Predicate Calculus and the Mathematical Thinking of Students

Ed Dubinsky
Georgia State University, Atlanta GA

Olga Yiparaki
Agnes Scott College, Decatur GA
University of Arizona, Tucson AZ

1 Preliminaries

This report is based on two related projects. The first, which took place about 10 years ago was an attempt to apply ideas taken from Piaget to analyze how students might come to understand predicate calculus and apply this analysis in designing and implementing instruction. It resulted in two papers (Dubinsky, Elterman, Gong, 1988, Dubinsky, in press) and was done in collaboration with Flor Elterman and Cathy Gong. The second project is more current and ongoing. It is a return to the earlier work in an attempt to apply a subsequently developed general framework for research and curriculum development in mathematics topics at the post secondary level. This framework is described in some detail in Asiala, Brown, DeVries, Dubinsky, Mathews & Thomas, 1996). The present work is being done in collaboration with Olga Yiparaki. We are at the early stages and our new results are, at best, highly preliminary.

According to the framework under which we are operating, the research has strong theoretical, empirical, and practical components. It is theoretical in that the epistemology of the concept(s) in question is studied and used to propose mental constructions that could lead to learning these concepts. It is empirical (in both quantitative and qualitative senses) in that data is gathered about whether the proposed mental constructions are being made and whether they appear to help learning. The research is practical because all data is gathered in the contexts of actual courses and students who are attempting to learn the material in those courses, and also because it is used to directly influence design of instruction.

In this report, we will begin with a somewhat pedagogical/philosophical (political?) statement about what we think is the role of predicate calculus in understanding mathematics. Here we will point out, with examples from the current study with Yiparaki, that this role does not appear to work for many students, and we will specify goals for instruction in predicate calculus. In the next section we will give a very brief outline of the general epistemological or theoretical framework under which we engage in research and curriculum development that is aimed at these goals. Then we will explain how it applies to predicate calculus. Next we will describe

the instructional treatment that follows from this epistemological analysis. Finally we will refer briefly to the results of using this approach for predicate calculus.

2 The role of predicate calculus in understanding mathematics

Aside from one's philosophical beliefs about the nature of mathematics, there is no question about the pervasiveness of predicate calculus in mathematics. One role of predicate calculus in mathematics is as a tool for making sense of and analyzing mathematical statements which are not otherwise understood. Before expanding on this, let us set up some examples to illustrate the point.

It is almost impossible to think of a topic which does not involve quantifiers, at least in principle, in descriptions, relationships and results. For example, in abstract algebra, the definition of identity is a two level exist/forall (EA) statement and the Inverse Axiom is a two level forall/exist (AE) statement.

A perhaps more mathematically difficult example is the distinction between the definition of a continuous function on a domain and a uniformly continuous function on that domain. It is perhaps of value to write these definitions formally and compare their logical structure. For a function f from a subset D of the real numbers to the real numbers, we have for continuity at each point in D,

$$(\forall x \in D) \quad (\forall \epsilon > 0) \quad (\exists \delta > 0) \quad (\forall y \in D) \quad [\, |x - y| < \delta \rightarrow |f(x) - f(y)| \leq \epsilon \,],$$

and for uniform continuity on D,

$$(\forall \epsilon > 0) \quad (\exists \delta > 0) \quad (\forall x \in D) \quad (\forall y \in D) \quad [\, |x - y| < \delta \rightarrow |f(x) - f(y)| \leq \epsilon \,].$$

Formally, the only difference is the movement of the first quantifier to the third position. We might say that the first statement is an $\mathrm{A_1A_2EA_3}$ statement and in the second it has been transformed to an $\mathrm{A_2EA_1A_3}$.

The simplicity of the formal distinction does not mean that very many students can understand the difference between the two statements, can realize that uniform continuity implies continuity, can see why a constant function is uniformly continuous or check that various elementary examples of continuous functions are or are not uniformly continuous. It is important to try to understand why not and what can be done about it.

