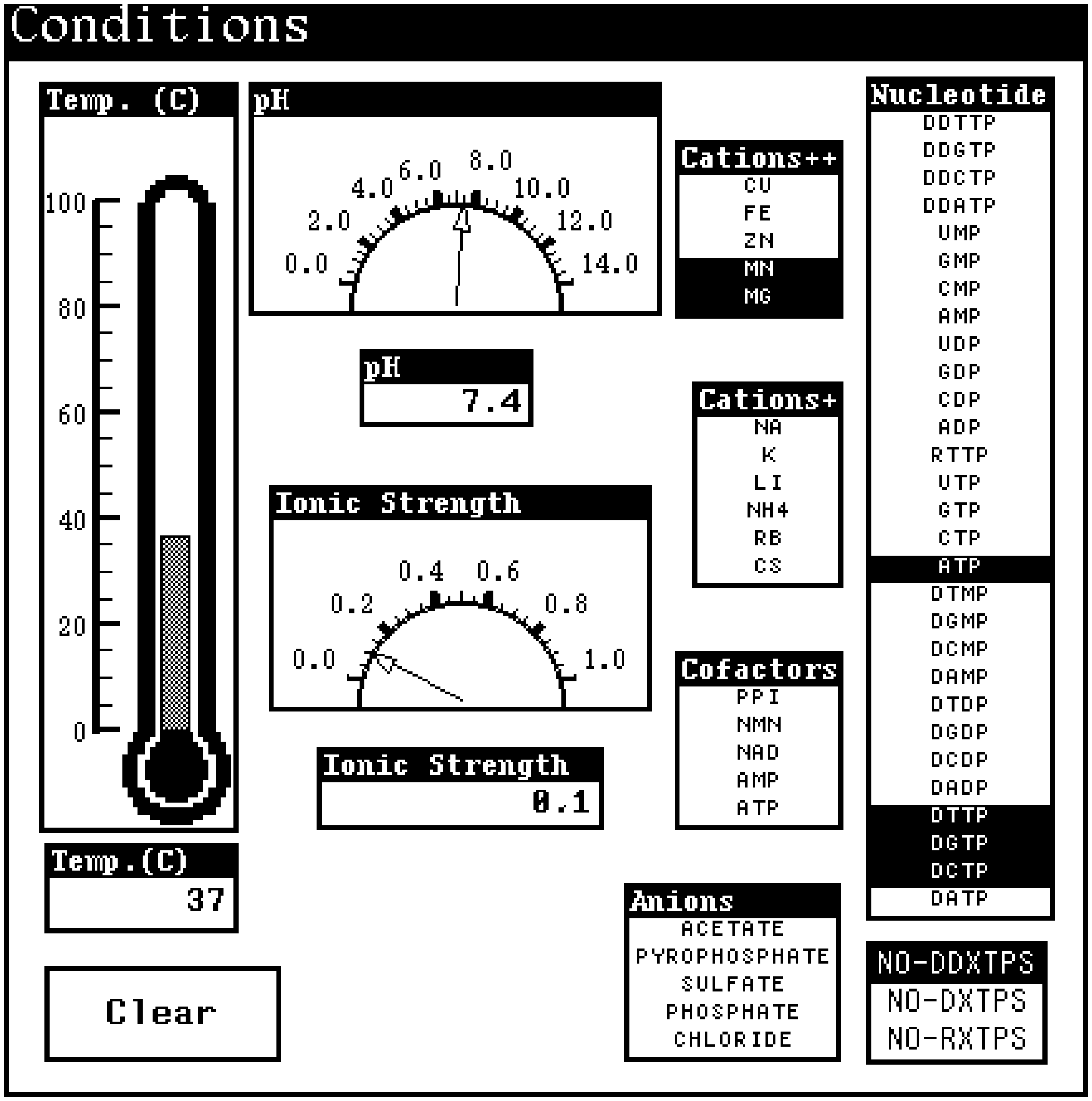
Nucleotide
DDTTP
DDGTP
DDCTP
DDATP
UMP
GMP
CMP
AMP
UDP
GDP
CDP
ADP
RTTP
UTP
GTP
CTP
ATP
DTMP
DGMP
DCMP
DAMP
DTDP
DGDP
DCDP
DADP
DTTP
DGTP
DCTP
DATP

