ED 263 881 IR 011 867

AUTHOR	Streibel, Michael J.
TITLE	A Critical Analysis of Computer-Based Approaches to Education: Drill-and-Practice, Tutorials, and Programming/Simulations.
PUB DATE	Apr 85
NOTE	65p.; Paper presented at the Annual Meeting of the American Educational Research Association (69th, Chicago, IL, March 31-April 4, 1985).
PUB TYPE	Viewpoints (120) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC03 Plus Postage.
DESCRIPTORS	*Computer Assisted Instruction; *Computer Simulation; *Computer Software; *Drills (Practice); Futures (of Society); Man Machine Systems; Mastery Learning; Problem Solving; *Programed Tutoring; P.ograming; Teaching Methods
IDENTIFIERS	*Computer Uses in Education; *Tutorial Mode

ABSTRACT

Three major approaches to the use of computers in education are examined, serious limitations of each are presented, and questions are raised as to the efficacy of technologolizing education. The drill and practice approach is shown to embody a deterministic, behavioral technology that turns learning into a systematically-designed and quality-controlled form of work. Computerized tutorial programs are shown to extend the behavioral and technological approach to learning even further by shaping interactions via an external agent's intentions in order to maximize the learner's performance gains. Most seriously, computerized tutorial interactions pre-empt the personal intellectual agency and ultimately inner-directed learning. Finally, the use of computers is shown to limit the learner's mental landscape to objective, quantitative, and procedural tools. A list of references completes the document. (JB)

ED263881

A CRITICAL ANALYSIS OF COMPUTER-BASED APPROACHES TO EDUCATION: DRILL-AND-PRACTICE, TUTORIALS, AND PROGRAMMING/SIMULATIONS*

by

Michael J. Streibel, Ph.D.

Assistant Professor

Educational Technology Program

Department of Curriculum & Instruction

528F Teacher Education Building

The University of Wisconsin

Madison, Wisconsin 52706

(608) 263-4674

(608) 263-4600

*Presented at the annual meeting of the American Educational Research Association, Chicago, Illinois. Mar. 31 - Apr.4, 1985.

IR011867

A Critical Analysis of Computer-Based Approaches to Education: Drill-and-Practice, Tutorials, and Programming/Simulations

General Introduction

Microcomputers are being introduced into public and private schools at an exponential rate and this trend shows no signs of abating (Becker, 1983). It is therefore crucial that we examine how computers are used in education and what implications this has for the furture of education (Sloan, 1984). In this paper, I will present a critical analysis of the three major approaches to the use of computers in education: the drill-and-practice approach, the tutorial approach, and the simulation and programming approach. I will do this by presenting the characteristics of these approaches and then by examining their assumptions and contexts-of-use. Finally, I will try to uncover a common framework that runs throughout these approaches and discuss the implications of this framework for education. The analysis will not, however, deal with the social, political, or economic dimensions of education even though such dimensions are ultimately involved (Apple, 1979, 1982).

A number of philosophical questions helped guide me in the course of this analysis. For example, what is the logic behind each of the computer-based approaches to education mentioned above and how does this logic express itself in a learning situation? These and other questions helped me understand that computerized drill-and-practice approaches to education are a behavioral form of learning technology which may not be the best way to supplement instruction. That is, computerized drill-and-practice may run counter to the dialectics of learning. I also asked myself how the various approaches mentioned above treated the human learner and what consequences this could have for education. My analysis revealed that a concept of an individual as a generic information-processor was embedded in each of the three approaches. This, in turn, revealed how system's goals (e.g., efficient production of learner performance) were invariably masked behind the rhetoric of individual "mastery" and individual "needs." It also revealed that the learner's personal intellectual agency was decreased rather than increased in the case of computer-based tutorials even though the rhetoric promised the exact opposite.

Finally, I asked myself how the "intellectual tool" use of computers helped or hindered us to formulate, understand, and solve problems. That is, what kinds of "solutions" were legitimized and delegitimized when we formulated problems in computable forms? This was part of the larger question about the potential that computer languages and simulations had for thinking and learning about problems. My answer to this question required a very careful analysis of the "expressive potential" of computer languages.

I have explored the underlying assumptions of the three most common approaches to the use of computers in education in order to understand the potentials and limitations of this technology within education. This should provide a basis for further critical study of the subject.

1. Drill-and-Practice Computer Programs

Drill-and-practice courseware programs (i.e., computer programs that guide learning with a drill-and-practice instructional strategy) are the dominant use of computers in education today (Suppes, 1966; Coburn *et al*., 1982). They are currently run on mainframe, time-sharing, and micro-computers (Murphy & Appel, 1977; Alderman, 1978; Stewart, 1979). I will deal with this topic by first describing the major characteristics of the drill-and-practice approach and then by discussing a number of related issues. The latter issues, however, warrant some preliminary remarks.

The most general issue deals with how drill-and-practice courseware programs are biased towards behavioral forms of learning. I will demonstrate this by showing how such courseware programs combine several broad traditions such as the mastery-learning paradigm, the philosophy of individualized learning, and the concepts of educational work and efficiency. I will then show how such programs represent a very one-dimensional form of education because they restrict the goal structures, reward structures, and meaning structures of educational events to the domain of educational productivity. This, in turn, undermines the possibility of integrating sub-skill performances into other educational experiences.

Finally, I will deal with the issue of how drill-and-practice courseware programs work best in a learning culture that legitimizes behavioral performances over other types of educational goals. Ironically, behaviorally-oriented learning cultures must still be created, mediated, and sustained by interpersonal interactions which have the potential for forming alternate cultures (and thereby alternate types of educational goals). Drill-and-practice courseware programs, however, do not permit these alternate goals to develop. My discussion of the importance of dialogue will then lead to an analysis of the "dialogue" component of human-computer tutorial interactions.

Description of Drill-and-Practice Courseware

Drill-and-practice courseware programs make a number of assumptions about instruction (Bunderson, 1981; Salisbury, 1984):

a. previous instruction in the concept or skill has already taken place,

b. regular instruction is only being supplemented and not replaced,

c. instruction is to follow a controlled, step-by-step linear sequence of sub-skills according to an algorithm embedded in the computer program. This algorithm does not constitute a model of a student or an expert but constitutes a model of rote skill-building in the case of drill and a model of patterned skill-building according to the logic of the content and an instructional theory in the case of practice (Skinner, 1968; Gagne *et al.*, 1981),

d. there exists a right/wrong answer dichotomy in the logic of the content,

e. the basic unit of instructional interaction is a question-answer-branch episode (Dennis, 1979b). Continuous learner responses in the form of correct answers are therefore expected, and,

f. the best feedback by the program from an intructional point of view is an immediate check on a student's responses according to the logic of the content:

   i. positive feedback when the answer is correct,

   ii. corrective (rather than judgemental) feedback when the answer is incorrect.

The characterictics described above make several things clear: drill-and-practice courseware programs are designed to provide immediate corrective interventions in the learning process when continuously-monitored performance-measures indicate incorrect responses. The learner is viewed as a "black box" and his or her behaviors are shaped by an external, mechanical process (i.e., by an instructional algorithm that uses feedback mechanisms to guide the learner towards a pre-specified behavioral goal). Drill-and-practice courseware programs therefore constitute a deterministic form of behavioral technology (Skinner, 1968). This may be adequate for beginning skill-building but not for higher levels of learning (Dreyfus & Dreyfus, 1984). Within the beginning skill-building domain, however, drill-and-practice programs do result in significant performance-gains (Kulik, Kulik, &

Cohen, 1980).

