MENDEL: AN INTELLIGENT COMPUTER TUTORING SYSTEM FOR GENETICS PROBLEM-SOLVING, CONJECTURING, AND UNDERSTANDING*

Michael J. Streibel

Educational Technology Program
University of Wisconsin
Madison, Wisconsin

James Stewart

Curriculum and Instruction Department
University of Wisconsin
Madison, Wisconsin

Kenneth Koedinger

Carnegie-Mellon University
Pittsburgh, Pennsylvania

Angelo Collins

Curriculum and Instruction Department
Kansas State University
Manhattan, Kansas

John R. Jungck

Biology Department
Beloit College
Beloit, Wisconsin

*An earlier version of this paper was presented at the 1986 annual meeting of the *American Educational Research Association.*

Machine-Mediated Learning, Volume 2, No. 1 & 2, 129–159, 1987
0732–6718/84/01029–00$3.00/0

Abstract This paper describes an advice-giving computer system for genetics education called the **MENDEL** system that is based on research in learning and genetics problem solving as well as on recent advances in expert systems. The **MENDEL** system is designed to help students gain a better understanding of genetics and scientific inquiry by providing them with the opportunity to solve realistic genetics problems and obtain tutorial assistance that is tailored to their genetics knowledge and level of proficiency at problem-solving. **MENDEL** consists of a problem GENERATOR component and a TUTOR component. The TUTOR includes: a rule-based, expert SOLVER; a problem-solving ADVISOR; a student MODELER; and, a video/graphics LIBRARIAN.

Introduction

There is a growing literature in education and psychology that addresses the need for open-ended problem-solving in science education [1, 2]. There is also an increasing call for the instructional use of microcomputers in science education (as seen in the pages of *The American Biology Teacher* and*The Science Teacher*). Finally, there is an emerging discipline within artificial intelligence research that deals with the design and use of intelligent tutoring systems and advice-giving systems [3, 4]. These trends are converging so that the time is right to bring the theoretical and practical advances within each discipline to bear on the design and use of computers in science education. For example, research in education and psychology has focused on: student alternate conceptions [5-8]; problem-solving [9, 10]; and teaching for conceptual change [11,12]. Research in artificial intelligence, on the other hand, has focused on: the development of knowledge representation schemes (e.g., frames, production rules, semantic networks, etc.), the design of intelligent tutoring systems [13, 3, 14-17], and the instructional potential of intelligent tutoring systems [18-20]. These developments complement and reinforce each other so that educational software can now be based on theories of teaching, learning and problem-solving[13].

For the past several years, we have carried on a research and development effort that has focused on promoting improvements in teaching genetics at the high-school and college levels. This work has entailed the analysis of high school students' knowledge of transmission genetics as well as how their knowledge influences their problem-solving performance [21-23]. More recently, we have been studying the strategies that beginning university students [24], high school students [25] and geneticists [26] use to solve realistic genetics problems generated by a microcomputer.

We have also developed genetics simulation programs [27] that allow students to act like genetics researchers. These programs, called strategic simulations, provide students with the opportunity to develop problem-solving skills and long-range research strategies similar to those used by transmission geneticists [28,29]. Finally, we have been involved with the rapidly developing technology of interactive videodiscs [30] and the critical analyses of the use of computers in education [31].

Drawing on our own interests and research as well as on the recent research on expert systems [3, 32, 33, 17, 34], we are developing an intelligent computer tutoring system called the **MENDEL** system. This system will help students become more knowledgeable problem-solvers.

In this paper, we will describe the logic of the **MENDEL** system as it generates genetics problems and offers tutorial advice to students. The **MENDEL** system is an example of the design approach to science education [35] because it encourages students to develop their understanding of genetics while they conduct experiments and test their hypotheses about genetics mechanisms against the resulting data. This calls for a student to entertain multiple hypotheses, tentatively treat each hypothesis as a conclusion, and construct a set of confirmatory/ disconfirmatory and logical/empirical arguments in support of the final conclusion. The tutorial component stays true to the design flavor of the open-ended problem-solving activity.

Finally the paper ends with a discussion of several larger issues that are involved in the design approach to science education: problem-solving with understanding; problem-based, experiential learning; the integration of rule-based with model-based reasoning; and, the role of human col- laboration in machine-mediated learning environments. The **MENDEL** system described in this paper can be viewed as an experiment in applying the theoretical positions on learning, problem-solving and teaching to the design and use of computer software in education.

A Description of the MENDEL System

The MENDEL *system's goals*

The primary goal of the **MENDEL** system is to provide students with tutorial help to increase their conceptual understanding of genetics as well as their problem-solving skills. This is accomplished by creating a computer environment that will supplement (but not replace) laboratory problem-solving experiences in transmission genetics.

More specifically, the **MENDEL** system has the following goals:

1. to help students develop an understanding of genetics and genetics problem-solving. Students, in turn, will:
   a. improve their problem-solving performance,
   b. gain a better understanding of the conceptual structure of transmission genetics, and,
   c. improve their ability to explain and justify their problem-solving strategies in terms of the conceptual structure of genetics;
2. to help students develop their understanding of scientific research skills such as problem identification, hypothesis generation and testing,

data gathering and long-term inference making.

These two goals are intimately interconnected. They will be elaborated throughout the rest of the paper.

The MENDEL *system's components*

The **MENDEL** system has two primary components:

1. a problem GENERATOR program that includes:
   a. a CUSTOMIZE section, and,
   b. a problem-solving environment;
2. an expert TUTOR program that includes:
   a. a problem SOLVER,
   b. a problem-solving ADVISOR,
   c. a video/graphics LIBRARIAN, and,
   d. a student MODELER.

These components are summarized in Figure 1.

We have completed the GENERATOR program and a prototype of the problem SOLVER component. We are currently working on a prototype of the MODELER and ADVISOR components, and, are working on the design of the video/graphics LIBRARIAN.

Each of **MENDEL**'s components has a unique interface structure. The specific interfaces, however, are integrated into an overall visual interface on the IBM PC-AT screen. For example, each component embodies the following functions in a different way [16]:

1. reduce the working-memory load of a student;
2. aid conceptualization of the genetics content and problem-solving strategies;
3. decompose the problem into manageable subunits, and;
4. help structure the student's thinking.

The overall visual interface, on the other hand, tries to:

1. maintain a consistent command structure;
2. facilitate ease of interaction;
3. be visually-compelling and aesthetically pleasing;
4. be pedagogically sound with respect to the project goals.

The GENERATOR Program in the MENDEL System

The GENERATOR program is termed a "strategic simulation" and places students in a computer environment that simulates the problem-solving situations faced by transmission geneticists in a laboratory [28, 29].