NO-DDXTPS
NO-DXTPS
NO-RXTPS

DNA Polymerase I

Activity
XMP-BINDING
XTP-BINDING
DNA-BINDING
3P-EXONUCLEASE
5P-EXONUCLEASE
DIDEOXY-CHAIN-TERMINATION
RIBO-CHAIN-TERMINATION
LIMITED-SYNTHESIS
RIBO-INCORPORATION
SYNTHESIS
STRAND-DISPLACEMENT

Figure 10



Conditions

Temp. (C)

Temp.(C)
37

Clear

pH

pH
7.4

Ionic Strength

Ionic Strength
0.1

Cations++
CU
FE
ZN
[✓ MN]
[✓ MG]

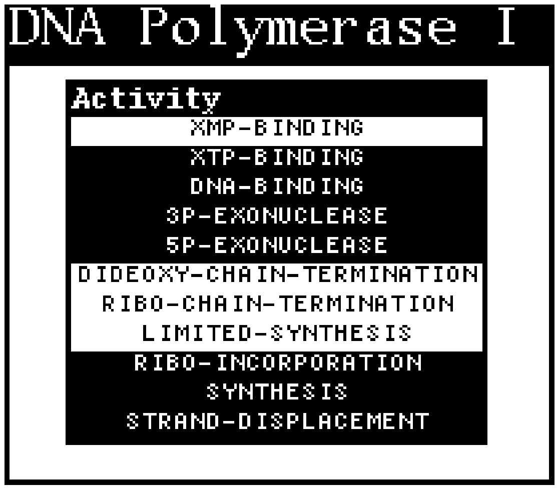
Cations+
NA
K
LI
NH4
RB
CS

Cofactors
PPI
NMN
NAD
AMP
ATP

Anions
ACETATE
PYROPHOSPHATE
SULFATE
PHOSPHATE
CHLORIDE

Nucleotide
DDTTP
DDGTP
DDCTP
DDATP
UMP
GMP
CMP
AMP
UDP
GDP
CDP
ADP
RTTP
UTP
GTP
CTP
[✓ ATP]
DTMP
DGMP
DCMP
DAMP
DTDP
DGDP
DCDP
DADP
[✓ DTTP]
[✓ DGTP]
[✓ DCTP]
DATP

