HUMAN PROCEDURAL SKILL ACQUISITION: THEORY, MODEL AND PSYCHOLOGICAL VALIDATION*

Kurt VanLehn

Xerox Parc
3333 Coyote Hill Rd.
Palo Alto, CA 94304

Abstract

It is widely held that ordinary natural language conversations are governed by tacit conventions. called felicity conditions or conversational postulates (Austin. 1962; Grice. 1975; Gordon & Lakoff, 1975). Learning a procedural skill is also a communication act. The teacher communicates a procedure to the student over the course of several lessons. The central idea of the theory to be presented is that there are specific felicity conditions that govern learning. In particular, five newly discovered felicity conditions govern the kind of skill acquisition studied here. The theory has been embedded in a learning model. a large AI-based computer program. The model's performance has been compared to data from several thousand students learning ordinary mathematical procedures: subtracting multidigit numbers, adding fractions, and solving simple algebraic equations. A key criterion for the theory is that the set of procedures that the model learns should exactly match the set of procedures that students actually acquire, including their "buggy" procedures. However, much more is needed for psychological validation of this theory, or any complex AI-based theory, than merely testing its predictions. The method used with this theory is presented.

Introduction

This paper summarizes research reported in a much longer document (VanLehn, 1983). It is intended only to convince the reader that the research problem is interesting and that the approach that was taken to solving it is worth taking.

The goal of this research is a psychologically valid theory of how people learn certain procedural skills. There are other AI-based theories of skill acquisition (Anderson, 1982; Anzai & Simon, 1979; Newell & Rosenbloom, 1981). However, their objectives differ from the ones pursued here. They concentrate on *knowledge compilation*: the transformation of slow, stumbling performance into performance that is "faster and more judicious in choice" (Anderson, 1982, pg. 404). They study skills that are taught in a simple way: first the task is explained, then it is practiced until proficiency is attained. The research presented here studies skills that are taught in a more complex way: the instruction is a lesson sequence, where each lesson consists of explanation and practice. This shifts the central focus away from practice effects (knowledge compilation) and towards a kind of student cognition that could be called *knowledge integration:* the construction of a procedural skill from lessons on its subskills.

This study puts more emphasis on the teacher's role than the knowledge compilation research does. It is not the case that multi-lesson skill acquisition occurs with just any lesson sequence. Rather, the lesson sequences are designed by the teacher to facilitate knowledge integration. Knowledge integration. in turn. is "designed" to work only with certain kinds of lesson sequences. So. what is really being studied is a teacher-student system that has both cognitive and cultural aspects. It might be more appropriate to label the central focus of this research *knowledge communication:* the transmission of a procedural skill via lessons on its subskills.

The skills chosen for the present investigation are ordinary written mathematical calculations. The main advantage of mathematical procedures, from a psychological point of view, is that they are virtually meaningless for the learner. They seem as isolated from common sense intuitions as the nonsense syllables of early learning research. In the case of the subtraction procedure, for example, most elementary school students have only a dim conception of its underlying semantics, which is rooted in the base-ten representation of numbers. When compared to the procedures they use to operate vending machines or play games, the subtraction procedure is as dry, formal and disconnected from everyday interests as a nonsense syllable is different from a real word. This isolation is the bane of teachers, but a boon to psychologists. It allows psychologists to study a skill that is much more complex than recalling nonsense syllables, and yet it avoids bringing in a whole world's worth of associations.