The characteristics described above also make another thing clear: drill-and-practice courseware programs are a one-dimensional form of education. The reasons for this are simple and take us into broader issues:

a. they contain a uni-dimensionality of goal structures since:

    i. they only focus on pre-specified behavioral performances,

    ii. they exclude non-behavioral educational goals (e.g., emergent attitudinal and cognitive strategy outcomes),

    iii. they are designed without any interaction with the intended learners,

b. they contain a uni-dimensionality of reward structures because:

    i. they only reward successful performance on one sub-skill with an opportunity to work on the next sub-skill,

    ii. they define "individualized feedback" in terms of randomly-selected, generic messages rather than in terms of personal, semantic, and affective engagement,

c. they contain a uni-dimensionality of meaning structures because:

    i. they only define "mastery" (i.e., success at any point) by the quantitative aspects of performance (e.g., total number of correct responses) and not by the qualitative aspects of performance (e.g., expert performance).

In order to understand the consequences of this one-dimensionality for education, I will delve into the philosophy behind drill-and-practice courseware programs as well as into the context of their use. I will do this by examining the mastery-learning paradigm, the philosophy of individualized learning, and, the concepts of educational work and efficiency.

Mastery Learning and Drill-and-Practice Courseware

The mastery-learning paradigm assumes that most students can learn most things to a specific level of competence in varying amounts of time (Bloom, 1976). It therefore bases differences in the amount of learner-performances at any point in time on differences in the rates of learning. Second, the mastery-learning paradigm assumes that instruction can be consciously designed to guarantee specific outcomes. It therefore places a heavy emphasis on the quality of instructional materials. Finally, the mastery-learning approach uses criterion-referenced tests with each objective to decide whether a student has met the criterion of success. Students are then permitted to go on to the next set of objectives.

You can see from this description that the mastery-learning paradigm closely resembles a rationally-managed input/output model of educational performance (Apple, 1979, 1982). It therefore makes a number of assumptions about the pedagogical principles, classroom practices, and instructional arrangements involved (Barr & Dreeben, 1978; Bolvin, 1982). These assumptions elaborate the input/output model:

a. pedagogical principles:

The mastery-learning paradigm assumes that students vary in their aptitude, ability to understand instruction, and perseverance. These factors contribute to the distribution of performances by students at any given point in time. Although all students are believed to be able to achieve mastery _given enough time_, the amount of time it takes to achieve mastery is normally-distributed. In comparison, in a norm-referenced learning paradigm, the actual performance-scores are also distributed normally at any particular point in time but there is no claim that low-achievement students will _ever_ reach higher levels of performance (Bloom, 1971, 1976).

The pedagogical assumption that all students are able to achieve mastery given enough time allows learning to be treated as a rationally-managed process since only time and resources have to be considered in order to guarantee a predictable outcome (i.e., levels of student performance). Considerations such as the dialectics of learning, accommodation to individual uniqueness, and the possibility of emergent goals have been factored out of the process. I will show later that these considerations are essential for learning - even at the level of simple skill-building.

b. _classroom and school practices_:

The mastery-learning paradigm manipulates the _time_ allowed for learning and the _quality of instructional stimuli_ as the main factors to help students achieve mastery. It therefore entails the rational planning of classroom and school time, schedules, organization, and conditions of instruction, as well as the rational design of instructional materials (Nunan, 1983). Such design and planning activities are only called rational to the extent that they are guided by the pragmatics of instructional and organizational theories. Such design and planning are not guided by the pragmatics of classroom teaching (Wolcott, 1977). For example, predictability and manageability of process and product are a prime consideration and not whether some unique classroom event becomes an occasion for further learning. Hence, the conception of instructional events is separated from the execution of such events and the conceptual phase directs and controls the execution phase (Apple, 1982). In this scheme of things, we would eventually expect teacher performance (and ultimately administrator and even system performance) to be evaluated in terms of student performance since correct student behavior is the ultimate output. However, I will show later that this in fact serves the system's needs more than the learner's needs.

c. instructional arrangements:

The mastery-learning paradigm follows a number of procedures to guarantee that students will perform at pre-specified levels (Bloom, 1976):

i. pre-instructional assessment procedures are used to measure the presence or absence of pre-requisite knowledge in the learner,

ii. initial teaching methods are used to inform the learner of the objectives and pre-requisite knowledge,

iii. training procedures are used to help the learner acquire the appropriate knowledge and skills,

iv. continual assessment procedures are used to assure the presence or absence of sub-skills,

v. immediate remediation procedures are used if the sub-skills are not present, and,

vi. certification of mastery is added when some pre-determined criterion performance is reached by the learner.

You can see from all of these techniques that the mastery-learning paradigm conceptualizes the instructional process in quality control terms. Each step in the paradigm is expressed as a procedure and all instruction is arranged to maximize an output. All that is left for humans to do is to get to work!

The mastery-learning paradigm described above therefore provides a broad theoretical framework for drill-and-practice courseware. It specifies what kinds of things are to achieved (i.e., measurable performance gains) and how these things are to be achieved (i.e., through the manipulation of time and instructional stimuli). Drill-and-practice courseware functions as the training and remediation component within this framework. Although group work is permissable within the mastery-paradigm, drill-and-practice courseware is usually individualized. This leads me to the philosophy of individualization.

Individualization and Drill-and-Practice Courseware

Individualization can mean many things: independent study, individual pacing, individual diagnosis, individual educational outcomes, etc. (Bolvin, 1982). It arises out of the larger recognition that individuals differ from each other (Sperry, 1972; Messick, 1976; Cronbach & Snow, 1977). Within computer-based forms of individualized learning, however, it refers to generic outcomes for generic individuals rather than to personal goals for unique individuals. More of this later.

The philosophy of individualization, as it has developed within individualized systems of education, contains a number of specific assumptions (Lukes, 1973; Talmadge, 1975):

a. a belief that each person has a unique set of characteristics or aptitudes that ultimately influence the rate at which competent performance in a particular skill is achieved,

b. a belief that a well-defined and well-structured sequence of instructional events can be designed for each individual to facilitate their progress towards pre-planned outcomes,

c. a belief that only time and quality of instructional materials will influence successful completion of an objective,

d. a belief that assessment procedures of student "needs" and characteristics will indicate a readiness for the objectives by the learner,

e. a belief that an evaluation mechanism can be found for constantly monitoring the student's progress towards a pre-planned outcome (these same evaluation mechanisms also provide data for the instructional system's performance), and,

f. a shift in the role of the teacher from a pedagogical one towards that of instructional decision-maker (i.e., placement of students; selection, use, and allocation of space, time, and materials; data collection and report-writing, etc.).

These assumptions are implemented in a number of ways. In some systems of individualized learning such as the Keller Plan, students have a great amount of flexibility in setting schedules, getting help from student tutors, and following any number of paths towards a pre-defined goal (Keller, 1968). Each of these factors is adjusted for the sake of individual rates of learning. In the Skinnerian version of the philosophy of individualization, on the other hand, instruction is broken down into much smaller units, and, instructional events are more controlled and automated (Skinner, 1968). It is this latter version that has become implemented in drill-and-practice courseware.