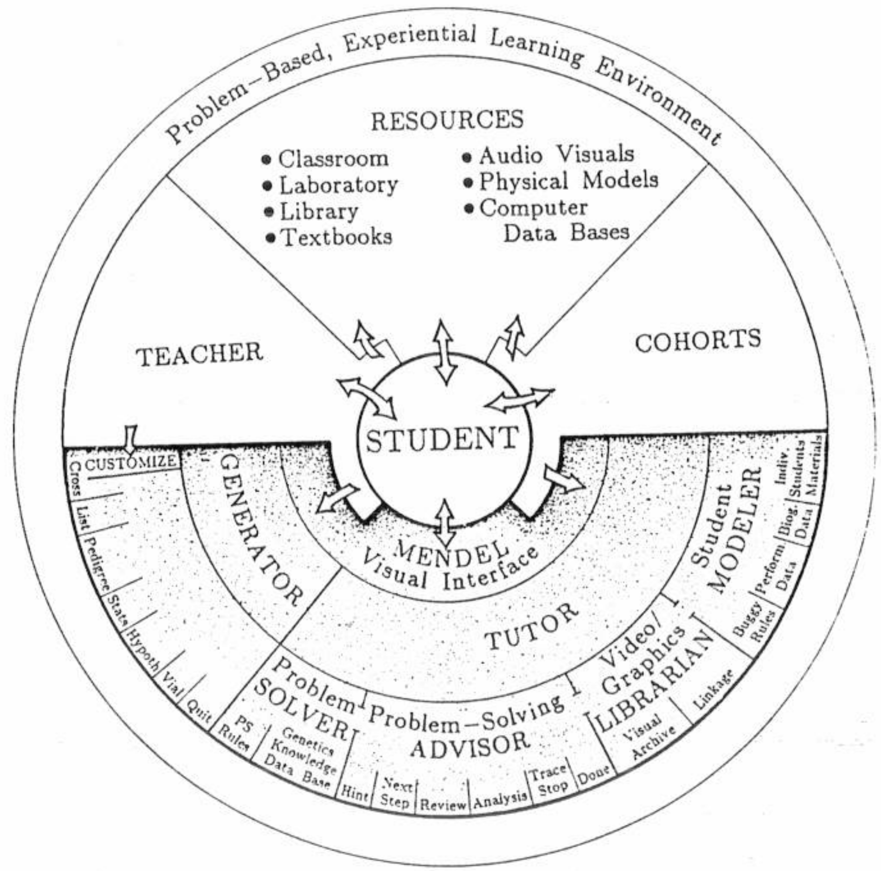


Figure 1. Summary diagram of the **MENDEL** system's components.

Students who use the GENERATOR program have to pose their own problems and then use their genetics knowledge, their ability to perform genetics crosses, and their ability to use computational tools such as CHI square analysis to work out appropriate solutions. The students' experiences with the GENERATOR program are more realistic than those possible with textbook problems.

There are two parts to the GENERATOR program: a *CUSTOMIZE section* where users create classes of problems (within which cases are randomly generated later by the GENERATOR for students) and a *problem-solving environment* where users perform crosses to produce data and use data-management tools to manipulate and view the data (see Figure 1).

The CUSTOMIZE Section of the GENERATOR. Within the CUSTOMIZE section, a user can create *classes of problems* and define *sets of trait and variation names*. Classes of problems are created by filling in templates such as the one shown in Figure 2.

On each of these templates, the user can select the number (1-4) of traits for the problem, the range (1-99) of progeny from a cross and a set of *primary inheritance patterns:* simple dominance (the default value), codominance and multiple alleles. For each problem class, users can set the probability of the appearance of any particular inheritance pattern. In addition, users can select a set of *modifiers* to these primary inheritance patterns: sex linkage, lethality, penetrance, pleiotropy, gene interaction, and autosomal linkage. The modifiers can further be adjusted to set their maximum occurrence and probability of occurrence. For example, in the template shown in Figure 2, two inheritance patterns are possible in the same problem: simple dominance and codominance. Codominance, however, will never appear in more than one trait (since MaxCodom is set at 1) and it might not appear at all (since the CodomProb is set at 60%). These settings, as well as other genetics-specific parameters, permit a user to create a wide range of simple to very complex problems. Thus, the program can be used anywhere from junior high school up through graduate-level genetics.

Trait and variation names are also defined in the CUSTOMIZE section. A sample bodypart template screen for the Antennae trait is shown in Figure 3.

The traits (or Bodyparts) that might appear in any problem are selected along with variation names for that trait. In the sample problem to be discussed in this paper, we will use two body parts as traits: Antennae and Wings. The variables chosen in the CUSTOMIZE section of the GENERATOR "define" the problems that the user encounters in the problem-solving section.

The Problem-solving Environment of the GENERATOR. In the problem-solving section of the GENERATOR program, the student begins with a field-collected vial of organisms on the computer screen and then selects one of several functions. Figure 4 below depicts a "field-collected"


```
CUSTOMIZE
Bodypart #6

Body Part:Antennae___

Fill the following blanks with
adjectives appropriate to this body part

#1:Straight______   #2:Crinkled______   #3:Thread______
#4:Stiff______      #5:Floppy______     #6:Missing______
#7:Tiny______       #8:Arispedia______  #9:Stunted______
#10:Aristaless______ #11:Forked______   #12:Wisp______
#13:Blunt______     #14:Crooked______   #15:Bent______

Is this the last bodypart?N

PRESS A KEY: ESC)when done part   ARROWS)to move around   A-Z/0-9)to fill blanks
```

Figure 2. Sample Menu from the CUSTOMIZE Problem-Definition Screen.

```
CUSTOMIZE Menu Item #1
Enter problem name on the next line:
Simple_Problem____________________

Numtraits     2   MinProgeny 20   MaxProgeny  50
Codominance   Y   Maxcodom   1    CodomProb   60    MaxAlleles 0
MultAlleles   N   MaxMult    0    MProb       0_

Sexlink       N   MaxSexLink0_    SexLinkProb 0_
Linkage       N   HiDistance 0_   LoDistance  0_
Interference  N   HiInt      0_   LoInt       0_
Lethality     N   Maxlethal  0_   LethalProb  0_
Interaction   N   IntProb    0_
Penetrance    N   Maxpen     0_   PProb       0_
                  HMpen      0_   HTpen       0_
Pleiotropy    N   PlProb     0_

Will this be the last menu item? Y

PRESS A KEY: ESC)when finished   ARROWS)to move around   A-Z/0-9)to fill blanks
```

Figure 3. Sample Menu from the CUSTOMIZE Bodypart-Definition Screen.

vial (i.e., Vial#0) whose contents have been elaborated by the List function.

Note that the vials on the computer screen display a shorthand representation of the trait's variation names (e.g., T = "Tiny"). A user can invoke the List option to see the full names of the traits and their variations. In addition, the graphic pedigree diagram on the computer screen represents a redescription of the Vial#0 data into a form that is appropriate for pedigree analysis. In this example, there are 12 females with tiny antennae (i.e., 2 Tiny/Dumpy, 5 Tiny/Lobed, and 5 Tiny/Short). The second variation names (i.e., Dumpy, Lobed and Short) refer to the Wings trait.

Figure 4 also shows some of the functions that are available to students:

C)ross — enables a student to cross individuals and obtain offspring;

L)ist — described above;

P)edigree — represents the vial data in a graphic form and is used by the problem solver to analyze the data produced from a cross experiment. The pedigree diagram is a useful, abstract redescription of cross data that makes it easier to see patterns and thus make inferences about genotypes across generations. The user's hypothesis about genotypes are entered over the question marks (underneath each pedigree box on the screen);

S)tatistics — allows the student to do mathematical calculations and CHI square tests with probabilities;

H)ypotheses — whereas the Pedigree option allows users to make specific hypotheses about parents and offsprings, the Hypotheses command allows users to enter hypotheses about the genetics of the population as a whole;

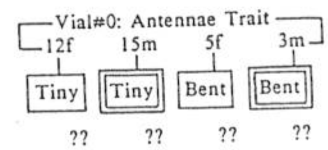
V)ial-options — helps students store and retrieve vials on the screen (for more space on the screen);

Q)uit — allows the student to abandon the current problem before going on.