After one has constructed an understanding of a concept in question, such reasoning can be done and the predicate calculus is not much more than a relatively minor convenience used by

some and not by others according to taste. This may be the situation for many students with regard to identities and inverses. But predicate calculus becomes important when dealing with concepts which are not well understood and for which analogies and mental pictures are hard to come by — such as uniform continuity.

We would argue that rather than focus on just making better analogies or clearer explanations, pedagogical strategy for concepts like uniform continuity should aim at helping students understand quantifiers and learn to use them to construct mental images of complex situations.

Preliminary report of a study

In the first step of this study, we administered a questionnaire to 42 students, 21 at a large midwestern research university and 21 at a small southern liberal arts college. This questionnaire, which is included in Appendix A, presents the student with 11 statements. The student is asked to decide if the statement is true or false and to explain why. In the second step, we conducted in-depth interviews of selected students in which we tried to probe into what they had in mind with their answers. The interviews are still being analyzed and what we have to say in this report relates only to data from the first step.

In Table 1 is listed, for each statement, the interpretation (AE or EA) that the authors make of the statement, the number of students whose interpretation appeared to agree with ours, the number whose interpretation seemed to disagree with ours, the number for whom we could not tell what was their interpretation, the number who, given their interpretation, answered the question correctly, and the number who answered it incorrectly in terms of their interpretation. (Note that the second column of the Table 1 lists our own interpretation for the statements in our questionnaire. However, in a few cases (especially Statements 4 and 8) our intention was to choose statements that could be seen as having ambiguous quantification order. We expect to have more comments on this topic after we finish the analysis of our interviews.)

There are several comments we can make about these numbers. For the first nine statements, which referred to "every day" situations, there were a large number of cases in which the student interpreted an EA statement as an AE statement and only one case in which the reverse was true. In general, given their interpretations, the students did reasonably well with these first nine statements, giving a total of 290 correct answers out of 378 responses. On the other hand, they did much worse on the last two statements which involved mathematics. Here there were only 23 correct answers out of 84 responses.

One possible explanation for the incorrect interpretations is that there is a general tendency

Statement	AE/EA	# correct interpretations	# incorrect interpretations	# undetermined interpretations	# correct	# incorrect
1	AE	36	0	6	36	0
2	AE	39	0	3	38	1
3	EA	27	6	9	32	1
4	EA	4	37	1	41	0
5	AE	33	0	9	31	2
6	EA	9	17	16	23	3
7	AE	34	1	7	34	1
8	EA	13	17	12	30	0
9	EA	10	18	14	25	3
10	AE	24	0	18	19	5
11	EA	5	9	28	4	10

Table 1: Results of Questionnaire Administered to 42 Students

to prefer AE statements, possibly because EA statements are harder, or at least, less common in ordinary discourse. Another possible explanation is that there is a tendency to prefer true statements and an AE statement is more likely to be true than the corresponding EA version. The relatively high number of correct interpretations of Statement 3 is consistent with the above explanations as several students appealed to a deity as justification for the statement being true.

Turning to the last two questions we note that several students explained their interpretation of Statement 11 by saying that it was exactly the same as Statement 10 (in some cases there was even the explicit addendum to the affect that only the word order was changed — which did not seem to change the meaning for the student).

Given that each of the logical structures appearing in the last two statements is the same as one of the logical structures appearing in the first nine statements, we cannot explain the sharp drop in the number of correct answers by saying that the students could not handle the logical structure. Moreover, given that many of these students had taken a number of mathematics courses, it does not seem reasonable that the comparison of positive numbers was the only obstacle for the students.

Here is one tentative explanation of these results. It could be that the good performance of the students on the first nine statements is *not* due to their ability to use, consciously or not, the formal structure of a sentence and predicate calculus to make sense of the situation. If it were, then they would do the same with the last two statements. Rather, it might be that what the student does is try to build a mental picture, based on personal experience or imagination, that represents what the statement is saying. All of the first nine statements are about situations that are familiar to the students, or could be. The student uses the mental picture of a situation, *and not the logic*, to decide on the truth or falsity of the statement. This

is consistent with the preference for true statements and the fact that the logical structure might even be ignored in order to have a statement that is true. (We note that our preliminary work with the interview transcripts suggest that this data might contain supporting evidence for both of these tendencies.)