Drill-and-practice courseware programs overwhelmingly use rate-of-progress as their major dimension of individualization although they sometimes also include level of difficulty (Suppes, 1966). Other dimensions such as cognitive style are often called for but rarely implemented because of the difficulty of specifying these factors in computable form (Scriven, 1975).

Drill-and-practice courseware programs also break the instructional process into very small steps (called "frames"). They then assess each response by the learner and specify a finite number of paths for the learner to follow. They are therefore "individualized" in a very narrow sense of the term. That is, they have restricted the meaning of individualization to a finite set of choices within a measurable and computable domain (i.e., individual rates of progress along finite, forced-choice paths that lead to pre-specified, measurable outcomes). Since they control both the presentation of information as well as the learner's interactions with that information, they control the individual's total attention during the time that they are used.

Finally, drill-and-practice courseware programs relegate the teacher to a managerial function (e.g., resource manager or exception handler) (Boyd, 1983a, 1983b). This is not to suggest that a teacher or school system is forced by the computer to organize the classroom according to the Skinnerian philosophy of individualization, only that drill-and-practice courseware programs are biased towards such an orientation. In many schools, the shift towards teacher-as-manager has already taken place without the computer (Apple, 1975; Berliner, 1982).

In many ways, the philosophy of drill-and-practice courseware is consistent with the movement in the curriculum field towards the "technical control" of learning (Kliebard, 1971; Franklin, 1974, 1982; Carlson, 1982). I will describe this notion under the rubric of the "technological framework" throughout the rest of my paper. For now, the concept of replicated work will help us begin to understand the concept of "technical control" in courseware.

Courseware as Replicated Work

Victor Bunderson has developed one of the most thorough analysis of the concept of courseware-as-replicated-work (Bunderson, 1981). The following discussion owes much to his analysis and hopes to add to his ideas. Computer courseware, he argues, has both a product and a process dimension. As a product, courseware consists of the "consumable [materials of instruction] that operate on and with a technologically-mediated instructional delivery system." As a process, courseware constitutes (Bunderson, 1981):

> an economically replicable and easily portable package which when used in combination with a technologically-mediated instructional delivery system, is capable of performing work related to training and performance improvement.

An instructional delivery system, on the other hand, consists of both the physical objects and structures "designed to perform or facilitate the work necessary to achieve educational and training goals," and, a human culture of "traditions, values, and habits that inform and constrain the use of the physical artifacts" (Bunderson, 1981). The instructional delivery system is therefore the broadest category within which all other components of the ensemble are situated and from which they derive their meaning.

We can see from the description above that the concept of computer courseware contains a "technological control" orientation. That is, the technical structure of the delivery system shapes the form and function of the human culture and the physical artifacts. The technical structure also orients these components towards some external goal (i.e., educational performance) and then tries to maximize the levels of this goal. A technological delivery system will therefore ultimately influence the nature of the classroom culture - unless, of course, the classroom is already organized as a work culture.

From my earlier discussion, we can also begin to see that the student's control over the pace of learning is really a form of pseudo-control because he or she can only choose from a finite number of paths towards a pre-determined goal. Bunderson acknowledges this somewhat by saying that "learner-centered will emphasize learner productivity, not necessarily learner control" (Bunderson, 1981). This restricts the meaning of "individualized learning" to that of "individualized productivity level."

Bunderson continues his discussion of educational work by criticizing the inability of current teacher-centered "delivery systems" to be more "productive." The teacher-centered culture, he argues, has reached the "limits of [its] improvability." His solution is a technological one: "when education is analyzed into the work that is required, technology is seen as the <u>only</u> way to make a fundamental difference" (Bunderson, 1981). Bunderson's argument is very general and even applies to book-based technology (i.e., a teacher with books can accomplish more than a teacher without books). His argument has to be analyzed very carefully, however, for both book-based as well as for computer-based technologies.

Before turning to my analysis, some general comments are in order. First of all, conceiving of the classroom as an "instructional delivery system" (rather than, say, an instructional setting for the dialectical encounter of mutually-respected and unique individuals) narrows the debate about what can happen in such a setting. The classroom, in effect, becomes a place for training and development. I mentioned earlier that the mastery-learning paradigm turned the classroom into a workplace for both teachers and students. The concept of the classroom as a workplace is further compounded by highlighting the work potential of classroom technologies. Second, the discussion by Bunderson about classroom instructional delivery systems lays the conceptual groundwork for accepting computer courseware as more efficient versions of the same thing.

Looking at books as "productivity tools" has a number of problems - some of which also apply to computers:

a. what really counts in books (even for the restricted productivity paradigm of education) is the <u>intelligence</u> embodied in the print medium and <u>not</u> the presence or quantity of print information. This intelligence goes beyond procedural knowledge and is the key to what is transmitted during learning. Hence, teachers with unintelligent books are no better off than teachers without books. They may, in fact, be worse off because they might be tempted to assume that they can accomplish more than before,

b. the fact that teachers with books can accomplish more than teachers without books does <u>not</u> mean that the "work potential" of book-based technology <u>determines</u> what a teacher does. True, reading skills might be required simply because books are used but this does not mean that books need only be used for training. Conceptualizing books in terms of their work potential, however, forces them into a productivity scheme,

c. analysing books in terms of their work potential diverts attention away from their dialectical potential (i.e., their ability to confront one with alternate points-of-view from one's taken-for-granted reality). Books, for example, also have the ability to reveal how other humans have integrated the dialectical tension between, say, justice and love or between wisdom and courage. In these instances, books become guides for experiential learning that point the way to personal and communal integration. All of this is hidden when we look at books in terms of their work potential. We are then only left with an image of a book as a training aid,

d. and finally, books as such leave open the choice of interactions which a person can carry out. Hence, books leave open the way an individual uses and makes sense out of the information. They are therefore not a good analogy with computer courseware programs which control both the presentation of information as well as the user's interactions with that information.

Bunderson's argument permits one to compare teacher-delivered instruction and technology-delivered instruction because both are conceptualized in similar terms. Computer courseware is then seen as a more efficient mechanism. I must admit that when I observe certain classrooms which contain hierarchical authority, rigid schedules, and mindless workbooks, I am inclined to agree with Bunderson. But I must also point out that even in such classrooms, students still have some opportunity for personal integration of experiences and skills. That is, students can still integrate their drill-and-practice activities with exploration, planning, and collaboration - even if it is done as a subtrafuge. This is essential for education (Kolb, 1984). In individualized drill-and-practice courseware, on the other hand, where a student's total time, attention, and interactions are controlled by the computer, such integrations are no longer possible (Dreyfus & Dreyfus, 1984). Is this loss worth the price - even for low levels of learning where only the acquisition of procedural rules is involved?

When we examine the actual characteristics of Bunderson's concept of educational work, we find that they all embody the extensional side of education (i.e., the measurable and procedural). For example, according to Bunderson, a teacher: presents information to students, models processes for students, provides students with trials and feedback, discusses "individual needs" with students, uses affective appeals to motivate learning, trains students how to use the delivery system, assesses student performance, manages the assessed information, and, manages classroom interactions. Where, here, is there a teacher's affective and semantic engagement with students beyond maximizing performance gains? All that Bunderson describes are procedural skills and information-processing functions. When conceptualized in this way, such functions _can_ be carried out more efficiently and effectively by technology (e.g., video, microcomputer, etc.). Efficiency here means maximizing educational productivity at the lowest financial cost. Effectiveness means reliably reproducing the process and the product (Bunderson, 1981). Efficiency and effectiveness, in effect, are no longer subject to the qualitative criteria of excellence and expertise within a particular subject area but to the quantitative criteria of economics. No wonder teachers cannot compete!