It is worth a moment to review how mathematical procedures are taught. In the case of subtraction, there are about ten lessons in a typical lesson sequence. The lessons introduce the procedure incrementally, one step per lesson, so to speak. For instance, the first lesson might show how to do subtraction of two-column problems. The second demonstrates three-column problem solving. The third introduces borrowing, and so on. The ten lessons are spread over about three years, starting in the second grade. They are interleaved with review lessons and lessons on many other topics. In the classroom, a typical lesson lasts an hour. The teacher solves some problems on the board with the class, then the students solve problems on their own. If they need help, they ask the teacher, or they refer to worked examples in the textbook. A textbook example consists of a sequence of captioned "shapshots" of a problem being solved, e.g.,

Take a ten to make 10 ones.	Subtract the ones.	Subtract the tens.
$\begin{array}{r} 2\ \ \\ \cancel{3}{}^{1}5 \\ \underline{-\ 1\ 9} \\ \end{array}$	$\begin{array}{r} 2\ \ \\ \cancel{3}{}^{1}5 \\ \underline{-\ 1\ 9} \\ 6 \end{array}$	$\begin{array}{r} 2\ \ \\ \cancel{3}{}^{1}5 \\ \underline{-\ 1\ 9} \\ 1\ 6 \end{array}$

Textbooks have very little text explaining the procedure (young children do not read well). Textbooks contain mostly examples and exercises.

* This research was supported by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research, under contract number N00014-82C-0067, contract authority identification number NR667-477.

Math bugs reveal the learning process

It will be assumed that the teacher and the student somehow build a knowledge structure of some kind in the student's mind, and that this knowledge structure somehow directs the student's problem solving efforts. A critical experimental task is to determine what that knowledge structure is. This is difficult if all the experimenter does is watch the student solve problems. There are many ways for the experimenter to get a better view, e.g., analyzing verbal protocols, measuring the latencies between writing actions, tracking eye movements, and so on. The technique used in this study is somewhat unusual: Students are given problems to solve that are beyond their current training. For instance, a subtraction student who has not yet been taught how to borrow is given problems which require borrowing, such as 43 – 18. When the student applies his current knowledge structure to solve such problems, it "breaks" in certain ways. The experimenter can infer much about the student's knowledge structure by analyzing his struggle to adapt it to solve the problem at hand. Metaphorically speaking, such testing is like snapping a bar of metal and examining the fracture under a microscope in order to find out the metal's crystalline structure.

Tools have been built for doing such detailed analyses. John Seely Brown and Richard Burton developed computer systems (Buggy and Debuggy) that automatically analyze a student's errors into one or more *bugs* (Brown & Burton, 1978; Burton, 1981). Bugs serve as a precise, succinct representation of errorful problem solving behavior. Bugs are the kind of data on which the theory rests.

Many different kinds of bugs have been observed (77 for subtraction alone; Vanlehn, 1982). Until recently, most bugs defied systematic explanation. As an illustration, consider a common bug among subtraction students: the student always borrows from the *leftmost* column in the problem no matter which column originates the borrowing. Problem *a* below shows the correct placement of borrow's decrement. Problem *b* shows the bug's placement.

a. $\begin{array}{r} 5 \\ 3\;6^{1}5 \\ -\,1\;0\;9 \\ \hline 2\;5\;6 \end{array}$ b. $\begin{array}{r} 2 \\ 3\;6^{1}5 \\ -\,1\;0\;9 \\ \hline 1\;6\;6 \end{array}$ c. $\begin{array}{r} 5 \\ 6^{1}5 \\ -\,1\;9 \\ \hline 4\;6 \end{array}$

(The small numbers represent the student's scratch marks.) This bug has been observed for years (c.f. Buswell, 1926, pg. 173, bad habit number s27), but no one has offered an explanation for why students have it.

The theory offers the following explanation, which is based on the hypothesis that students use induction (generalization from examples) to learn where to place the borrow's decrement. Every subtraction curriculum that I know of introduces borrowing using only two-column problems, such as problem *c* above. Multi-column problems, such as *a*, are not used. Consequently, the student has insufficient information for unambiguously inducing where to place borrow's decrement. The correct placement is in the left-adjacent column, as in *a*. However, two-column examples are also consistent with decrementing the leftmost column, as in *b*. If the student chooses the leftmost-column generalization, the student acquires the bug rather than the correct procedure. According to this explanation, the cause of the bug is twofold: (1) insufficiently variegated instruction, and (2) an unlucky choice by the student.