In the case of the last two statements, most students do not find that these statements describe situations that are familiar to or imaginable by them, and so they are unable to decide on the truth or falsity. Apparently, no more than 1/4 of the students were able to use the logical structure of the situation to interpret and understand these two statements.

If this explanation is valid then the pedagogical problem is more difficult than may have been thought. As one moves into more advanced mathematics, there are more and more statements which do not describe familiar, or easily imagined situations and the logical structure may be the only tool available to make sense out of a situation. That is, we believe, the way mathematicians make sense out of complex statements. They use the tool of predicate calculus (implicitly or explicitly) to convert these strange statements into familiar situations by building new images that may be fundamentally different from what is previously in their experience. This would suggest that relating a complex statement to an "everyday situation" is actually of little value in helping students construct meaning for a statement describing an unfamiliar mathematical situation. The reason might be that the situation can only be imagined and related to the statement *after* an understanding has been built; for example by analyzing the formal structure of the statement with the aid of predicate calculus.

On the other hand, our explanation of this data implies that not many students are able to use this tool. Thus, the pedagogical task is both to get students to understand the logical structure of quantified statements and, second, to get them to use this structure to make sense out of unfamiliar situations.

We turn now to a program for doing just that. It begins with an epistemological analysis of predicate calculus as a conceptual tool.

3 Epistemological Perspective

We begin with a brief summary of our general framework (for a more complete description, see Asiala et al, 1996) and then we will describe how the theoretical component of the framework is applied to quantification.

Summary of framework for research and curriculum development

As we have indicated, our framework for research and curriculum development has three components: theoretical analysis, instructional treatments, and gathering data.

The first step in our approach is to make an initial theoretical analysis using our theoretical perspective on learning theory, the epistemology of the concept being studied based upon past research, literature, and the mathematical knowledge of the researchers. The purpose of the theoretical analysis is to propose a *genetic decomposition* or model of cognition: that is, a description of specific mental constructions that a learner might make in order to develop her or his understanding of the concept. These mental constructions are called actions, processes, objects, and schemas, so that the theoretical framework we use is sometimes referred to as the APOS Theory.

According to APOS theory, an *action* is a transformation of mathematical objects that is performed by an individual according to some explicit algorithm and hence is seen by the subject as externally driven. It is a reaction to stimuli which the subject perceives as external. The action tends to control the individual.

Consider, for example, the proposition,

> Every member of my family is unemployed.

If an individual can only think of this statement in terms of the reality of certain specific members of her or his family and whether or not they have jobs, then he or she is understanding this statement as an action. In this case, there would be no indication that the individual can think of all members of a perhaps extended family and imagine whether the statement is true or not, because the individual can only focus on a specific person who may or may not be employed.

When the individual reflects on an action and constructs an internal operation that performs the same transformation then we say that the action has been *interiorized* to a *process*. In a sense, the individual establishes control over the action and is able to do more with it, for example, by imagining the transformation without needing to perform it explicitly. In the above example, it is possible for an individual to think about all members of her or his family, or even someone else's family, to imagine checking the employment status of each individual, and to decide on the truth or falsity of the statement by summarizing all of the data obtained.

When an individual reflects on operations applied to a particular process, becomes aware of the process as a totality, realizes that transformations (whether they be actions or processes) can act on it, and is able to actually construct such transformations, then he or she is thinking of

this process as an object of thought. In order to perform actions on a process, the subject must first *encapsulate* it to become a total entity, or an *object*. In many mathematical operations, it is necessary to de-encapsulate an object and work with the process from which it came.

In our example, an individual might think about applying negation to this statement and so imagine it being true, say for one family, but false for another. In this way, the statement becomes an object that has a different status depending on the family in question. For each family, however, there is a definite proposition which may be true or false.

Finally, a *schema* is a coherent collection of processes, objects and previously constructed schemas, that is invoked to deal with a mathematical problem situation. As with encapsulated processes, an object is created when a schema is *thematized* to become another kind of object which can also be *de-thematized* to obtain the original contents of the schema.