There is a fundamental contradiction in Bunderson's point of view, however. The rhetoric stresses the "needs" of the individual but the terms of the debate emphasize instructional systems concerns. By shifting the educational interactions away from the intensional logic of interpersonal interactions and towards the extensional logic of procedural skills and information-processing functions, the following criteria are emphasized:

a. _systems efficiency_ (i.e., maximizing the throughput of students for the time and resources invested - rather than developing individual talents),

b. _systems reliability_ (i.e., quality control and replicability of output - rather than the establishment of individual, communal, or cultural diversity), and,

c. _systems economy_ (i.e., more scholar for the dollar - rather

than personally-determined pursuit of excellence).
Hence, only those "individual needs" amenable to systems' logic are served.

There is another contradiction in Bunderson's concept of educational work. The very work culture which has to exist in a classroom for a technological instructional system to operate can only result from intensional human engagement, negotiation, and interaction. Students and teachers are therefore essential and continuous agents in the creation of a classroom work culture (Sarason, 1982). However, the very processes that are required to produce the work culture then have to be denied because they contradict the technological framework. This is the case because an instructional delivery system embodies a technological culture that tries to shape the human culture to its own ends whereas human cultures shape their own ends (Nunan, 1983).

A good illustration of the contradiction described above can be seen in the teacher roles which Bunderson believes will predominate in a technological environment: corrector of imperfect and outdated information, illustrator and augmentor of delivered instruction, illustrator and augmentor of the expert algorithms embodied in the instructional system, creator of a technologically-acceptable setting, and, interpreter of automatically-tested and recorded results. Each of these roles shows how the technological delivery system has become the central organizing factor in classroom life. The classroom has thereby been structured as a workplace by someone other than the teacher (Wollcott, 1977;

Nunan, 1983). A simple drill-and-practice courseware program is therefore not all that innocent an aid to teaching in a classroom community (Benne, 1975). In fact, such programs may ultimately conflict with the nature of teaching because teaching is not a highly rational, decision-making affair (Jackson, 1968).[1]

Summary of Drill-and-Practice Courseware

Drill-and-practice courseware programs embody a deterministic form of behavioral learning technology. They also embody narrow aspects of the mastery-learning paradigm, the philosophy of individualization, and the concepts of educational work and efficiency. That is, they convert the learning process into a form of work that tries to maximize performance-gains and they restrict the meaning of individualism to rate-of-progess and level-of-difficulty (and ultimately to individualized productivity level).

Computerized drill-and-practice courseware programs may also not be the best way to supplement instruction even though they maximize sub-skill performance. The reason for this is simple: drill-and-practice courseware programs restrict the type of interactions involved to a decontextualized performance-domain and diminish integration of sub-skills with higher-level skills.

Finally, drill-and-practice courseware programs are part of a behavioral learning culture that mitigates against non-behavioral goals. Hence, such programs do not lead to critical thinking or personal empowerment. The question therefore arises whether computer-based tutorials have such a potential or whether they simply develop the behaviorally-oriented learning philosophy in a more sophisticated way. Many authors have argued that computer-based tutorials do in fact solve some of the limitations of drill-and-practice programs (O'Neil & Paris, 1981).

2. Tutorial Computer Programs

Do tutorial courseware programs go beyond drill-and-practice approaches to education? They do in an obvious sense because they are intentionally designed to "take total responsibility for instruction" and to contain a "mixed-initiative dialogue" (Dennis, 1979a; Bork, 1980a; O'Shea & Self, 1983). But what exactly do the terms "dialogue" and "initiative" mean here? I will examine the nature of "dialogue" in human-computer interactions as a way of analyzing the nature of tutorial programs. My reason for this is simple: dialogue is seen by many authors to be the basic building block for higher levels of learning (Freire, 1973, Greene, 1978; Shor, 1980). I will also examine the types of "quality-control" procedures which are used and the nature of the tutorial engagement. I will then show that many of the themes which I uncovered in my analysis of computerized drill-and-practice re-emerge in computerized tutorials in a more sophisticated form. This will stand in sharp contrast to the rhetoric about tutorial courseware programs which claims that such tutorials resemble real conversations and real teaching (Bork & Franklin, 1979; Dennis, 1979a; Bork, 1980b).

Before dealing with these larger issues, however, I would like to describe the various types of human-computer interactions in tutorial courseware programs: on-line tests, remedial dialogues, and interactive proofs (Bork, 1980b).

On-line tests are initiated by a computer as part of the tutorial interaction. They involve a comparison between a model of the student (which was either pre-programmed into the computer or constructed by the computer on the basis of student performances) and a model of an expert. In simpler tutorials, on-line tests only involve a comparison between student performances and pre-specified, content-determined performance-levels (O'Shea & Self, 1983). In either case, however, on-line tests provide continual diagnoses of students' performances.

An immediate consequence of having on-line tests in computerized tutorials is that the learner is subject to constant "quality control." This does not seem unreasonable since, in interpersonal interactions, humans also check out their inferences about each other (Nisbett & Ross, 1980). Why shouldn't a computer do the same? However, in interpersonal dialogues, such monitoring takes place in the context of semantic engagement and conjoint intentions. In human-computer interactions, on the other hand, such monitoring is guided by an external agent's (i.e., author, instructional designer, or programmer) intentions which are fixed for the duration of the interaction. These external intentions establish pre-set, non-negotiable, and measurable performance outcomes for the learner. Constant monitoring is therefore not intended to understand the learner and his or her messages (as in interpersonal dialogue) but rather intended to guarantee a behavioral outcome. In drill-and-practice courseware, this was rather obvious. In "mixed-initiative" computer "dialogues," this is not always so evident.

The constancy and immediacy of diagnosis and feedback in on-line tests has several other consequences:

a. it emphasizes accretion learning because the computer is constantly assessing evidence of normal progress towards a pre-specified goal. This tends to discourage "messing around" with the subject matter because "messing around" behavior is not evidence that a learner is building up an experiential basis for a quantum leap of understanding (Hawkins, 1974),

b. it tends to focus learning on generic <u>means</u> as well as generic <u>ends</u> in spite of the fact that tutorials are individualzed. The reason for this is simple: by continually measuring and diagnosing educational performances with context-free response-analyses, on-line tests pre-empt personally-constructed means. The computer, in effect, <u>controls the means as well as the ends</u> and constitutes a powerful "other" that structures and dominates the entire interaction (Weizenbaum, 1974; Scheibe & Erwin, 1979; Gardner, 1979; Turkle, 1984).

In interpersonal interactions, on the other hand, learning tends to focus on the ends-in-view and not on the means (Polanyi, 1958; Greene, 1978). Hence, learning incorporates personally-constructed means and meanings. This is not to suggest that human teachers cannot dominate an interaction with a learner but only that learners have the opportunity to develop personal ways to reach a particular goal, and,

c. it tends to accelerate the learning process because it creates a set of temporal expectations (notice that rate of learning was the major dimension of individuality and faster rates were considered better because of the efficiency orientation of the system). This in turn, biases the tutorial interaction against reflectiveness and critical thinking (Freire, 1973; Shor, 1980). Some courseware authors have suggested that this bias can be countered by using "individualized" wait loops in the courseware programs (Shneidermann, 1980). However, reflectiveness is not a matter of waiting longer.