Students who use the GENERATOR program are faced with an open-ended problem--how to explain the genetic mechanisms responsible for the phenotypes (i.e., appearance) of the population of organisms that

Vials Filled: V0

Vial#0
2 fTD 4 mTD
2 fBS 1 mBS
5 fTL 4 mTL
5 fTS 7 mTS



CONTENTS OF Vial#0

P#	#I	SEX	Antennae	Wings
1	2	f	Tiny	Dumpy
2	4	m	Tiny	Dumpy
3	2	f	Bent	Short
4	1	m	Bent	Short
5	5	f	Tiny	Lobed
6	4	m	Tiny	Lobed
7	5	f	Tiny	Short
8	7	m	Tiny	Short
9	1	f	Bent	Lobed
10	1	m	Bent	Lobed
11	2	f	Bent	Dumpy
12	1	m	Bent	Dumpy

PRESS LETTER: C)ross L)ist P)edigree S)tatistics H)ypotheses V)ials Q)uit

Figure 4. Sample GENERATOR Screen of a Two-Trait Problem with the L)ist Option for Vial#0 (the Parental Vial).

they see on the screen. Underlying the generation of the field-collected vial and all subsequent offspring vials is a model of the inheritance patterns and modifiers as defined in the CUSTOMIZE component of the GENERATOR.

Within the context of the general problem, students are responsible for posing their own specific problems and for selecting the most appropriate approaches to a solution. This is done by performing crosses on the original set of organisms and/or successive generations and by doing statistical analyses. Thus, decisions such as whether enough data has been collected or what the results of statistical tests may mean must be made by students as they develop genetics-specific problem-solving strategies as well as more general scientific inquiry skills.

As rich as the GENERATOR environment is, it does not completely simulate the genetics laboratory experience. Aside from not having to feed, house, and mate actual organisms, students are also not faced with a critical first step in real genetics problem-solving--how to perceptually divide an organism into discrete, analyzable traits. This is already done by the GENERATOR program. Students therefore bypass the initial abstraction processes (of recognition and identification of traits and variations) involved in confronting data in scientific inquiry. In addition, they do not see many of the complex interactions that an organism's genotype (i.e., genetic makeup) has with its environment (both external and internal). These interactions can lead to a wide variation in the phenotype and are only approximated in the GENERATOR's environment. Nonetheless, GENERATOR- created experiences are far richer than the problem-solving experiences in typical undergraduate courses) [29].

The TUTOR program in the MENDEL *system*

The development of the TUTOR program has emerged from a consideration of the roles and responsibilities of a human tutor who is working with students in the GENERATOR environment. For example, a *human* tutor must be able to:

1. make inferences about the data generated by the student problem-solver;
2. maintain a history of a student's actions (including the crosses performed and the statements made about the data and crosses);
3. make inferences about the reasons for the student's problem-solving actions. These are drawn from a combination of what the student has done and has said. In so doing, the human tutor is building a model or representation of each student's or group of students' knowledge of genetics problem-solving;
4. compare the model of a student's knowledge with the tutor's understanding of the problem;
5. make decisions on the form of tutorial advice and the timing of this advice;
6. evaluate whether or not the student has benefitted from the advice.

Our work on the TUTOR component of the **MENDEL** system is guided by, but not necessarily limited to, these roles of a human tutor. Hence, we are developing a computer TUTOR that will be able to:

1. solve genetics problems;
2. interpret data generated by students;
3. develop a model of student knowledge;
4. compare this model with the TUTOR's knowledge;
5. decide whether or not to intervene;
6. decide on the nature of the tutorial intervention;
7. evaluate the success of the tutorial help.

In addition, our TUTOR will provide students with:

1. a set of computational tools for genetics problem-solving (Punnett squares, expression charts, etc.);
2. data-management tools to manipulate the data that they generate (pop-up calculators, data storage and retrieval, etc.);
3. graphical representation of genetics data and conceptual relations (pedigree and chromosome diagrams);
4. multiple windows into the reasoning of the TUTOR.

These last four features are normally *not* available from a human tutor.

The SOLVER Component of the TUTOR. In this section, we will present a simple example from its first appearance on the computer screen to a point where the inheritance pattern of one of the traits has been identified by the SOLVER. This will illustrate the internal logic of the SOLVER insofar as solving a problem is concerned although it will not indicate any tutorial interventions that might occur. This is an example of the TUTOR's TRACE-STOP mode of operation and will only be seen by students when they ask the ADVISOR within the TUTOR to solve an entire problem and explain its actions each step of the way. Because of the stochastic manner in which data is produced by the GENERATOR, two different TRACE-STOPs for the same problem would not be the same.

We begin with the GENERATOR-created screen of a two-trait problem shown in Figure 4. The goal is to infer which inheritance patterns and modifiers account for the distribution of phenotypic data in the population. Several actions can accomplish this goal: generating an hypothesis about a possible inheritance pattern and modifier, generating new data (i.e., invoking the GENERATOR program to perform a cross), checking to see if the data are consistent with the tentative hypothesis, and disconfirming alternate hypotheses. The TUTOR can perform each of these steps on its own because it has a SOLVER component that contains a high-level problem-solving Agenda and specific production rules for solving problems (see Figure 5 below for the SOLVER's Agenda).

This Agenda and related rules were extracted from research on how experts solve similar problems [26] and were formalized as condition/action relations (i.e., IF/THEN production rules). The SOLVER's Agenda items are described below along with a discussion of the example:

1. Redescribe Data from Initial Population for Each Trait: The first step in the Agenda directs the SOLVER to go to the GENERATOR-created population of organisms (see Vial#0 in Figure 4), extract key information (e.g., names and numbers of traits and variations) and store this information in the TUTOR's own internal data structures. It also directs the SOLVER to carry out some simple inferences that can be made from the initial population. For example, by focusing on the first trait (i.e., Antennae), the SOLVER can conclude that there are 12 female organisms and 15 male organisms with tiny Antennae in the initial population. Another example would be that the Antennae trait had only 2 variations (i.e., tiny and bent).

2. Entertain an Hypothesis about Inheritance Pattern: The redescribed data now serves as a set of "conditions" for the Solver's condition/action rules. Hence, the Agenda directs the SOLVER to search through its Hypothesis-Generating Rules (HGR) which in turn "fires" the following rule:

SOLVER AGENDA

1. Redescribe Data from Initial Population for Each Trait

2. Entertain an Hypothesis about Inheritance Pattern
{hypothesis generation rules: HGR}

3. Test Inheritance Pattern Hypothesis:
{find genotype to phenotype mapping}

a. Make a cross {cross rules: CR}
b. Redescribe data from a cross
c. Explain cross in light of hypothesis
{cross explanation rules: CER}
d. Done?
- If there are no consistent explanations, goto 2
- If there is more than one explanations, goto 3
- If there is exactly one explanations, goto 4
- If there is absolutely no explanation, goto 1

4. Check Your Result:

a. Make a prediction to test your hypothesis
b. Are the crosses already performed consistent?
{definitive cross rules: DCR}
c. Disconfirm competing hypothesis
{disconfirmation rules: DR}

Figure 5. Problem-Solving AGENDA for the SOLVER Component of the TUTOR.