Theoretical analysis of quantification

We will refer to Figure **??** in applying this analysis to describe the genetic decomposition of quantification that was developed in Dubinsky, Elterman & Gong (1988).

The construction begins with simple declarations that may be true or false. These are the basic mental objects that form the building blocks for all that follows. They are made more complex in two ways. First, several such objects are collected in a set and, making use of the learner's schema for function (the existence of which is a prerequisite for understanding quantification), variables are introduced to obtain proposition-valued functions, interpreted as processes. This means that the individual is able to think about iterating through the function's domain, checking the truth or falsity of the proposition for each value of the variable. Second, two or more declarations are coordinated by linking them with the standard logical connectors of either conjunction or disjunction.

Once these two processes have been constructed, the learner is ready to move on to the construction of single-level quantifications, which consist of a single quantification, universal or existential, applied to a proposition-valued function of one variable. In the diagram, this is indicated by the joining of the two large branches at a node in the middle of the page. The transition is achieved by coordinating the two processes previously constructed. That is, the propositions that are obtained for the various values of the variable are all connected by conjunctions (for universal quantification) or disjunctions (for existential quantification) resulting in a single proposition. Thus the learner interiorizes a process of iterating a variable through its domain to obtain a set of propositions to which a quantifier is applied.

Figure 1: Genetic Decomposition of Quantification

Next come two-level quantifications, displayed below the middle of the diagram, in which two (usually different type) quantifiers are applied in succession to a proposition-valued function of two variables. The process which we just described for constructing single-level quantifications must be encapsulated so that the result becomes a proposition which is a mental object. Note that the effect of the process is to eliminate a variable. If the original proposition-valued function had two variables, then the resulting object actually depends on the value of the other variable and the schema for single-level quantifications can again be applied to this proposition-valued function. Thus, when analyzing a statement which requires a two-level quantification over two variables, the subject begins by parsing it into two quantifications. There is an inner quantification over one of the variables in a proposition-valued function of two variables. There is also an outer quantification over the other variable. What we have described is a coordination of these two quantifications to obtain a third which will be two-level quantification. In order to proceed to higher-level quantifications this new process must again be encapsulated to obtain a single proposition.

Given a statement which is a three-level quantification, the subject can group the two inner quantifications and apply the two-level schema to obtain again a proposition which depends on the outermost variable. This proposition-valued function is then quantified as before to obtain a single proposition. The entire procedure can now be repeated indefinitely to obtain quantifications of any level.

4 Instructional Treatments

Genetic decompositions of mathematical concepts are the basis for the design of the instructional treatments. The pedagogical approach used to implement the instructional treatments for helping students make the proposed mental constructions is called the ACE teaching cycle (**A**ctivities, **C**lass tasks and **E**xercises). The main strategies of this method include having students construct mathematical ideas on the computer using a mathematical programming language, investigate mathematical concepts using a symbolic computer system, and work in cooperative learning groups to engage in problem solving activities and in discussions of the results of the computer and problem solving activities.

Thus the next phase in our paradigm is to design instructional activities that will induce the students to make the desired constructions. This is a much more delicate operation than is perhaps generally assumed. The point is that if students are mentally active, if they are listening to lectures, doing homework, studying for tests, then they will be making mental constructions.

It is by no means certain that they will make useful constructions — that is, that they will construct schemas that will help them do mathematics. Perhaps more likely, they will construct ineffective procedures which some have called "bugs". Examples of this include the observations of students' (and teachers') errors in solving the Student/Professor problem (Clement, Lochhead, & Soloway, 1980), the "repairs" of Brown and vanLehn (Brown & vanLehn, 1980; vanLehn, 1980) misconceptions in Physics (DiSessa, 1985), the almost universal difficulty with Calculus (Douglas, 1986), and the trouble students have with implications. The literature reports not only on the initial errors that students make, but on the persistence of misunderstandings and on the lack of success that education seems to have in eliminating them. One possible explanation could be that the student's total experience leads her or him to construct an incorrect schema which is then reinforced by doing a large number of examples and misinterpreting explanations and illustrations.