Remedial dialogues are initiated by the computer when the learner's performance does not match some pre-specified performance-criteria (Bork, 1980b). They assume that the student already knows the area and can work with the information which is presented. Here again, this parallels what happens in interpersonal dialogues but with some very important distinctions:

a. in interpersonal tutorials, remedial dialogues are initiated by a teacher on the basis of his or her tacit knowledge about the unique characteristics of the learner. Furthermore, the teacher tries to understand the learner's state of mind by "thinking like the student" in order to unravel the student's conceptual bind or misunderstanding (Hawkins, 1974). This is a unique, constructive, and intentional act of empathy and engagement by the teacher and <u>only nominally entails the student's behaviors</u>,

b. in human-computer tutorials, remedial "dialogues" are initiated on the basis of a set of explicit and computable rules (expert or content-related algorithms) (Sleeman & Brown, 1982; O'Shea & Self, 1983). The tutorial courseware program, in effect, constitutes a generic, rule-driven process that engages an internal, <u>generic</u> model of the learner. The actual human agent (i.e., student) in this "dialogue" only provides the data for the computer's generic model of the learner. This is even the case when the computer builds up a model of the student based on a history of student performances. The resulting model of the student is still a formal, rule-oriented, generic model. Remedial "dialogues" therefore do not involve <u>this</u> student but rather this <u>type</u> of student. I will explain this in greater detail below.

<u>Interactive proofs</u> are a type of computer-based tutorial that permit the learner to make decisions beyond a pre-defined set of choices (Bork, 1980b). Hence, students can ask for information, work through a variety of examples that embody some concept, and even construct their own models of the problem. However, the very nature of the computing environment still constrains the terms of the debate. The best examples of interactive proofs

usually come from mathematics and science where the nature of the content parallels the nature of the computing environment (i.e., a computable formula captures both). This in turn permits an author (or instructional designer or programmer) to create an interactive proof out the formula. Of course, even non-mathematical subject areas can be reformulated to be amenable to interactive proofs. Hence, the socio-political problem of hunger can be recast into economic terms and then reduced to a formula that relates an arbitrarily-chosen set of variables. The interactive "proof" then proceeds <u>as if</u> it were a mathematical problem. This ultimately treats a problem such as hunger as if it were a computerized numbers game - no matter how complex the mathematics. However, computational complexity will <u>never</u> match real-world complexity unless the processes in the world are controlled by comparable mathematical or procedural rules.

<u>Human-computer "Dialogues" in Tutorial Courseware</u>

We now come to the central assumption of computer-based tutorials. Namely, that human-computer "dialogues" should resemble interpersonal conversations. Bork has modified this claim somewhat by saying that the student is really dialoguing with the author of the computer tutorial rather than with the computer itself, but this is a facile reformulation. Bork also admits that the author of a computer tutorial is trying to manipulate the student by "stimulat[ing] meaningful responses which contribute to learning" (Bork, 1980b). Hence, human-computer "dialogues" are a form of behavioral technology where dialogic interactions are controlled by an author who is not part of the actual interaction. Responses are only meaningful in light of their contribution to educational performance-gains. The actual confrontations between humans and computers are therefore one-sided affairs because the computers have fixed goal structures, interactive strategies, and deductive capabilities.

What should human-computer interactions in a tutorial be called? To answer this we have to compare them with interpersonal interactions. Interpersonal dialogues contain an essential component of conjoint control (in spite of the power differentials that may exist between students and teachers). Such conjointness is missing in human-computer interactions. In human-computer "dialogues," students only control the:

a. rate (i.e., pacing of pre-defined sequences),

b. route (i.e., any one of a finite number of predefined, or algorithmically-constrained, paths towards a pre-defined goal), and,

c. timing (i.e., speed of individual responses).

All other control resides in the courseware program. Students therefore only have a form of pseudo-control because the actual interaction follows a pre-planned, goal-oriented, procedural network. Hence, human-computer tutorial interactions are best called "utilogs" rather than "dialogues" (Sheiderman, 1980). Of course, utility here is defined by a courseware author who in turn is restricted to certain categories within the technological framework (Ellul, 1980; Turkle, 1984).

What are the deeper implications of having utilogic interactions in a computer tutorial shaped by the external intentions of an author? I will summarize a number of these implications for education by comparing them with interpersonal interactions:

a. humans are treated as data-based, rule-following, symbol-manipulating, information-processors:

This implication emerges from the nature of the computer technology that is used to carry out the actual tutorial interactions. Machine processes can only operate on explicit information according to algorithmic rules (Weizenbaum, 1976). Computers cannot semantically or affectively engage human beings. Humans therefore have to adapt to the nature of the computational environment - although within that framework, computer processes can be designed to adapt to the "individual differences" of humans.

Earlier, I mentioned that computers only engage data from an individual and not the actual person. This data is organized by the program into a model of the learner (in simpler programs, this is merely a data-base of variables and values). The particular model of an individual which the program contains (or builds up) always remains a formal and abstract type. Furthermore, this model is a means for the computer to carry out the interaction. The human is therefore treated by the computer as a generic type and a means to an end. This is even the case when many models of students (i.e., many generic types) are programmed into the computer. This has serious implications for education.

Since humans beings develop personal intellectual agency through dialogic interactions (Greene, 1983), the learner in computer-based tutorial interactions can _never_ develop such agency. The only control a learner ever has is a form of pseudo-control within a technological framework. More significantly, since human beings tend to model the "other" in dialogic interactions (Scheibe & Erwin, 1979), computer-based tutorials may actually teach students to treat other dialogic partners as _anthropomorphized processes_ and _means_ rather than as ends (Turkle, 1984).

In interpersonal dialogic interactions, on the other hand, individuals encounter, confront, accept (to a greater or lesser degree), and engage each other as unique individuals. True discourse, in this case, requires the acceptance of the "other" in the discourse as a unique and intentional being. It also requires a similar image of self. The individual, in interpersonal dialogues, is therefore a _unique ontological entity_ (rather than a generic type) and an _end_ (rather than a means). This sets the stage for personal agency in learning. Interpersonal dialogue can, of course, become mechanical if the humans involved act on the basis of some stereotypical inferences about each other. However, the potential for true discourse is _always_ present in interpersonal interactions. This potential can _never_ exist in human-computer interactions (Weizenbaum, 1976).

Another consequence of viewing humans as rule-following information-processors (as opposed to individuals with unique intentionalities) is that uniform educational goals, methods, and outcomes are legitimized. Uniformity in education is enforced not only because the instructional systems attempt to shape a uniform product (i.e., pre-specified learning outcomes) but also because the very conceptualization of the individual places "semantic and syntatical constraints on acceptable language for the discussion of human beings" (Strike, 1984). This in turn makes it impossible to express and legitimize other conceptions of human beings, educational goals, and methods outside of the technological framework (Ellul, 1980). A simple example of an alternative framework can make this clear.

In order to create a community and carry out community action, we must (Newman & Oliver, 1967):

i. unconditionally accept individual uniqueness and diversity,

ii. carry out an on-going dialectical synthesis of opposing viewpoints with the actual members of the community, and,

iii. respect emergent community goals.