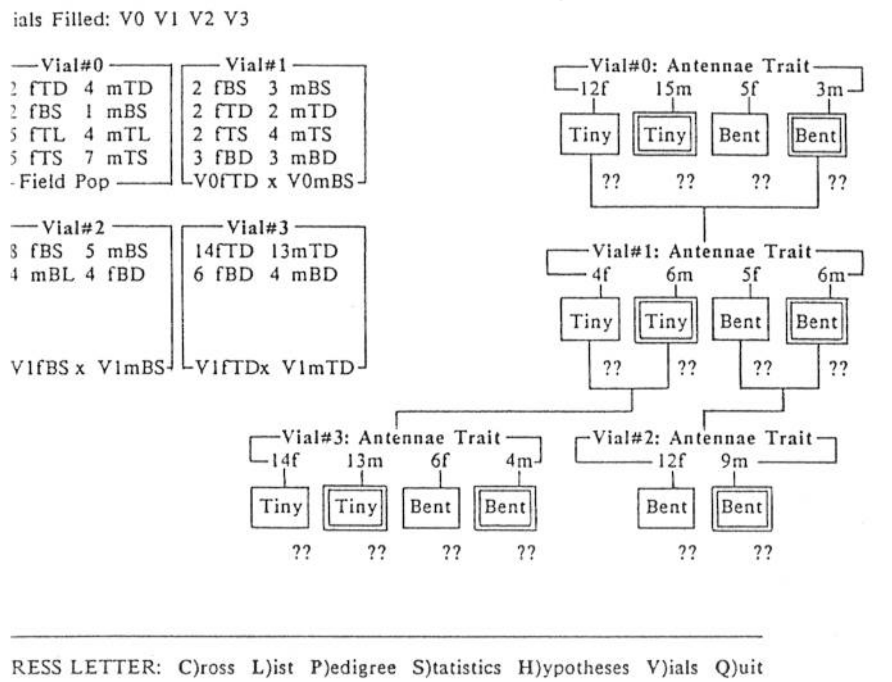
HGR1:	**IF**	(1)	goal: generate an inheritance pattern hypothesis
		(2)	there are 2 variations for a trait
	THEN		assume simple dominance is the inheritance pattern for that trait

HGR1 states that after having broken the larger problem into a sub-problem (i.e., focusing on one trait at a time), the SOLVER should proceed on the assumption that simple dominance may be the inheritance pattern responsible for the phenotypic data. This accomplishes several things. First, it simplifies the search space of possible underlying mechanisms that might account for the phenotypic data. Second, it makes a best first guess at such a mechanism the way an expert problem-solver would do. (Of course, there are several levels of genetics knowledge compiled into HGR1 which would have to be explained to a student who wanted to understand *why* this particular rule was a useful first guess). And finally, it translates a problem-solving strategy into a specific procedure. The SOLVER now has a way to match the phenotypic-level data against genotypic-level causal relationships.

3. Test Inheritance Pattern Hypothesis: The Agenda now directs the SOLVER to cross a female and male organisms from Vial#0. A Cross Rule (CR) fires because the appropriate conditions exist in the redescribed data. This rule directs the GENERATOR program to cross unlike variations (i.e., a tiny-antennaed female with a bent-antennaed male) because such a cross produces the most knowledge about the current hypothesis. (As mentioned above for rule HGR1, Cross Rules contain several levels of genetics knowledge). Hence:

CR2:	**IF**	(1)	goal: plan a cross within a trait
		(2)	there is a variation, V1, for which you don't have a genotype
	THEN		cross unlikes: V1 with some other variation

The SOLVER also tells the GENERATOR to randomly choose one of the 12 female tiny-antennaed organisms and one of the 3 male bent-antennaed organisms. The resulting offsprings are placed in Vial#1. Figure 6 shows the computer screen at the end of the problem-solving session. For the time being, we need only focus on Vial#0 and Vial#1.



igure 6. Sample GENERATOR Screen of a Two-Trait Problem Solved for the ntennae Trait.

The data in Vial#1 represent a new set of conditions for the SOLVER's rules to consider. Following the Agenda (see Figure 5), the SOLVER first redescribes the new data (Agenda item 3b) and then applies a series of Cross Explanation Rules (CER) (Agenda item 3c). One rule fires because the appropriate conditions in Vial#0 *and* Vial#1 exist. Hence:

CER6: IF
(1) goal: explain a cross within a trait
(2) assumed inheritance pattern is simple dominance for that trait
(3) parents are of different variations
(4) offspring are of both variations

THEN
(1) one parent and the offspring of the same variation are homozygous recessive
(2) the other parent and the offspring with this variation are heterozygous dominant

That is, the SOLVER finds that "unlikes" in the parents (tiny-antennaed and bent-antennaed) have produced "unlikes" in the offspring. If simple dominance was in fact the underlying mechanism in our example, the cross could be explained by the abstract genotypic pattern:

$$Aa \times aa \rightarrow 1/2^{Aa} + 1/2^{aa}$$

The capital "A" in the genotypic pattern above represents the dominant allele and the lower-case "a" represents the recessive allele. The "Aa" represents a heterozygous allele-pair and "aa" a homozygous recessive allele-pair. Figure 7 summarizes all of the possible genotype-to-phenotype matches for the simple dominance case.

Of course, the SOLVER cannot at this point determine which specific genotype (i.e., Aa or aa) corresponds with which phenotype (i.e., tiny-antennaed or bent-antennaed) in Vial#1. The SOLVER therefore has to perform more crosses to establish such a correspondence.

At this point, the SOLVER continues to test the current inheritance pattern hypothesis (Agenda item 3d) because Vial#1 has added new conditions for the original set of Cross Rules. Hence, the following Cross Rule fires:

CR16: IF
(1) goal: identify which of the offspring of an unlike cross are heterozygotes
(2) there are two variations in that offspring:

THEN
consider crossing likes from this offspring.

The SOLVER therefore crosses two organisms of the same variation (i.e., bent-antennaed) from Vial#1. The results of the GENERATOR-created data are stored in Vial#2 (See Figure 6). Note that

enotypic Level	Phenotypic Level	Cross Types	Number of Offspring Classes
AA x AA -> AA*	V1 x V1 -> V1**	likes	1
aa x aa -> aa	"	"	"
AA x Aa -> $_{1/2}$AA + $_{1/2}$Aa	'	"	"
Aa x Aa -> $_{1/4}$AA + $_{1/2}$Aa + $_{1/4}$aa	V1 x V1 -> $_{3/4}$V1 + $_{1/4}$V2	likes	2
AA x aa -> Aa	V1 x V2 -> V1	unlikes	1
Aa x aa -> $_{1/2}$Aa + $_{1/2}$aa	V1 x V2 -> $_{1/2}$V1 + $_{1/2}$V2	unlikes	2

'A" represents the dominant allele, "a" the recessive allele.
"AA" represents the homozygous dominant allele-pair.
"aa" represents the homozygous recessive allele-pair.
"Aa" represents the heterozygous allele-pair.