The bugs that students exhibit are important data for developing the theory. Equally important are bugs that students *don't* exhibit. When there are strong reasons to believe that a bug will never occur, it is called a *star* bug (after the linguistic convention of placing a star before sentences that native speakers would never utter naturally). Star bugs, and star data in general, are not as objectively attainable as ordinary data (VanLehn, Brown & Greeno, in press). But they are quite useful. To see this, consider again the students who are taught borrowing on two column problems, such as problem *c* above. In two-column problems, the borrow's decrement is always in the *tens* column. Hence tens-column is an inductively valid description of where to decrement. However, choosing tens-column for the decrement's description predicts that the student would place the decrement in the tens column *regardless of where the borrow originates.* This leads to strange solutions, such as *d* and *e* below:

d. $\begin{array}{r} 5 \\ 1^{1}5\;6\;5 \\ -\;\;9\;1\;0 \\ \hline 1\;6\;5\;5 \end{array}$ e. $\begin{array}{r} 15 \\ 3^{1}6\;5 \\ -\,1\;9\;0 \\ \hline 2\;6\;5 \end{array}$

This kind of problem solving has never been observed to my knowledge. In the opinion of several expert diagnosticians, it never will be observed. Always decrementing the tens column is a star bug. The theory should not predict its occurrence. This has important implications for the theory. The theory must explain why certain inductively valid abstractions (e.g., leftmost column) are used by students while certain others (e.g., tens column) are not.

These examples have illustrated the nature of the research project: trying to understand certain aspects of skill acquisition (i.e., knowledge integration, knowledge communication) by studying bugs. The next section outlines the theory.

Step theory, repair theory and felicity conditions

For historical and other reasons, it is best to view the theory as an integration of two theories. *Step theory* describes how students acquire procedures from instruction. *Repair theory* describes how students barge through problems situations where their procedure has reached an impasse.* The two theories share the same representations of knowledge and much else. I will continue to refer to them together as "the theory."

Repair theory is based on the insight that students do not treat procedures as hard and fast algorithms. If they are unsuccessful in an attempt to apply a procedure to a problem, they are not apt to just quit, as a computer program does. Instead, they will be inventive, invoking certain general purpose tactics to change their current process state in such a way that they can continue the procedure. These tactics are simple ones, such as skipping an operation that can't be performed or backing up in the procedure and taking another path. Such local problem solving tactics are called *repairs* because they fix the problem of being stuck. They do not fix the underlying cause of the impasse. Given a similar exercise later, the student will reach the same impasse. On this occasion, the student might apply a different repair. This shifting among repairs has been observed in the data. It is one kind of *bug migration*: the phenomenon of a student shifting from one bug (systematic error) to another over a short period of time.

Step theory is based on the insight that classroom learning is like a conversation in that there are certain implicit

*John Seely Brown originated repair theory (Brown & VanLehn, 1980). The present version remains true to the insights of the original version although most of the details have changed.

conventional expectations, called *felicity conditions*, that facilitate information transmission. Basically, students expect that the teacher will introduce just one new "chunk" of the procedure per lesson, and that such "chunks" will be "simple" in certain ways. Although students do not have strong expectations about *what* procedures will be taught, they have strong expectations about *how* procedures will be taught. Step theory takes its name from a slogan that expresses the students' expectations: procedures are taught "one simple step at a time." Several felicity conditions have been discovered:

(1) Students expect a lesson to introduce at most one new "chunk" of procedure. Such chunks are called *subprocedures*. This felicity condition will be described in more detail in a moment.

(2) Students *add* their new subprocedure to their current procedure rather than replacing large parts of it. That is, they expect the lesson to augment their procedure rather than making parts of it obsolete.

(3) Students induce their new subprocedure from examples and exercises. That is, students expect the lesson's material to correctly exemplify the lesson's target subprocedure.