We have found that computer experiences can be an effective way of not only helping students to construct reasonable schemas, but also to get them to reconstruct erroneous or incomplete conceptions. The basic principle is that anytime you construct something on a computer then, whether you are aware of it or not, you construct something in your head. By studying the connections carefully, we have found it possible to induce a considerable amount of learning. Specifically, in our theory, the main construction activities are interiorization, encapsulation, coordination, generalization, and reversal. These have counterparts in computer activities which we will describe in the next section. These activities can be used to get students to perform corresponding mental constructions.

Summary of mental constructions

Before going on to the instructional treatments and the description of the programming language that was used, let us summarize the most important mental constructions that, according to our analysis, the learner must make in order to construct the concept of quantification. These are the activities that the teacher must try to facilitate. The design of computer work in our instructional treatments was driven by the goal of finding tasks that would tend to lead students to make these particular mental constructions.

1. Coordinate two or more declarations by linking with logical connectors.
2. Generalize the schema of function to include proposition-valued functions.
3. Interiorize the action of iterating over the domain of a variable in a proposition-valued function, checking the truth or falsity of the proposition for each value of the variable.
4. Coordinate linking declarations with iterating over a proposition-valued function and apply a quantifier.

5. Encapsulate the process for a single-level quantification to obtain a single proposition.
6. Coordinate two instantiations of the single-level quantification to obtain a single proposition.
7. Encapsulate the process for a two-level quantification to obtain a single proposition.
8. Coordinate three or more instantiations of the single-level quantification schemas to obtain higher level quantifications.

In the next section we will try to explain how computer activities with ISETL can relate to each of these mental constructions.

Using ISETL

ISETL is an interactive programming language which runs on microcomputers and mainframes. It is based on the programming language SETL developed by Jacob Schwartz and his group at the Courant Institute. Full descriptions of these languages are found in Schwartz, Dewar, Dubinsky & Schonberg (1986) for SETL and Dautermann (1992) for ISETL. Baxter, Dubinsky, & Levin (1988) and Fenton & Dubinsky (1996) are texts for discrete mathematics courses (at two different levels) using ISETL. Both have chapters which contain a complete treatment of propositional calculus and quantification based on the research reported in this paper. In Dubinsky (1995) there is a description of how this approach is applied to a number of different undergraduate mathematics courses. We will give here only a very brief description of ISETL, concentrating on how it relates to the mental constructions listed at the end of the previous section.

ISETL is an interpreted language which means that, once it is invoked, expressions are typed on the terminal and, when a carriage return is entered, the system performs the activity indicated by the expression. This can be an evaluation, in which case a result is printed on the screen, an internal action such as an assignment to a variable, or some communication with the external system, such as writing to a file.

The syntax for simple constructions such as naming variables, assignments, if... then... clauses, loops, etc., is standard and very similar to Pascal. There are, however, almost none of the usual programming requirements such as type declarations, sizing, etc. As a result, the basic syntax of the language is very easy to learn and our experience with students suggests that relatively little time need be spent on teaching the language directly. Almost all of the students' ability to program in ISETL comes from working with mathematical constructions.

The syntax for operations and for the construction of complex objects is almost identical to standard mathematical notation. As a result, the student is often under the impression that he or she is studying the syntax of ISETL when, in fact, important mathematical ideas are being learned. This tends to make some things relatively painless.

ISETL supports standard types of values which can be entered directly, assigned to a variable, or obtained as the result of a calculation. These include integers, reals (actually floating-point approximations to real numbers), character strings, and Boolean. The last is the set of two values, $\{true, false\}$ and this permits the representation of simple declarations. The standard logical operations are supported and so it is possible to enter an expression such as

```
((P and Q) impl (R and P)) or (not Q);
```

The semicolon at the end is the command to evaluate the expression and so, if Boolean values for `P`, `Q` and `R` were previously assigned, then either true or false will appear on the screen.

The main use of this feature is to ask students to translate logical expressions given in English to ISETL, evaluate the expressions, write programs to construct truth tables, and so on. These activities relate to item 1 in the list of mental constructions at the end of the section on genetic decompositions above.