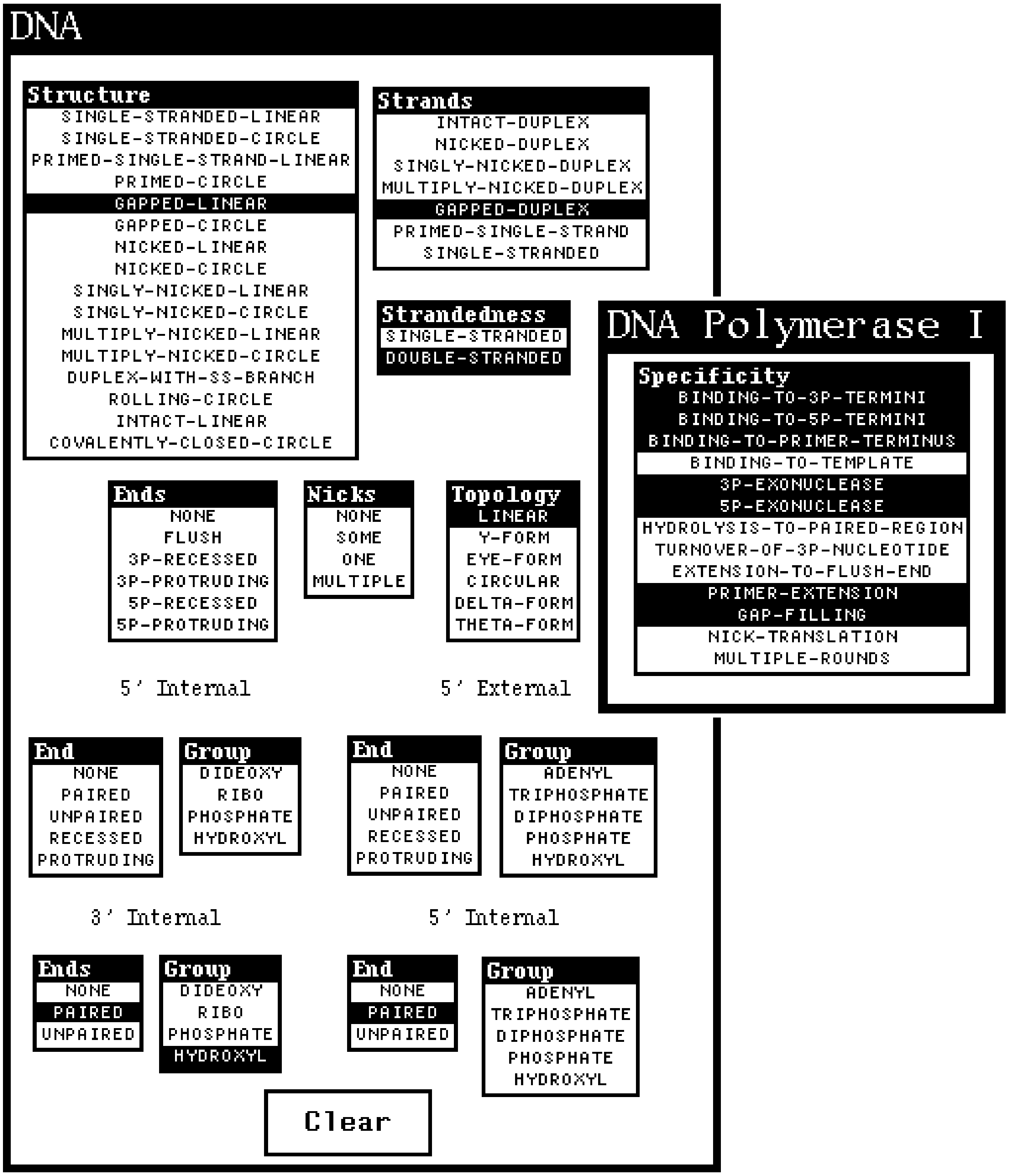
[✓ NO-DDXTPS]
NO-DXTPS
NO-RXTPS

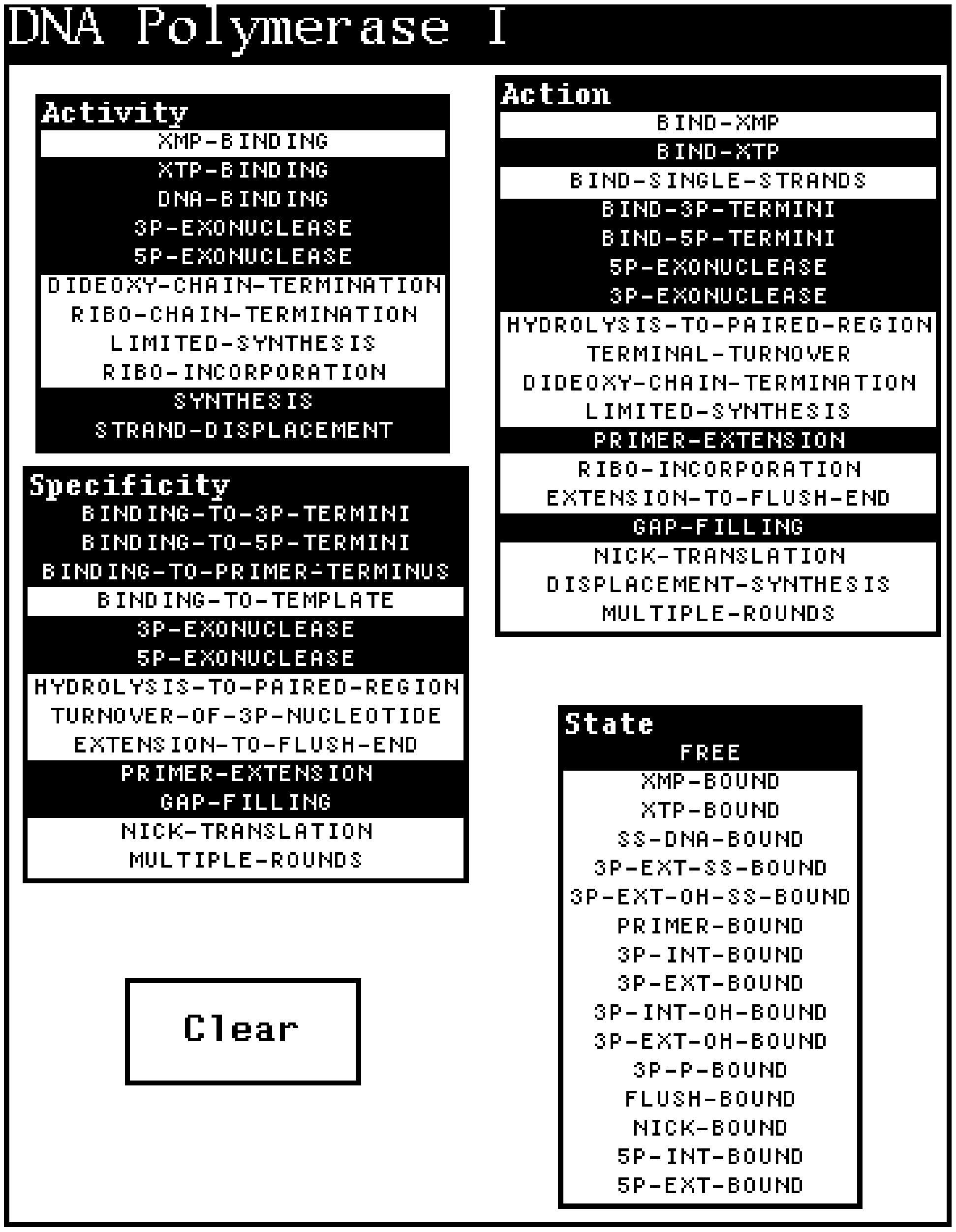


DNA Polymerase I

Activity
XMP-BINDING
[✓ XTP-BINDING]
[✓ DNA-BINDING]
[✓ 3P-EXONUCLEASE]
[✓ 5P-EXONUCLEASE]
DIDEOXY-CHAIN-TERMINATION
RIBO-CHAIN-TERMINATION
LIMITED-SYNTHESIS
[✓ RIBO-INCORPORATION]
[✓ SYNTHESIS]
[✓ STRAND-DISPLACEMENT]