The technological framework makes this perspective impossible to conceptualize (let alone operate) because it is based on an opposing set of assumptions. The community framework builds on three things: the uniqueness of each individual amd his or her viewpoint, a dialectical rationality that tries to synthesize opposing views, and, emergent community goals. The technological framework, on the other hand, builds on: the generic characteristics of individuals, a means-ends rationality, and, a pre-determined set of performance goals.[2]

b. machine processes will eventually match human processes:

This second implication derives from the first implication: if humans are ultimately rule-following information processors, then computers will eventually do everything that humans can do. Human-computer utilogs will then in fact become dialogues because both sides of the interaction will have identical ontological status.

This statement has some serious problems for education, however, even if we only restrict ourselves to the cognitive domain. If we as educators accept the responsibility for the growth of young minds, then we are obligated to ask how such minds do in fact grow. Furthermore, if we find that mental development at all levels requires a dialectical synthesis of personally- and socially-constructed meanings, then we can see that the very ontology of the technological framework (i.e., the world is made up of specifiable and controllable processes) is inadequate for the whole domain of intentions and interpersonal meanings (Greene, 1983). Machine processes, in this case, will never replace interpersonal interactions!

Finally, if we find that human skills and knowledge are ultimately based on tacit beliefs and judgments which cannot be analyzed into components, then computational processes (which by their nature reduce similarity judgments to computable comparisons of component identities) will never match human processes (Dreyfus, 1979).

c. education will be viewed as a form of training and be subject to explicit, extensional logic:

Having an expert author design the goals, rules, and actual messages for a human-computer interaction means that the logic of prediction and control (i.e., the technological framework) is applied to developing pre-planned performance outcomes. The whole educational enterprise is then reduced to a means-ends rationality because the ends are specified first and then the most efficient means are employed to guarantee a quality product (Nunan, 1983; Apple, 1982). The resulting mechanization of interaction is sometimes transparent when computers carry out the actual interaction because of the sophistication, speed, and variety of media involved (Weizenbaum, 1976). But we should never confuse sophisticated technique with sophisticated instruction (Amarel, 1983). Technique does not have a tacit dimension whereas all human knowledge and learning does (Polyani, 1966). Technique is solely subject to extensional logic whereas knowledge is subject to both intentional and extensional logic.

We can see the implications of this view for education most clearly when we examine the nature of experiential learning within the technological and non-technological frameworks (Kolb, 1984). I will do this by comparing the following notions: the nature of experience, events, and activities; the concept of individual; the methods of knowledge; and, the types of thinking involved. My discussion will necessarily be brief.

When restricted to a computational environment, "experiences" take on the form of puzzles <u>of the same type</u> (i.e., computable, quantifiable, procedural). Events are non-historical because they are reversible (i.e., declarative and procedural) and activities are restricted to a non-dialectial "artificial reality" (Kreuger, 1983). Furthermore, an individual is only trivially unique (i.e., the variables of a student model in the computer are generic, only the <u>values</u> of the variables are unique). Finally, an individual only needs the ability to decode abstract symbols because the "text" and "context" are pre-determined by an external agent, and, because knowledge is expressed in an explicit, abstract form. Critical and dialectical thinking are not needed because they make too many things problematical and non-controllable.

In a natural environment, on the other hand, "experiences" are made up of indefinite <u>types</u>. Events are ambiguous, historical, and irreversible (Whitehead, 1929). Activity involves a confrontation between persons and events, and, meanings are personally and interpersonlly constructed. Experiences and actions are dialectical and historical. Furthermore, natural experiences entail an accomodation to, and assimilation of, an indefinite variety of uniqueness in persons, ideas, and events (Piaget, 1970). These in turn, become the experiential basis for further critical and dialectical thinking. Finally, individuals need interpretive as well as decoding skills because they are forced to construct as well as deconstruct the meanings-in-use of others (Greene, 1978). Interpersonal dialogue plays a central role here because knowledge is dialectical, historical, and subject to transformation.

This brief discussion of experiential learning points out the tremendous restrictions that the technological framework places on the variety of educational experiences. Computer-based tutorials therefore seem to rule out everything that is of value to the individual in the natural and social worlds.

Summary of Tutorial Courseware Programs

I began my discussion about tutorial courseware programs with the question of whether human-computer "dialogues" go beyond drill-and-practice approaches. The answer is both yes and no: such tutorials do go beyond because they are a more sophisticated form of interaction but they also stay well within the bounds of the behavioral and technological framework. That is: behavioral outcomes are still pre-specified by expert agents outside of the actual interaction, "quality control" procedures are still used to guarantee that the learner will reach the intended outcomes, and, learners still only have a form of pseudo-control (i.e., rate, route, and timing). Furthermore, although the actual interaction is less rigid than in drill-and-practice courseware, the interaction is still constrained by a computable algorithm, is still focused on maximizing educational performance-gains, and still treats the learner as a means towards someone else's end. Computer-based tutorial interactions therefore provide an artificial "other" that pre-empts personal intellectual agency and ultimately inner-directed learning. Finally, computer-based tutorials are biased against experiential learning (outside of the technological framework), quantum-leaps in learning, and reflective thinking. Their value in education is therefore very limited.

A question now arises about the case where computers are used as "intellectual tools." Does this use of computers go beyond the limitations discussed so far? On first reflection, personal intellectual agency seems to be a natural concomitant of the "tool" use of computers but this conclusion requires a more careful analysis.

3. Computers as Intellectual Tools

What is the intellectual dimension, if any, of computers (Luehrmann, 1980)? To answer this question, I will build on my earlier discussion and then examine the nature of the computer as an intellectual, problem-solving tool. I will not deal with the computer as a personal productivity tool (e.g., word processor) but rather with the computer as an "object to think with" (Papert, 1980). This brings me into the realm of computer languages and simulations.

So far, I have described how drill-and-practice and tutorial courseware programs introduce a means-ends rationality into the learning process. Knowledge-acquisition and skill-building (the terms themselves are revealing) become subject to efficiency and performance criteria, and, learning becomes a systematically-designed and rationally-managed process. Furthermore, knowledge and skills become commodified because they are conceptualized in utilitarian terms and because the design and conception of instruction is separated from the execution of instruction. This commodification, in turn, permits a fine-grained control of the learner's, the teacher's, and even the system's performance. The computer formalizes this whole process and makes it capital intensive.

Does the situation described above apply to the case where the learner programs his or her own solutions to problems (Critchfield, 1979)? Surely here, we will not see the means-ends rationality of an external agent conceptualizing, designing, and managing the learning process. After all, the learner is now in control of the whole process!

The general question therefore becomes whether the student who controls the computer can go beyond the technological framework of the computer. My answer will proceed as follows: tools tend to insist that they be used in certain ways and intellectual tools tend to define the user's mental landscape (Bruner, 1975; Greene, 1978). Computational intellectual tools (i.e., computer programming and simulations) therefore bias our ways of knowing and dealing with the world towards extensional knowledge (i.e., the quantitative and procedural kinds of knowledge) and hide other kinds of knowledge. Intentional knowledge, of course, will not go away. It will only be delegitimized by computers.