(4) The students expect the lesson to "show all the work" of the target subprocedure. Even if the target subprocedure will ultimately involve holding some intermediate result mentally, the first lesson will write the intermediate result down. In a later lesson, the student is taught to omit the extra writing by holding the intermediate result mentally. This makes it possible for students use a simple form of reasoning to induce the relationships of the intermediate result to other parts of the procedure.

The last felicity condition, called the show-work principle, is a clear illustration of the nature of felicity conditions in general. Textbook authors probably do not consciously realize that the lessons they write obey the show-work principle. They strive only to make the lessons simple and effective. In doing so, they wind up obeying the principle. This occurs because the problem that the felicity condition solves is inherent in *any* inductive learning task. Effective learning requires its solution. The show-work felicity condition is one solution, the one used in this domain. The general point is this: Felicity conditions are conventions that have been tacitly adopted by our culture in order to make it easier for students to solve certain *inherent problems* in knowledge communication.

A competitive argument

The previous section made some assertions about human skill acquisition. Making hypotheses is only part of developing a theory. The other part is validating those hypotheses. An important validation technique used with this theory is *competitive argumentation* (VanLehn, Brown & Greeno, in press). Most competitive arguments have a certain "king of the mountain" form. One shows that a hypothesis accounts for certain facts, and that certain alternatives to the hypothesis, while perhaps not without empirical merit, are flawed in some way. That is, the argument shows that its hypothesis stands at the top of a mountain of evidence, then proceeds to knock the competitors down. This section presents an example of a competitive argument.

Consider the first felicity condition listed a moment ago. A more precise statement of it is:

> Learning a lesson introduces at most one new disjunction into a procedure.

In procedures, a disjunction may take many forms, e.g., a conditional branch (if-then-else). This felicity condition asserts that learners will only learn a conditional if each branch of the conditional is taught in a separate lesson—i.e., the then-part in one lesson, and the else-part in another.

The argument for the felicity condition hinges on an independently motivated hypothesis: mathematical procedures are learned inductively. They are generalized from examples. There is an important philosophical-logical theorem concerning induction: If a generalization (a procedure, in this case) is allowed to have arbitrarily many disjunctions, then an inductive learner can identify which generalization it is being taught only if it is given *all possible examples*, both positive and negative. This is physically impossible in most interesting domains, including this one. If inductive learning is to bear even a remote resemblance to human learning, disjunctions must be constrained. Disjunctions are one of the inherent problems of knowledge communication that were mentioned a moment ago.

Two classic methods of constraining disjunctions are (i) to bar disjunctions from generalizations, and (ii) to bias the learner in favor of generalizations with the fewest disjunctions. The felicity condition is a new method. It uses extra input information, the lesson boundaries, to control disjunction. Thus, there are three competing hypotheses for explaining how human learners control disjunction (along with several other hypotheses that won't be mentioned here): (i) no-disjunctions, (ii) fewest-disjunctions, and (iii) one-disjunction-per-lesson.

Competitive argumentation involves evaluating the entailments of each of the three hypotheses. It can be shown that the first hypothesis should be rejected because it forces the theory to make absurd assumptions about the student's initial set of concepts—the primitive concepts from which procedures are built. The empirical predictions of the other two hypotheses are identical, given the lesson sequences that occur in the data. More subtle arguments are needed to differentiate between them. Here are two:

(1) The one-disjunction-per-lesson hypothesis explains why lesson sequences have the structure that they do. If the fewest-disjunctions hypothesis were true, then it would simply be an accident that lesson boundaries fall exactly where disjunctions were being introduced. The one-disjunction-per-lesson hypothesis explains a fact (lesson structure) that the fewest-disjunctions hypothesis does not explain.