*"V1" represents the first arbitrary variation, Notice that several genotypic patterns can underlie the same phenotypic pattern.

igure 7. Relationship of Genotypic to Phenotypic Data for a Simple Dominance ase of Two Variations (V1 and V2) of One Trait (All Possibilities are Shown).

the SOLVER is now reasoning about the data from several generations of data. This strategy was chosen because it approximates optimal problem-solving performance--something that was not always displayed by the experts [26]. The SOLVER now redescribes the data in Vial#2 and tries to explain the data in light of the simple dominance hypothesis. A Cross Explanation rule fires because the SOLVER has found the correct conditions in both Vial#1 and Vial#2. Hence:

CER7: IF
(1) goal: explain a cross within a trait
(2) assumed inheritance pattern for that trait is simple dominance
(3) parents have like variations within this trait.
(4) parents are either heterozygous or homozygous- recessive
(5) offspring have the same variation within this trait as the parents

THEN parents are very likely homozygous-recessive while offspring are also very likely homozygous-recessive

CER7 helps the SOLVER conclude that the bent variation of the Antennae trait in Vial#2 is due to a homozygous recessive allele-pair. The reasoning proceeds as follows: the SOLVER has already established from the previous cross that the tiny-antennaed and bent-antennaed variations in Vial#1 are *not* due to a homozygous dominant genotype (i.e., the genotypic pattern

Aa x aa -> 1/2^{Aa} + 1/2^{aa}

accounted for the data--thus excluding AA). Of the three simple dominance mechanisms that could account for the appearance of a bent-antennaed phenotype data in Vial#2:

AA x AA -> AA

AA x Aa -> 1/2^{AA} + 1/2^{Aa} (both *appear* the same)

aa x aa -> aa

the first and second genotype patterns can be eliminated because both involve a homozygous dominant genotype. This leaves the homozygous recessive genotype pattern (i.e., aa x aa -> aa) to account for the data in Vial#2. By inference, the SOLVER can also conclude that the tiny-antennaed variation in Vial#0 is due to a heterozygous allele-pair (Aa) because that was the only other pair left in Vial#1. (The SOLVER fills in these hypotheses in the pedigree diagram in place of the question marks ɔelow the pedigree boxes on the screen for the benefit of the student.) At his point, the problem seems to be solved. However, there is one more step in the Agenda.

4. <u>Check Your Result:</u> The SOLVER has accounted for both variations of the Antennae trait in Vial#1 on the assumption that simple dominance was the case. The Agenda therefore directs the SOLVER to carry out one more step: checking the SOLVER's conclusion with an *independent* cross. Collins [26] has found that expert geneticists add a definitive cross of two heterozygous individuals at this point in the process. Hence, the SOLVER applies its Definitive Cross Rules (DCR) and fires the following rule:

DCR1:	IF	(1)	goal: become more confident in an inheritance pattern for a trait
		(2)	assumed inheritance pattern is simple dominance with a high degree of confidence
		(3)	heterozygotes have been identified
	THEN		cross the heterozygous individuals

This rule takes a previously-identified heterozygous individual from Vial#1 (i.e., tiny-antennaed), crosses a male and a female with this variation, and places the results in Vial#3 (See Figure 6). Again, because new data has been generated, new conditions exist for the application of the Cross Explanation Rules. This time, CER8 fires:

CER8:	IF	(1)	goal: explain a cross within a trait
		(2)	assumed inheritance pattern for that trait is simple dominance
		(3)	parents are heterozygous within this trait
		(4)	both traits are present within the offspring
		(5)	test comparing the ratios of offspring variations to 3:1 is significant
	THEN	(1)	increase confidence in identity of parents as heterozygous
		(2)	increase confidence in simple dominance as the inheritance pattern
		(3)	increase confidence that the parent's variation is dominant

This rule confirms that the tiny variation of the Antennae trait could only have come from a heterozygous allele-pair because only one simple dominance rule could account for this data:

$$Aa \times Aa \rightarrow 1/4^{AA} + 1/2^{Aa} + 1/4^{aa}$$

Notice that both AA and Aa show up as the same phenotypic variation

in the offspring because the allele "A" is dominant to the recessive allele "a". Hence, a 3 to 1 ratio for phenotype characteristics is expected to show up in the offsprings (i.e., $3/4^{A-} + 1/4^{aa}$).

Notice also that, although we have confirmed the simple dominance hypothesis for this set of data, there still exists the slightest possibility that some other inheritance pattern and/or modifiers could account for the data. Most genetics experts in such a situation eliminate (or disconfirm) these possibilities with some standard disconfirming crosses[26]. Hence, the Agenda (Item 4c) directs the SOLVER to try out some final Disconfirming Rules (DR) such as:

DR1:	IF	(1)	goal: disconfirm alternate hypotheses
		(2)	inheritance pattern is simple dominance
		(3)	sex-linkage is modifier under consideration
		(4)	a cross of a dominant male with a recessive female results in offsprings that are not limited to dominant females and recessive males
	THEN		sex-linkage modifier is not operating

The example discussed above illustrates the SOLVER's rule-based approach to generating hypotheses about inheritance patterns and to generating crosses within the constraints of these hypotheses. The example shows how rules are used for confirming and disconfirming hypotheses based upon the phenotypic data that emerge after each new cross. The SOLVER therefore has the ability to keep track of its own inferences and the ability to build up genetics knowledge appropriate to a given population of organisms. The TUTOR will have access to all of this information and can use it to provide tutorial advice.

Finally, the SOLVER, when solving problems on its own, performs all aspects of problem-solving. However, in the typical case, the SOLVER will not be making crosses. Rather, it will be suggesting crosses in light of certain student-chosen hypotheses and making inferences from student-generated data. In the latter case, the SOLVER works with the crosses that the student has made and then tries to extract as much knowledge as possible from this data in light of hypotheses that the *student* is entertaining.

The ADVISOR Component of the TUTOR: In the section above on the SOLVER, we described the user-requested TRACE-STOP mode of the ADVISOR. In addition to the TRACE-STOP mode, we will provide the student with other tutorial aids: HINT, NEXT-STEP, REVIEW, and ANALYSIS. Each of these commands can be categorized on two dimensions: one dimension deals with suggestions about a future action (HINT and NEXT-STEP) or an evaluation of past actions (REVIEW and ANALYSIS); the other dimension deals with specific actions (NEXT-STEP and ANALYSIS) or general strategies (HINT and REVIEW). These elationships are shown in Figure 8.

	General Advice (series of actions)	Specific Advice (single action)
Future Actions (SOLVER Data & Hypothesis)	HINT	NEXT-STEP
Past Actions (Student Data & Hypotheses)	REVIEW	ANALYSIS

Other ADVISOR commands include the TRACE-STOP and DONE options.

igure 8. User-Requested Tutorial Options of the ADVISOR Component of the UTOR (Other ADVISOR commands include the TRACE-STOP and DONE options.)

Although we feel it is important for the ADVISOR to have the ability to decide when it is appropriate to offer advice (i.e., to have some TUTOR-initiated intervention strategy), we are currently focussing on what that advice will be. We have made a deliberate decision to implement the user-initiated advice-giving capabilities of the ADVISOR prior to and independently from the intervention strategy. This approach has many advantages. First, by having the student decide when he or she would like advice, we can have a workable tutor before actually implementing a TUTOR-initiated intervention strategy. Second, it is easier to add a more sophisticated intervention strategy to an existing advice-giving capability than it is to design both features at the same time. Finally, by implementing these capabilities independently, we can study the effectiveness of alternative intervention strategies (i.e., user-initiated vs. mixed-initiative interventions) before implementing any one.