Procedures in ISETL are called `func`s. Students are asked to construct `func`s that represent proposition-valued functions. For example, if the following code is entered,

```
P := func(n);
        return n mod 223 = 0;
    end;
```

then a proposition-valued function has been constructed to represent the statement,

A number is divisible by 223.

This relates to item 2 in the list at the end of the section on genetic decompositions. The idea, in item 3, of iterating over the domain and checking the value is suggested by asking the students to evaluate `P(892)`, `P(100)` and many other values. It is not difficult, when the students have actually done this to get them to think about what the computer is doing when it makes those evaluations. We call this "thinking ISETL".

The coordination of items 1 and 3, which is item 4 in the list of mental constructions, involves the idea of a set, which is a fundamental mathematical concept, and another type of value in ISETL. Sets in ISETL must be finite. Following are some examples of simple ways to construct sets in ISETL,

```
{3, 7, 9};
{-2, 8.65, "a", {6, true, 87}};
{7 .  .  26};
{900, 902 .  .  1100};
```

Notice the mixing of value types and the nesting in the second set. The third set is the set of all integers from 7 to 26 and the fourth is the set of all even integers from 900 to 1100. (There

is also a more powerful method of constructing sets in ISETL that is very close to set formation in mathematics. The syntax is

```
{expr :  x in S | P(x)}
```

but this will not be of concern in this paper).

At this point the student works with lists of propositions. For example, there could be 5 numbers $b_1, b_2, \ldots, b_5$ and a situation is created where the student would form the conjunction of each of the propositions,

$$b_i > 2, \quad i = 1, 2, \ldots, 5.$$

In ISETL this would look like

```
(b1 > 2) and (b2 > 2) and (b3 > 2) and (b4 > 2) and (b5 > 2);
```

There is an ISETL construction that represents this with a combination of a proposition-valued function and a universal quantification. One first constructs the function,

```
F := func(b);
          return b > 2;
     end;
```

and the set,

```
B := {b1, b2, b3, b4, b5};
```

Then one can enter

```
forall b in B | F(b);
```

and, the result, true or false, depending on the values of the b's, will be printed on the screen.

As another example, the student might be asked to express in formal language a statement such as

Out of all of the even integers between 900 and 1100, one is divisible by 223.

Using the `func P` defined above and assuming that S has been assigned the value {`900, 902 ... 1100`}, then taking ISETL to be the formal language, the student can write

```
exists n in S | P(n);
```

A major assumption of our approach is that coordinating the two processes on the computer to construct such expressions that can be manipulated in various ways, and reflecting on what has been done, will lead the student to construct a process corresponding to a single-level quantification, and interiorize it in her or his mind.

The encapsulations required for items 5 and 7 seem to be very difficult for students. Sfard (1991) considers the overall difficulty of encapsulation and suggests that this mental construction may be possible for only a fraction of students. It is here that certain special features of ISETL are particularly helpful. For example, if the above constructions have been made, one can perform the assignment

```
Q := forall b in B | F(b);
```

The variable `Q` can then be used in any expression combining Boolean values. Our idea is that performing such an action will make it more likely that a student will reflect on this single-level quantification, see it as a totality and even convert it mentally into an object, which is the requirement of item 5 in our list. A similar procedure can be followed for two-level (and higher) quantifications, once they have been constructed as processes.

To satisfy item 6 by coordinating two single-level quantifications it is necessary to exploit the fact that a process such as a `func` in ISETL is a piece of data and can be made an element of a set, assigned to a variable, or returned as the value of another `func`. Suppose, for example, the student is asked to analyze the following statement and express it in formal language.

There is a number in the set S which is dominated by every number in the set B.