<u>The Computer as an Intellectual Problem-Solving Tool</u>

Before answering the general question, I would like to discuss how a computer might bias our ways of knowing. A computer is basically a box that manipulates symbols (and information) according to a plan. When someone else writes the plan (i.e., the program), then we are forced to follow their set of procedures. When we write the plan, then we are forced to use the computer's language. In both cases, we are confronted with a question about the nature of the plans and the types of symbols which the computer can manipulate. We therefore need to explore how programming a particular problem in a computer language helps us learn and think about a problem (Taylor, 1980).

Computers, as I mentioned, are boxes that manipulate symbols according to a plan. The symbols are actually only energy states in an electronic machine which are transformed according to formal, algorithmic rules. Hence, a <u>computer does not add</u> "1+1" to get "2". Rather, the computer initiates an electronic process where two energy states, which we identify as representing the numeric unit "1", are transformed according to an internal structural process (hardwired or software, it matters not) into another energy state which we identify as representing the numeric unit "2". I am belaboring this point because of its generality. If a computer manipulates two high-level representational constructs such as "ALL MEN ARE MORTAL" plus "SOCRATES IS A MAN" and ends up with "THEREFORE SOCRATES IS A MORTAL", then <u>nothing has been added by the computer beyond my first example</u>. The computer only manipulates semantically-empty energy states (which we call input or data or symbols) according to syntactical rules - no matter how high-level those rules. It is <u>we</u> who actively construct and ascribe meaning to these semantically-empty mechanisms. A computer language is therefore <u>not</u> a language in the traditional sense of the term (i.e., expressive, intentional, and connotative as well as denotative and based on qualitative knowing, etc.) but rather a set of syntactical notations to control computer operations (Wirth, 1976; Iverson, 1980). Hence, <u>a computer's expressive potential only extends over the syntactical dimension of its formal operations</u>. Of course, for those who equate cognition with computation, the expressive potential of computer languages extends into the semantic domain because "all relevant semantic distinctions [are] mirrored by syntactic distinctions" (Pylyshyn, 1980). Semantics, in the latter case, become a set of rule-governed, cognitive operations that act on symbolic representations.

What are the epistemological implications of using computer languages to represent events in our world? It is clear that if humans are going to use computers as intellectual tools, then they must work within the epistemological limitations of these tools (Mowshowkowitz, 1976). Since computers can only manipulate explicit data and symbols according to formal, syntactical rules, computers tend to legitimize those types of knowledge which fit into their framework and delegitimize other types of knowledge (Strike, 1974). The latter types of knowledge can only be processed when they are reformulated into computable terms. I described such a transformation earlier with the hunger example. Hence, computers tend to <u>legitimize</u> the following characteristics of knowledge (Broughton, 1984): rule-governed order, objective systematicity, explicit clarity, non-ambiguity, non-redundancy, internal consistency, non-contradiction (i.e., logic of the excluded middle), and, quantitative aspects. They also tend to legitimize deduction and induction as the only acceptable epistemological methods.

By way of contrast, computers tend to <u>delegitimize</u> the following characteristics of knowledge (Streibel, 1983): emergent goals, self-constructed order, organic systematicity, connotation and tacitness, ambiguity, redundancy, dialectical rationality, simultaneity of multiple logics, and, qualitative aspects. And finally, they tend to delegitimize the following epistemological methods: abduction, interpretation, intuition, introspection, and, dialectical synthesis of multiple and contradictory realities.

The more computers are used as intellectual tools, therefore, the more this process of legitimization and delegitimization takes hold. The more we rely on the formal characteristics of knowledge, the less we rely on the tacit and interpretative dimensions of knowledge. It is almost as if the technological framework is not only incompatible with other ways of knowing, but inevitably excludes them from our mental landscapes as well (Ellul, 1980). Of course, the formal and the tacit dimensions of knowledge can never be separated from each other (Polanyi, 1966). The tacit dimension can only become hidden.

The foregoing discussion brings us back to an earlier conclusion: computers force us to act _as if_ we were rule-governed information processors. They also force us to construe thinking as "'cognitive problem-solving' where the 'solutions' are arrived at by formal calculation, computation, and rational analysis" (Broughton, 1984). Even if we are active and constructive and intuitive in our approach to the world, we must still analyze and reduce problems into explicit and procedural terms. Hence, we must restrict our thinking to cognitive operations. The concept of the computer as an intellectual tool is therefore _not_ a neutral formulation because it forces us to objectify ourselves as agents of prediction, calculation, and control (Weizenbaum, 1976). Personal intellectual agency has thereby been limited to the technological framework. This has serious consequences for education.

Computer Programming and Computer Simulations

We can easily see how programming is a paradigm of _thinking_ in the context of the tool use of computers. If the only legitimate knowledge entails objective facts, explicit representations of facts as data, and formal operations on these representations, then programming is the ideal way to process such knowledge. The same can be said of programming as a paradigm for learning. If the only way to think about things is through analysis and procedural debugging, then programming is also the ideal way to learn how deal with the world. After all, we are not just learning to act as if we were computers, we are developing operational and representational cognitive structures to deal with any aspect of the world. Gone are aesthetic, metaphoric, artistic, affective, interpretive, and moral structures for dealing with the world!

We can therefore understand how many of the chief advocates of the tool use of computers see computer literacy as the ability to "do computing" (Luehrmann, 1981) and see computer programming as the best way to shape a child's cognitive development (Papert, 1980). However, in this scheme of things, we can also see that our rational life is thereby reduced to a set of operational, problem-solving skills - to say nothing about our emotional life.

Is there anything positive to be gained from programming aside from the actual technical skills? In several studies, very little positive transfer was found from programming to other domains of cognitive problem-solving (Coburn et al., 1982; Pea & Kurland, 1983). However, this conclusion is only tentative because the field is still too new. We must therefore fall back on an analysis of the nature of programming in order to see what is possible with this approach.

Computing, as Arthur Luehrmann, one of the chief advocates of programming, argues (Luehrmann, 1981):

> belongs as a regular school subject for the same reason as reading, writing, and mathematics. Each gives the student a basic intellectual tool with wide areas of application. Each gives the student a distinctive means of thinking about and representing a problem, of writing his or her thoughts down, of studying and criticizing the thoughts of others, whether they are embodied in a paragraph of English, a set of mathematical equations, or a computer program. Students need practice and instruction in all these basic modes of expressing and communicating ideas.

This certainly is an admirable statement because it integrates computing (i.e., algorithmic, procedural thinking) into the other "basics" of education (i.e., reading, writing, and arithematic) (Kroener, 1981; Ershov, 1981). Luehrmann's argument also casts programming as an aid to understanding.

Given the arguments described above, how could anyone possibly object to programming as a subject matter in our schools? The answer is very simple and applies to the other "distinctive means of thinking and representing a problem": whenever technique is emphasized over grappling with content, then the innermost principles of that content are lost (Ershov, 1972). Although this is also true for reading and writing, this is especially true for computer programming because the computer is an instrument of technique par excellence. The computer can only manipulate content-free symbols according to formal procedures. Hence, although computer programming may force one to structure information in precise and systematic ways and carry out logical operations on abstract representations of that information, it tells us nothing about what information should be treated in this way. It also tells us nothing about the nature of the real world. A simple computer-simulation example should make this clear. The same argument applies to programming the simulation.