We will now describe the user-initiated advice-giving capabilities of the ADVISOR:

1. The HINT Command of the ADVISOR: Students invoke the HINT option when they want a suggestion for what to do next. The ADVISOR then gives them general prompts, and, if that advice is not helpful, gives them increasingly specific hints. Even though HINT provides suggestions about future actions, these suggestions may make little sense to a student if there is something seriously wrong with what he or she has already done. In this case, the ADVISOR will comment on the error before providing a hint. If there is nothing seriously wrong, HINTs will be given that are appropriate to one of the following categories of action: performing crosses (via the Cross command); making hypotheses about individual or offspring class genotypes (via the Pedigree command); or making hypotheses about the genetics of the population as a whole (via the Hypotheses command). For example, if the SOLVER determines that it is possible to make a hypothesis about the genetics of the population, then the hints given to the student might proceed from general to specific as follows:

a. Hints to try to generate a hypothesis. For example: "Can you make any hypotheses? If so, please enter them."

b. Global redescription hints to help a student generate an inheritance pattern hypothesis. These include:
"What can you tell me about the initial population?"
  "How many traits? What are they?"
  "How many variations in each trait? What are they?"
"Have you done other problems with the same number of variations?"
  "What does the number of variations suggest to you?"
  "What if there were 3 variations instead of 2?"

c. Hypothesis generating hints (corresponding to HGR rules).

2. The NEXT-STEP Command of the ADVISOR: The NEXT-STEP command spells out exactly what the TUTOR's SOLVER would do next in

ight of the student's current cross data and hypothesis. There are two possible next steps: *perform a cross* and *state an hypothesis*. When a student receives NEXT-STEP advice, he or she can ask why that advice was given by using the WHY command. In response to WHY, the rule that prompted the specific action is given. If the student seeks further explanation of this rule, the ADVISOR may offer [14].

a. *strategy explanations*, which the student requests by the CLARFY command, and
b. *support explanations*, which the student requests by the JUSTIFY command.

Strategy explanations are designed to clarify the rule by explaining it in terms of more general strategies applicable to many classes of genetics problems. Support explanations employ content knowledge and examples to justify the rule by describing or illustrating the genetic mechanisms underlying the rule.

For example, a student may have crossed Vial#0 individuals with the same phenotypes six times while indicating a current hypothesis of simple dominance. If the NEXT-STEP command is now invoked, the ADVISOR would recommend that the student use some of the offspring that have been produced and make a cross of individuals with unlike variations. If the student invokes the WHY command, the ADVISOR would present Cross Rule 2 (which was used earlier to illustrate the SOLVER's rules). If the student then invoked the CLARIFY command, the ADVISOR would offer a more general strategic explanation (e.g. that crossing unlikes makes it possible for a solver to either construct or identify heterozygous individuals). If the student still wasn't satisfied he or she could invoke CLARIFY again and get explanations of a more general nature, such as:

a. to match phenotypes with genotypes requires the identification of heterozygous individuals,
b. to test inheritance pattern hypotheses requires that all phenotypic variations be matched with genotypes, and,
c. one action in the solving strategy is to Test Inheritance Pattern Hypotheses (Figure 5, Agenda Item 3).

The purpose of CLARIFY is to help the student understand the specific advice provided by the NEXT-STEP command.

The student might also invoke the JUSTIFY command. CR2 relies on the empirical associations of the genotype-to-phenotype relationships illustrated in Figure 7. The tutor might justify crossing unlikes at this point in the problem-solving process by highlighting relationships 5 and 6--that when the variations of the parents are unlike, heterozygous offspring are produced. The next level of explanation would employ relationship 4 to illustrate how crossing parents with like variations can be used to match genotypes with phenotypes.

3. The REVIEW Command of the ADVISOR: The REVIEW command uses data from the student MODELER and possible student errors to look back over the student's performance and make appropriate comments. REVIEW is like ANALYSIS (described below) in that it looks back at student actions. However, REVIEW does a more general evaluation

based on student behaviors spanning the entire problem solution up to the point when a student asks for a REVIEW. REVIEW will make general comments about the student's strategy such as "You didn't use offspring as parents very often". Comments like this can be helpful to a student in future problem-solving sessions.

4. The ANALYSIS Command of the ADVISOR: Whereas the TRACE-STOP command walks students through a solution of crosses that were generated by the SOLVER, the ANALYSIS command walks students through the crosses that *they* made and points out what knowledge the SOLVER can extract from each cross. The ANALYSIS option then debriefs students about the potential significance that each cross had for the problem-solving process and where students may have made one or more of three types of errors: an inconsistent hypothesis, an unwarranted inference, or missed a warranted inference.

5. The DONE Command of the ADVISOR: The student invokes the DONE command when the problem is finished. The ADVISOR will then:

a. check the student's solution for consistency and point out inconsistencies,
b. check the student's solution for completeness and make comments about incompleteness,
c. allow the student to return to the problem-solving environment if they would like to continue working,
d. ask the student if they would like a REVIEW or an ANALYSIS.

The Video/Graphics LIBRARIAN Component of the TUTOR

The video/graphics LIBRARIAN manages both computer-generated graphics and visuals stored on a video disk. Each type of graphics information is accessible to the TUTOR when a decision has been made that a student would benefit from tutorial advice. The information in the video library will also be directly available to a student.

The graphics material will be invoked to provide support explanations (e.g. about meiotic events) to accompany tutorial advice. The graphics managed by the LIBRARIAN are of two types--fixed visuals from the video disk and interactive, computer-generated graphics. The fixed visuals will include, for example, both commercially-produced stills and moving visuals of actual cells undergoing meiosis as well as stylized equivalents that illustrate only the most salient features of meiosis. Such immediate access to high quality video materials is not typically part of genetics instruction.

The second type of visual materials under the management of the LIBRARIAN is computer-generated graphics. For example, an understanding of the mechanism of meiosis can help a student explain his or her solution to a problem (a desired learning outcome) and recognize trends in the data which may not correspond to a simple independent assortment pattern. Once students recognize such a situation, they can begin to think of how linkage (including variable map distances and/or interference) might help to explain the patterns observed in the data. We have chosen to work vith meiosis first since it is so central to understanding genetics roblem-solving and because students have difficulty understanding meiotic rocesses [21, 36]. One of the ways that we have done this is through the levelopment of a module called LINKAGE.

When LINKAGE is invoked by the LIBRARIAN or the student, it can ielp the student better understand meiosis by providing an opportunity to est various hypothesis that they may have to explain their data. By nvoking LINKAGE, the student can create customized chromosome/gene nodels. This is done by allowing the students to:

1. create chromosome/gene arrangements for two parental organisms;
2. vary the map distances separating any linked genes and turn interference *on* or *off*;
3. observe the chromosomes that they have created undergo meiosis;
4. select the number of offspring to result from crossing two parents;
5. observe the offspring phenotype distribution that results from the cross;
6. change any of the above variables and observe how the offspring phenotype data is effected.

Thus a student working with a three-trait problem might begin with a model in which each individual had three pairs of homologous chromosomes (e.g. where the chromosomes assort independently and therefore are not linked). Two individuals could be identified as parents and that offspring phenotype distributions for a specified number of offspring in that generation could be observed. It would then be possible to construct a single pair of chromosomes so that all three genes are on the same chromosome pair (e.g. linked) and do the exact same thing that was just done for the unlinked situation. The student constructs as many alternative chromosome/gene arrangements as desired, thus having relatively immediate opportunities to observe how multiple chromosome/gene models lead to different patterns in the phenotypic data. The importance of programs like this, which the LIBRARIAN manages, is not only that they serve a tutorial function, but they provide a student with opportunities to work with multiple models of phenomena--something that is common in science, but less so in science instruction.