The first step would be to replace the function `F` above by the following function of two variables,

```
G := func (b,n);
        return b > n;
     end;
```

Then the fairly complicated coordination can be implemented in two steps. First one "quantifies out" the variable b in G to obtain the following proposition-valued function of one variable,

```
G1 := func(n);
         return forall b in B | G(b,n);
      end;
```

and then the coordination is achieved by entering,

```
exists n in S | G1(n);
```

Actually one can go even further with this delicate point. It is possible in ISETL to automate all or part of the process. For example, one can construct a `func` that will accept as input a set and a proposition-valued function of two variables. It will then return a proposition-valued function of one variable obtained by "quantifying out" the other variable. The whole idea of "returning a function", that is, to work a problem whose solution is not a number but a function, is very new and profound for many students. Thus this activity helps with three important mathematical concepts: quantification, finding a function as the solution to a problem, and replacing a function of two variables by a function of one variable whose values are each functions of one variable.

The rest of the ACE Teaching Cycle

It should be noted that the implementation of the ACE Teaching Cycle involves students working in cooperative groups. These are permanent groups which the instructor assigns at the beginning of the semester and which remain intact for the entire course. Students do all of their work in these groups. This includes the lab activities, classroom discussions, homework, and some of the examinations. For more details on how cooperative learning is used in this approach, see Hagelgans, Reynolds, Schwingendorf, Vidakovic, Dubinsky, Shahin & Wimbish (1995).

In the ACE cycle, there are three components. First, the students work on computer tasks as described above in a computer lab. The purpose of these activities is for the students to begin to make the mental constructions called for by the theory.

Then, in class sessions with no computer available, discussions are conducted concerning the mathematical concepts related to the mental constructions which the students are expected to have made, or are in process of making, as a result of the computer work. From time to time, the opportunity is taken for the instructor to sum up what the students are learning, or to describe mathematical ideas that they are constructing in standard mathematical terms. The main goal of the classroom sessions is for students to reflect on what they have been doing in the lab and to relate their constructions of actions, processes, objects and schemas to mathematical concepts.

Finally, the students are assigned problems to be done, for the most part, with paper and pencil and without the aid of technology. That is, these are standard mathematical problems for which technology is not particularly relevant. The purpose of these problems is to reinforce the understandings the students have constructed, to probe the limits of their knowledge about the course material, and to get them to begin thinking about new ideas that are about to be considered in the course.

5 Results

Implementing the instruction provides an opportunity for gathering data. There are two ways in which these data are related to the genetic decomposition. First, the genetic decomposition directs the analysis of data by asking the question: did the proposed mental constructions appear to be made by the students? Second, the results of this analysis may lead to revisions in the genetic decomposition, which may lead to changes in the instructional treatment. The cycle is then repeated as many times as is necessary for the researcher to come to a deeper understanding of how students can construct their understanding of the concept. Initially, the genetic decomposition is based primarily on the researchers' own understandings of the concept, and on their experiences as learners and teachers. As the cycle is repeated, however, the genetic decomposition comes more and more to reflect the analysis of data. In each repetition of the research cycle, additional data are gathered to report on the performance of students on mathematical tasks related to the concept in question. The analysis of this additional data tries to determine directly what mathematics may have been learned, rather than what mental constructions might, or might not have been made.

In other words, we collect different kinds of data. Some of it is analyzed to see what mental constructions the students might have made. If their constructions are what is required by the genetic decomposition, then the theory predicts that mathematics will have been learned. A second kind of data is analyzed to see if this prediction is supported. In this way, both our theoretical analysis and our instructional treatments are evaluated.

Thus the outcome of work under this framework is, by its nature, two-fold. One result of the research is the deepening of our understanding of the epistemology of the concept. The second result is the creation of pedagogical strategies which are better aligned with the way we believe that students come to understand the concept; these improved strategies should thus lead to increased learning by the students.

Data and results for predicate calculus

In Dubinsky (in press) there is a report of a study in which the method described in this paper was used in two different discrete mathematics classes in two different universities: one a major research institution on the west coast and the other a small engineering school in upstate New York.

The method by which we collected data consisted of sets of questions on which students were asked to work individually and provide written responses. Some of these were given in

class without warning and some were to be taken away and returned two days later. Some of the problems were fairly difficult and the results suggest that the students did develop some understanding of quantification and the ability to work with it. This view is supported by the overall performance of the students in the courses.