(2) The fewest-disjunctions hypothesis predicts that students would learn equally well from a "scrambled" lesson sequence. To form a scrambled lesson sequence, all the examples in an existing lesson sequence are randomly ordered then chopped up into hour-long lessons. Thus, the lesson boundaries fall at arbitrary points. The fewest-disjunctions hypothesis predicts that the bugs that students acquire from a scrambled lesson sequence would be the same as the bugs they acquire from the unscrambled lesson sequence. This empirical prediction needs checking. If it is false, as I am sure it is, then the fewest-disjunctions hypothesis can be rejected on empirical as well as explanatory grounds.

This brief argument sketched the kind of individual support that each of the theory's hypotheses has been given. Such competitive argumentation seems essential for demonstrating the psychological validity of a theory of this complexity.

Six components are involved in validation

The preceding sections indicated the kind of skill acquisition under study, sketched a few hypotheses about it, and discussed the validation method. This section summarizes the research project by listing its main components.

(1) *Learning model.* The first component is a learning model: a large, AI-based computer program. Its input is a lesson sequence. Its output is the set of bugs that are predicted to occur among students taking the curriculum represented by the given lesson sequence. The program, named Sierra, has three main parts: (i) The *learner* learns procedures from lessons. (ii) The *solver* applies a procedure to solve test problems. (iii) The *diagnostician* analyzes the solver's answers to see which bugs, if any, have been generated. The diagnostician is a part of Burton's Debuggy system (Burton, 1981). The solver is a revised version of the one used to develop repair theory (Brown & VanLehn, 1980). The learner is similar to other AI programs that learn procedures inductively. For instance, ALEX (Neves, 1981) learns procedures for solving algebraic equations given examples similar to ones appearing in algebra textbooks. LEX (Mitchell et. al., 1983) starts with a trial-and-error procedure for solving integrals and evolves a more efficient procedure as it solves practice exercises. Sierra's learner is similar to LEX and ALEX in some ways (e.g., it uses disjunction-free induction). It differs in other ways (e.g., it uses lesson boundaries crucially, while the instruction input to ALEX and LEX is a homogeneous sequence of examples and exercises). As a piece of AI, Sierra's learner is a modest contribution. Of course, the goal of this research is not to formulate new ways that AI programs can learn.

(2) *Data from human learning.* The data used to test the theory come from several sources: the Buggy studies of 2463 students learning to subtract multidigit numbers (Brown & Burton, 1978; VanLehn, 1982), a study of 500 students learning to add fractions (Tatsuoka & Baille, 1983), and various studies of algebra errors (Greeno, 1982; Wenger, 1983). The raw data are worksheets and/or protocols from students taking diagnostic tests. Their answers and scratch work are analyzed in terms of bugs. In the Buggy studies, the analysis is automated. In the others, it is done by hand. The bugs are what are actually used to test the theory.

(3) *A comparison of the model's predictions to the data.* The major empirical criterion for the theory is *observational adequacy:* (i) the model should generate all the correct *and buggy* procedures that human learners exhibit, and (ii) the model should not generate procedures that learners do not acquire, i.e., star bugs. Although observational adequacy is a standard criterion for generative theories of natural language syntax, this is the first AI learning theory to use it.

(4) *A set of hypotheses.* Until recently, most AI-based theories of cognition used only the three components listed so far: a model, some data, and a comparison of some kind. Such theories leave one to accept or reject the model *in toto*. Such an "explanation" of intelligent human behavior amounts to substituting one black box, a complex computer program, for another, the human mind. Recent work in automatic programming and program verification suggests a better way to use programs in cognitive theories: The theorist develops a set of specifications for the model's performance. These serve as the theory's hypotheses about the cognition being modelled. The model becomes a tool for calculating the predictions made by the combined hypotheses. The present theory has 32 such hypotheses. The felicity conditions listed earlier are four of the 32.