Figure 11


DNA
Structure
SINGLE-STRANDED-LINEAR
SINGLE-STRANDED-CIRCLE
PRIMED-SINGLE-STRAND-LINEAR
PRIMED-CIRCLE
[✓ GAPPED-LINEAR]
GAPPED-CIRCLE
NICKED-LINEAR
NICKED-CIRCLE
SINGLY-NICKED-LINEAR
SINGLY-NICKED-CIRCLE
MULTIPLY-NICKED-LINEAR
MULTIPLY-NICKED-CIRCLE
DUPLEX-WITH-SS-BRANCH
ROLLING-CIRCLE
INTACT-LINEAR
COVALENTLY-CLOSED-CIRCLE
Strands
INTACT-DUPLEX
NICKED-DUPLEX
SINGLY-NICKED-DUPLEX
MULTIPLY-NICKED-DUPLEX
[✓ GAPPED-DUPLEX]
PRIMED-SINGLE-STRAND
SINGLE-STRANDED
Strandedness
SINGLE-STRANDED
[✓ DOUBLE-STRANDED]
Ends
NONE
FLUSH
3P-RECESSED
3P-PROTRUDING
5P-RECESSED
5P-PROTRUDING
Nicks
NONE
SOME
ONE
MULTIPLE
Topology
[✓ LINEAR]
Y-FORM
EYE-FORM
CIRCULAR
DELTA-FORM
THETA-FORM
5' Internal
End
NONE
PAIRED
UNPAIRED
RECESSED
PROTRUDING
Group
DIDEOXY
RIBO
PHOSPHATE
HYDROXYL
5' External
End
NONE
PAIRED
UNPAIRED
RECESSED
PROTRUDING
Group
ADENYL
TRIPHOSPHATE
DIPHOSPHATE
PHOSPHATE
HYDROXYL
3' Internal
Ends
NONE
[✓ PAIRED]
UNPAIRED
Group
DIDEOXY
RIBO
PHOSPHATE
[✓ HYDROXYL]
5' Internal
End
NONE
[✓ PAIRED]
UNPAIRED
Group
ADENYL
TRIPHOSPHATE
DIPHOSPHATE
PHOSPHATE
HYDROXYL
Clear
DNA Polymerase I
Specificity
[✓ BINDING-TO-3P-TERMINI]
[✓ BINDING-TO-5P-TERMINI]
[✓ BINDING-TO-PRIMER-TERMINUS]
BINDING-TO-TEMPLATE
[✓ 3P-EXONUCLEASE]
[✓ 5P-EXONUCLEASE]
HYDROLYSIS-TO-PAIRED-REGION
TURNOVER-OF-3P-NUCLEOTIDE
EXTENSION-TO-FLUSH-END
[✓ PRIMER-EXTENSION]
[✓ GAP-FILLING]
NICK-TRANSLATION
MULTIPLE-ROUNDS
Figure 12



DNA Polymerase I

Activity
XMP-BINDING
XTP-BINDING
DNA-BINDING
3P-EXONUCLEASE
5P-EXONUCLEASE
DIDEOXY-CHAIN-TERMINATION
RIBO-CHAIN-TERMINATION
LIMITED-SYNTHESIS
RIBO-INCORPORATION
SYNTHESIS
STRAND-DISPLACEMENT

Action
BIND-XMP
BIND-XTP
BIND-SINGLE-STRANDS
BIND-3P-TERMINI
BIND-5P-TERMINI
5P-EXONUCLEASE
3P-EXONUCLEASE
HYDROLYSIS-TO-PAIRED-REGION
TERMINAL-TURNOVER
DIDEOXY-CHAIN-TERMINATION
LIMITED-SYNTHESIS
PRIMER-EXTENSION
RIBO-INCORPORATION
EXTENSION-TO-FLUSH-END
GAP-FILLING
NICK-TRANSLATION
DISPLACEMENT-SYNTHESIS
MULTIPLE-ROUNDS

Specificity
BINDING-TO-3P-TERMINI
BINDING-TO-5P-TERMINI
BINDING-TO-PRIMER-TERMINUS
BINDING-TO-TEMPLATE
3P-EXONUCLEASE
5P-EXONUCLEASE
HYDROLYSIS-TO-PAIRED-REGION
TURNOVER-OF-3P-NUCLEOTIDE
EXTENSION-TO-FLUSH-END
PRIMER-EXTENSION
GAP-FILLING
NICK-TRANSLATION
MULTIPLE-ROUNDS

State
FREE
XMP-BOUND
XTP-BOUND
SS-DNA-BOUND
3P-EXT-SS-BOUND
3P-EXT-OH-SS-BOUND
PRIMER-BOUND
3P-INT-BOUND
3P-EXT-BOUND
3P-INT-OH-BOUND
3P-EXT-OH-BOUND
3P-P-BOUND
FLUSH-BOUND
NICK-BOUND
5P-INT-BOUND
5P-EXT-BOUND

Clear

Figure 13