Data provided by written responses to questions are not sufficient to draw definite conclusions. Follow-up studies need to conduct in-depth interviews to probe into what is the nature of the students' understanding of these concepts as the instructional treatment is being conducted and after it is finished. It would also be useful to make some sort of comparison with what happens given standard instructional treatments of quantification.

Because of the paucity of data, we can only say that we are in the preliminary stages of our work on helping students learn quantification. The strongest statement which can be made from our results is that our approach is promising. It is hoped that future work will provide more conclusive data about the mental constructions students are making and the mathematics they are learning as a result of the pedagogical strategy described in this report.

REFERENCES

Asiala, M., Brown A., DeVries D., Dubinsky E., Mathews D., & Thomas K. (1996). A framework for research and curriculum development in undergraduate mathematics education. *Research in Collegiate Mathematics Education II*, 1-35.

Baxter, N., Dubinsky, E., & Levin, G. (1988). *Learning Discrete Mathematics with ISETL.* New York: Springer.

Brown, J. S. & vanLehn, K. (1980). Repair Theory: A Generative Theory of Bugs in Procedural Skills, *Cognitive Science 4,* 379-426.

Clement, J., Lochhead, J., & Soloway E. (1980). Positive effects of computer programming on students' understanding of variables and equations, *Communications of the Association for Computing Machinery* 467-474.

Dautermann, J. (1992). *ISETL: A language for learning mathematics*, St. Paul: West Educational Publishing.

DiSessa, A., Knowledge in Pieces, Address to the Fiftieth Annual Symposium of the Jean Piaget Society, Philadelphia, June, 1985.

Douglas, R. G. (1986). Toward a Lean and Lively Calculus, Report of the Conference/Workshop to develop Curriculum and Teaching Methods, *Mathematical Association of America, Notes and*

Reports 6.

Dubinsky, E. (1995). ISETL: A Programming Language for Learning Mathematics, *Comm. in Pure and Applied Mathematics*, 48, pp. 1-25.

Dubinsky, E. (in press.) On Learning Quantification, *Journal of Computers in Mathematics and Science Teaching.*

Dubinsky, E., Elterman, F. & Gong, C. (1988). The student's construction of quantification, *For the Learning of Mathematics, 8*(2), 44-51.

Fenton, W. & Dubinsky, E. (1996) *Introduction to Discrete Mathematics with ISETL*, New York: Springer.

Hagelgans, N., Reynolds, B., Schwingendorf, K., Vidakovic, D., Dubinsky, E., Shahin, M., & Wimbish, G. (1995). *A Practical Guide to Cooperative Learning in Collegiate Mathematics* (MAA Notes Number 37). Washington, DC: The Mathematical Association of America.

Schwartz, J. T., Dewar, R. B. K., Dubinsky, E. & Schonberg, E., (1986). *Programming with Sets*, New York: Springer.

vanLehn, K., (1980). Bugs are not enough: empirical studies of bugs, impasses, and repairs in procedural skills, *The Journal of Mathematical Behavior, 3*(2), 3-71.

A Questionnaire

This questionnaire is to be filled out at one continuous sitting. There is no time limit. Please do not consult with anyone or any book (or other source) while you are still working on your answers.

Here are 11 statements. Decide for each one if it is True or if it is False, and indicate your answer by circling one of them. Then, underneath the statement, provide a brief explanation for your answer.

1. Everyone hates somebody. True False
   Why?
2. Every pot has a cover. True False
   Why?
3. Someone is kind and considerate to everyone. True False
   Why?
4. There is a mother for all children. True False
   Why?
5. All good things must come to an end. True False
   Why?
6. There is a magic key that unlocks everyone's heart. True False
   Why?
7. All medieval Greek poems described a war legend. True False
   Why?
8. There is a perfect gift for every child. True False
   Why?
9. There is a fertilizer for all plants. True False
   Why?
10. For every positive number a there is a positive number b such that $b < a$. True False
    Why?
11. There is a positive number b such that for every positive number a $b < a$. True False
    Why?

How much time did you spend on this questionnaire? _____