(5) *A demonstration that the model generates all and only the predictions allowed by the hypotheses.* Such a demonstration is necessary to insure that the success or failure of the model's predictions can be blamed on the theory's hypotheses and not on the model's implementation. Ideally, I would give a line-by-line proof that the model satisfies the hypothesis. This just isn't practical for a program as complex as Sierra. However, what has been done is to design Sierra for transparency instead of efficiency. For instance, Sierra uses several generate-and-test loops where the tests are hypotheses of the theory. This is much less efficient than building the hypotheses into the generator.* But it lends credence to the claim that the model generates exactly the predictions allowed by the hypotheses.

(6) *A set of arguments, one for each hypothesis, that shows why the hypothesis should be in the theory, and what would happen if it were replaced by a competing hypothesis.* This involves showing how each hypothesis, in the context of its interactions with the others, increases observational adequacy, or reduces degrees of freedom, or improves the adequacy of the theory in some other way. The objective is to analyze *why these particular* hypotheses produce an empirically successful theory. This comes out best in competitive argumentation. Each of the 32 hypotheses of the theory has survived a competitive argument.

References

Anderson, J.R. Acquisition of Cognitive Skill. *Psychological Review*, 1982, *89*, 369-406.

Anzai, Y. & Simon, H.A. The theory of learning by doing. *Psychological Review,* 1979, *86*, 124-140.

Austin, J.L. *How to do things with words.* New York: Oxford University Press, 1962.

Brown, J.S. & Burton, R.B. Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science,* 1978, 2, 155-192.

Brown, J.S. & VanLehn, K. Repair Theory: A generative theory of bugs in procedural skills. *Cognitive Science,* 1980, *4*, 379-426.

Burton, R.B. DEBUGGY: Diagnosis of errors in basic mathematical skills. In D. H. Sleeman & J. S. Brown (Eds.), *Intelligent tutoring systems.* London: Academic Press, 1981.

Buswell, G.T. *Diagnostic studies in arithmetic.* Chicago: University of Chicago Press, 1926.

Gordon, D. & Lakoff, G. Conversational postulates. In D. Davidson & G. Harmon (Eds), *Semantics of Natural Language.* Dordrecht: Reidel Press, 1975.

Grice, H.P. *Logic and conversation.* In D. Davidson & G. Harmon (Eds), *Semantics of Natural Language.* Dordrecht: Reidel Press, 1975.

Greeno, J. Personal communication, 1982.

Mitchell, T.M., Utgoff, P.E., & Banerji, R.B. Learning problem-solving heuristics by experimentation. In R.S. Michalski, T.M. Mitchell, & J. Carbonell, (Eds.), *Machine Learning.* Palo Alto, CA: Tioga, 1983.

Newell, A. & Rosenbloom, P. Mechanisms of skill acquisition and the law of practice. In J.R. Anderson (Ed.) *Cognitive skills and their acquisition.* Hillsdale, NJ: Erlbaum, 1981.

Neves, D.M. *Learning procedures from examples*, Pittsburgh, PA: Carnegie-Mellon University, Department of Psychology, unpublished doctoral dissertation, 1981.

Tatsuoka, K. & Baille, R. Personal communication, 1983.

VanLehn, K. Bugs are not enough: Empirical studies of bugs, impasses and repairs in procedural skills. *Journal of Mathematical Behavior*, 1982, *3*, 3–72.

VanLehn, K., Brown, J.S. & Greeno, J.G. Competitive argumentation in computational theories of cognition. In W. Kinsch, J. Miller & P. Polson (Eds.) *Methods and Tactics in Cognitive Science.* New York: Erlbaum, forthcoming.

VanLehn, K. Felicity conditions for human skill acquisition: Validating an AI-based theory. Cambridge, MA: Massachusetts Institute of Technology, unpublished doctoral dissertation, 1983.

Wenger, R. Personal communication, 1983.

*It takes Sierra about 100 hours of Dorado time to process a single subtraction lesson sequence. This style of research would be infeasible without fast Lisp machines, such as the Dorado.