<u>The Student MODELER Component of the TUTOR</u>. In order for the TUTOR to intervene in the student's problem-solving process with tutorial advice, it must have access to information about that student. The function of the student MODELER is to gather such information, make inferences from it about the state of the students's knowledge (both strategic and conceptual), and make that information available to the TUTOR.

At the very least, the MODELER must keep a history of student actions such as: the vials(s) from which organisms are selected for crosses, the making and checking of hypotheses, the making of inferences about the genotypes of individuals or phenotype classes, and if and when students do statistical analyses. Some of this information will be directly available from

a student's interactions with the basic GENERATOR program (the vials from which parents were taken) or by taking advantage of other GENERATOR functions (statistics or the Pedigree chart function).

Beyond this, the MODELER will need to recognize patterns in a set of individual actions and to make inferences about some student actions. For example, it is possible to recognize quickly that a student is taking all parental organisms from Vial#0. Although a problem could be solved by doing this, it is not an ideal approach because it does not acknowledge the importance of looking at data from within a lineage of several generations. It is therefore necessary to recognize when a student either misses a warranted inference or makes an unwarranted inference. This could be done directly by noticing when a student fails to enter genotype information on the pedigree chart or enters an unwarranted genotype. In order to recognize either student action, or lack of action, it is necessary to make comparisons with what action the SOLVER could make in response to the same data.

A student solving problems will execute a set of actions similar to the SOLVER'S agenda. These actions can be modeled as problem-solving rules. In addition, there should be conceptual knowledge (more than rules or empirical associations) which underlie the rules. This causal knowledge (e.g. of meiosis) is the basis for problem-solving with understanding and model-based reasoning. Both rule-based and model-based reasoning are ultimately important [37]. Rule-based reasoning is easier for the MODELER to process, however, so we plan to develop this capability of the MODELER first. The MODELER's ability to infer student conceptual knowledge will be added gradually, bolstered by our research on novice knowledge of genetics and how that knowledge relates to problem-solving actions.

Concluding Remarks

In this paper, we have described an on-going research and development project that will result in a unique genetics problem-solving environment. The environment both simulates a transmission genetics laboratory and provides computer-generated advice. It is intended to supplement undergraduate genetics education although it is flexible enough to be used in high-school biology or graduate courses.

The **MENDEL** system embodies certain values and commitments to science education that have guided us in our design choices and research questions. Our commitments can be categorized around the following themes:

1. problem-solving with understanding;
2. problem-based, experiential learning;
3. integration of rule-based and model-based reasoning, and;
4. collaborative, machine-mediated learning environments that

embody the foregoing themes.

Our commitment to the importance of problem-solving with understanding (as opposed to efficient problem-solving performance *per se*) is based on our own experience as science teachers, our research on problem-solving, and our critical analysis of the potential dangers of mindless learning in computer-based education.

The importance of problem-solving with understanding was driven home in one of our studies with high-school genetics students who were using the GENERATOR program. At one point, when a group of these students was having a particularly hard time with one of the computer-generated problems, the instructor inadvertently suggested what our research had shown to be a very powerful problem-solving rule. The students henceforth applied that rule to similar problems without thinking of the underlying genetics mechanisms. We had inadvertently created students who mindlessly followed rules. This is not to suggest that we are against rules or rule-following. Rather, we want rules to emerge in the minds (and behaviors) of our learners as a result of experience and understanding. A tutor must therefore do much more than reveal problem-solving rules. This brings up our second commitment.

Problem-based learning is emerging as an alternative approach within medical education [38] and experiential learning is already well established in organizational theory and business education. [39] We have learned from these traditions as well as from our work on strategic simulations that long-term inferencing is best learned through a series of experiments and associated problem-solving activities [28, 29].

In many ways, problem-based, experiential learning is nothing new because most scientists learn to do science in this way. However, most students who take introductory science courses do not become scientists and therefore do not have this experience. At most, they get a simplified, sanitized, rational-reconstruction of science from a text book while sitting in large lecture halls. This is not science but a rhetoric of conclusions.

What we are trying to do is to offer these students some experience at conducting genetics experiments, generating and testing hypotheses, and developing some understanding of genetics problem-solving. The **MENDEL** system is one way to make this feasible. We realize that some aspects of problem-based learning and experiential learning cannot be simulated in our environment. For example, we do not include the initial abstraction stages of identifying traits and variations of organisms. How important perceptual discernment and abstraction are for genetics understanding remains an open research question. Whether we could use, or would want to use, the videodisc to simulate these initial stages of doing science also remains to be seen. We have chosen to give the videodisc a different role in our project.

Our version of problem-based, experiential learning provides students with significant and realistic transmission genetics problems to solve. Our environment then provides students with computational tools, graphical representation of genetics concepts, and tutorial advice that encourage

conceptualization about the underlying genetics mechanisms. It does so by letting students pose questions, make conjectures (i.e., enter hypotheses), and learn from their experience (i.e., perform crosses, use computational tools). Conceptualization here refers to both genetics-specific content and the nature of scientific inquiry. This brings us to our next commitment.

As mentioned earlier, students are quite willing to stop at the rule-following level of problem-solving. However, students are also able to understand the reasons behind problem-solving strategies. We, as educators, therefore have an obligation to help our students reach their full potential. In science education, this means reaching a certain level of scientific understanding and scientific inquiry. We try to achieve this within the constraints of the **MENDEL** system by helping students use model-based reasoning as well as rule-based reasoning. Rule-based reasoning is aided by the TRACE-STOP and NEXT-STEP commands where students are presented with the heuristic problem-solving rules that the SOLVER uses. These commands present rules in the exact problem-solving situation to which they apply. Thus, the student can actively engage in applying the rule. Model-based reasoning is aided by the JUSTIFY command as well as by the LIBRARIAN's routines. For instance, the LINKAGE module of the LIBRARIAN will be used to explain rules for generating and testing linkage hypotheses in model-based terms.

A key aspect of model-based reasoning is that the solution to a problem is actually the hypothesis in the mind of the student throughout the problem-solving process. Students therefore have to develop problem-solving strategies that exercise their critical and judgmental faculties and not just their technical abilities. Students also have to be sensitive to the data that emerge in their experiments. Model-based reasoning therefore becomes the link between theory-directed and data-directed problem-solving. Model-based reasoning can also be seen as the key to understanding the empirical associations of problem-solving rules.

Problem-solving with understanding, problem-based, experiential learning, and model-based reasoning do not occur in isolation. They are not merely individual psychological processes in the mind of the learner but are inherently social processes. We therefore believe that this type of learning requires collaboration with others. We try to structure our problem-solving environment and our tutorial advice so that collaboration between students and tutors can take place. Furthermore, we have made our simulation of a genetics laboratory complex enough so that robust experimentation can take place (i.e., the GENERATOR is *not* a toy universe) and so that heuristic approaches to solving problems can take precedence over algorithmic approaches (e.g., where multiple conceptualizations and mixed data-driven and theory-driven approaches can take place). This is fertile ground for collaboration.

Our final commitment deals with how we believe computers should be used in science education. We believe that computers should be used for strategic simulations in order to *supplement* science education. Strategic simulations remain a rational reconstruction of scientific experiments, no matter how complex they become, and so can never replace actual experimentation. We also believe that computer tutors should play an advisory rather than in a supervisory role. Computer tutoring is a new type of tutoring rather than a substitute for human tutorial engagement. Human tutoring still remains central for science education. Our final commitment therefore translates into a vision of the computer as a science teacher's assistant.

Acknowledgements

We gratefully acknowledge the support of the following individuals and organizations: Dean John Palmer and the University of Wisconsin-Madison School of Education; the IBM Corporation for the support that they ave given to UW-Madison's Project TROCHOS; the FIPSE (Fund for the Improvement of Postsecondary Education) program of the US Department of Education (Grant# G008301471 and Grant# G0085410.39); and the National Science Foundation (Grant# MDR-8470277).

Special thanks go to Judith Van Kirk and Gary Price for many helpful comments on earlier drafts of this paper.

References

1. National Academy of Science & National Academy of Engineering. *Science and Mathematics in the Schools: A Report of a Convocation,* Washington, DC, National Academy Press, 1982.

2. E. P. Volpe, "The shame of science education". *American Zoologist*, 24:433-443, 1984.

3. D. Sleeman and J. S. Brown (eds), *Intelligent Tutoring Systems*, New York: Academic Press, 1982.

4. W. J. Clancey and R. Letsinger, "NEOMYCIN: Reconfiguring a rule-based expert system for application to teaching," in W. J. Clancy and E. H. Shortliffe (eds.), *Readings in Medicial Artificial Intelligence; The First Decade*, Reading, Mass: Addison-Wesley, 1984.

5. M. McCloskey, A. Caramazza, and B. Green, "Circular motion in the absence of external forces: Naive beliefs about the motion of objects." *Science*, 210 (4474):1139-1141, 1980.

6. A. A. DiSessa, "Unlearning Aristotelian physics: A study on knowledge-based learning." *Cognitive Science*, 6(1): 37-75, 1982.

7. M. Hackling, and D. Treagust, "Research data necessary for meaningful review of grade ten high school genetics curricula."

Journal of Research in Science Teaching, 21:197-209, 1984.

8. M. Smith, M. and R. Good, "Problem-solving and classical genetics: successful vs. unsuccessful performance." *Journal of Research in Science Teaching,* 21(9): 895-912, 1984.

9. J. H. Larkin, J. McDermott, D. P. Simon, and H. A. Simon, "Expert and novice performance in solving physics problems." *Science*, 208:1335-1342, 1980.

10. F. Reif, "Understanding and teaching problem-solving in physics. Research on physics education." *Proceedings of the First International Workshop on Physics Education. Paris*, Editions du Centre National de la Reserche Scientifique, 1983.

11. P. Hewson, "A conceptual change approach to learning science." *European Journal of Science Education*, 3:383-396, 1979.

12. K. Strike, G. Posner, P. Hewson, and W. Gertzog, "Accommodation of a scientific conception: Toward a theory of conceptual change." *Science Education,* 66: 211-227, 1982.

13. J. S. Brown, and R. Burton, "Diagnostic models for procedural bugs in basic mathematical skills." *Cognitive Science*, 2(2): 155-192, 1978a.

14. W. J. Clancey, "The epistemology of a rule-based expert system--A framework for explanation." *Artificial Intelligence*, 20:215-251, 1983.

15. B. Woolf, and D. D. McDonald, "Building a computer tutor: Design issues." *Computer*, Sept., 61-73, 1984.

16. J. R. Anderson, C. F. Boyle, R. Farrell, and B. J. Reiser, "Cognitive principles in the design of computer tutors." *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*. Boulder, CO., 1984.

17. L. Steels, and J. A. Campbell (Eds.), *Progress in Artificial Intelligence.*, New York: John Wiley & Sons, 1985.

18. J. Lipson, C. V. Bunderson, B. Baillo, J. B. Olsen, and K. M. Fisher, "Evaluation of an intelligent videodisc for developmental biology instruction." *International Journal of Machine-Mediated Learning*, 1, 1983.

19. R. Tennyson, "Artificial intelligence methods in computer-based instructional design." *Journal of Instructional Development,*

7(3):17- 22, 1984.

20. T. Munakata, "Knowledge-based systems for genetics." *International Journal of Man-Machine Studies*. 23: 551-561, 1985.

21. J. H. Stewart and M. Dale, "Solutions to genetics problems: Are they the same as correct answers?" *Australian Science Teachers*, 27(3):59-64, 1981.

22. J. H. Stewart, "Student problem-solving in high-school genetics." *Science Education*, 67(4): 523- 540, 1983.

23. N. Thompson, and J. H. Stewart, "Secondary school genetics instruction: Making problem-solving explicit and meaningful." *Journal of Biological Education*, 19(1): 53-62, 1985.

24. W. Albright, "Problem-solving strategies used by university students to solve realistic genetics problems." (Master's Thesis, in progress).

25. S. Slack, "Problem-solving strategies used by high-school students to solve realistic genetics problems." (Master's thesis, in progress).

26. A. Collins, "Strategic knowledge required for desired performance in solving transmission genetics problems." (Ph.D. dissertation, University of Wisconsin - Madison), 1986.

27. J. R. Jungck, and J. N. Calley, *GENETICS: Strategic Simulations in Mendelian Genetics*., Wentworth, N.H.: COMpress, 1986.

28. J. R. Jungck, "Strategic simulations: Experiencing research-like problem-solving in computer biology modules." In L.R. Meeth & D.S. Gregory (Eds.), *Dunedin, Florida Studies in Higher Education*. 133- 134, 1982.

29. J. R. Jungck, and J. N. Calley, "Strategic simulations and post-socratic pedagogy: Constructing computer software to develop long-term inference through experimental inquiry." *The American Biology Teacher*, 47(1):11-15, 1985.

30. M. J. Streibel, "Dialog design and instructional systems design for an intelligent videodisc system. " *Videodisc and Optical Disc*, 43(3): 216-219, 1984.

31. M. J. Streibel, "A critical analysis of the use of computers in education." Accepted for publication in the *Educational Com-*

munications and Technology Journal.

32. M. J. Coombs, (Ed.) *Developments in Expert Systems*. New York: Academic Press, 1984.

33. M. H. Richer, and W. J. Clancey, W.J, "GUIDON-WATCH: A graphic interface for viewing a knowledge-based system." *I.E.E.E. CG&A*, November, 1985.

34. D. R. Peachy, and G. C. McCalla, "Using planning techniques in intelligent tutoring systems." *International Journal of Man-Machine Studies*, 24: 77-98, 1986.

35. N. Peterson, J. Jungck, D. Sharpe and W. Finzer, "A design approach to science: Simulated Laboratories, learning via the construction of meaning." *Machine-Mediated Learning*, 2:1, 1986.

36. J. Stewart,and M. Dale, "High school students' models of chromosome/gene behavior." MENDEL Technical Report #2, 1986.

37. P. A. Koton, "Empirical and model-based reasoning in expert systems." *Proceedings of the 9th International Conference on Artificial Intelligence*, 1: 297-298. Aug. 18-23, Los Angeles, 1985.

38. H. S. Barrows, and R. M. Tamblyn, *Problem-based Learning: An Approach to Medical Education*. New York: Springer Publishing, 1980.

39. D. A. Kolb, *Experiential Learning*. Englewood CLiffs, NJ: Prentice- Hall. 1984.