Oregon Trail is a popular computer simulation that records the problems which pioneers had in crossing the American frontier (Grady, 1983). It provides a simplified environment for elementary-school children where they can make "decisions" and watch the consequences of their "actions." Hence, forgetting to "buy" enough bullets inevitably leads to program termination. A student can "win" if he or she keeps a careful record of the "purchases" and analyzes the relationships between events, supplies, and milage. What we have here, however, is a quantitative, artificial reality with no hint of the lived reality (Grady, 1983). The simulation, in fact, represents the abstract world of algorithmic logic rather than the lived-experience of historical logic. Hence, whether one is simply using the finished simulation to learn about history or whether one is programming the simulation, historical logic is incapable of being represented. It would be more justifiable to say that winning here (i.e., solving the problem) is more the result of looking for patterns among the numbers than developing a sense of history (Grady, 1983). The simulation is therefore a well-disguised numbers game.

One might object to the foregoing discussion on a number of grounds:

a. the algorithmic logic of simulations does in fact parallel a similar logic in <u>some</u> content areas (such as mathematics) so that computer simulations have a place in education,

b. all learning proceeds from the known to the unknown (and from the simple to the complex) so that simulations are a stepping stone to life, and,

c. persons can learn to become autonomous inquirers within the limitations of a safe and simple artificial reality - a skill that they can later use in real life.

Each of these objections has an intuitive appeal and therefore warrents our attention. Each of these statements, however, can also be interpreted in several ways so that they deserve a careful analysis.

The first objection is easy to handle. It is certainly the case that many real-world activities contain the same logical and procedural structure that is found in the realm of computation. Hence, learning to subtract can be modeled in a computer program because procedural rules are all there is to the <u>process</u> of subtraction (Dreyfus & Dreyfus, 1984). But what does this tell one about the <u>reasons</u> for these procedures?

Brown & Burton have developed an "intelligent" computer tutor that recognizes over ninety ways to make a mistake during subtraction (Brown & Burton, 1979). Each of these procedural bugs models one way in which a person can go wrong in the process of subtraction. This is certainly a very sophisticated approach and may be very useful in some cases. However, it only elaborates the procedures surrounding subtraction. A logical positivist would say: fine, that's all there is to subtraction. A trainer might also say: fine, this will help establish the automaticity of the subtraction skill more efficiently. But an educator would say: wait a minute - subtraction is not an isolated, de-contextualized skill that leads to nothing but itself. At a minimum, it should lead to competence in <u>using</u> subtraction with real-world problems. At a maximum, it should lead to mathematical understanding. In both cases, it should be connected to experiences that ultimately generate personal expertise. And expertise <u>cannot</u> be reduced to procedures because it involves judgments as well as calculations. As Dreyfus and Dreyfus (1984) conclude:

> at the higher stages of skill acquisition, even if there are rules underlying expertise, the rules that the expert has access to are not the rules that generate his expertise ... [Hence], trying to find rules or procedures in a domain often stands in the way of learning <u>even at the earliest stages</u> [my emphasis].

Developing procedure-following skills, therefore, does <u>not</u> facilitate broader learning. I have used subtraction as an example in this discussion because it involves a procedural skill. My argument applies even more for non-procedural kinds of expertise and understanding (e.g., historical expertise and understanding).

The second objection is more difficult to handle: all learning proceeds from the known to the unknown and from the simple to the complex. But we have to be very careful how we define simple so that we do not prejudge the nature of the complex. This problem is a perennial concern in the philosophy of science: should we base our scientific concepts on our intuitions and lived experiences, or should we base them on counter-intuitive conceptual constructions that happen to fit empirical facts (Kuhn, 1962). This problem emerges in education in a number of forms. For example, in science education, should we teach young children to be Aristotelians before Newtonians - let alone, before Einsteinians (DiSessa, 1982)?

In the context of this discussion, the simplicity question becomes: is a simple, context-free, quantitative, and procedural simulation ever an adequate preparation for a complex, contextual, qualitative, and non-procedural lived-experience (Megarry, 1983)? If we wanted to prepare children to understand and deal with the real world, shouldn't we develop simple learning situations of the same kind as those they will later encounter in a more complex form. Isn't problem-solving, in fact, domain-specific no matter how high-level the activity (Newell & Simon, 1972; Lester, 1980; Pea & Kurland, 1983)? The "ivory tower" aspect of schooling might be just the right protection against the harsh realities of life, but this does not mean that schools should become "artificial realities." Using the computer to develop problem-solving skills, however, sets up just such a dichotomy between "simple artificial reality" and "complex natural reality." Notice that the artificial-natural dimension of the above dichotomy is usually hidden in the debate on the matter (Noble, 1984). Learning to program the computer may therefore not be the best way to prepare children for real life.

The final objection is the most difficult to answer: can persons develop analytical and inquiry skills within the limitations of a computational environment that can then be used in real life? After all, analytical and inquiry skills are very general and more like "frames of mind" than simple procedures (Streibel, 1985). The question can be reformulated, however, to reveal what has been hidden: are the analytical and inquiry skills which are developed within a non-contextual, non-dialectical, and judgement-free computational environment useful within a lived environment that requires tolerance for ambiguity, interpersonal construction of new meanings, dialectical thinking, the acceptability of incomplete solutions, and judgement-based actions? When the question is reformulated in this way, a positive answer becomes doubtful. The reason for this is twofold: the computer embodies a technological framework that crowds out other forms of conceptualizing and understanding problems, and, thinking is only ever thinking about something (i.e., problem-solving is domain-specific). Hence, mature analysis and inquiry can therefore only be the result of a history of dealing with similar kinds of things. Flight simulators work so well for this reason - both the simulation and the real-world event are controlled by the same kinds of procedures. Furthermore, the flight simulator is both simpler as well as true to the nature of the real world situation.

A final answer to the third objection remains to be seen. It does seem, however, that the computer restricts our rational life to utilitarian, problem-solving skills. Saying that such skills are under our control does not help very much because these skills have delegitimized other ways of knowing. Saying that such skills display "intelligence" does not help either because intelligence itself has been redefined in a restricted manner. As Broughton laments (Broughton, 1984):

> one can measure the educational impact of computers, and particularly of learning to program them, in terms of what is lost in the process. To the curriculum is lost the arts and the humanities. To pedagogy is lost the hermaneutic art and language that allows us to ask about the meaning of things and of life, to interpret them in their many and various cultural horizons. To both is lost the self and the autonomous capacity to examine critically what we interpret.

Hence, although problem-solving with a computer appears more desirable and high-level than computerized drill-and-practice, programming still limits us to the technological framework. Using a computer as an intellectual tool is therefore a more subtle form of behavioral learning technology because the computer and computational "languages" shape the very categories with which we apprehend and think about the world, and, because it is also done with the active consent and partication of the learner. It represents, as one author has called it, the "industrialization of intellectual work" (Ershov, 1972). This is particularly disturbing because programming (as well as drill-and-practice and tutorial courseware) is being introduced to children in their most plastic and formative years (Bitter, 1982/83; Cuffaro, 1984).

Summary of the Tool Use of Computers

I began this section with the question of whether the "intellectual tool use" of computers went beyond the limitations of tutorial interactions. The answer again is both yes and no. Computers do help us develop a limited personal intellectual agency by forcing us to structure information in precise, systematic ways and specify logical operations on that information. However, this agency only develops within the computational domain. Hence, we are left with an under-developed intellectual agency within the qualitative, dialectical, and experiential domain of natural and social events. Learning to program is therefore only a good way to learn and think about procedural problems - although even here there are some limitations.

The root of the difficulty seems to reside in the nature of computer languages: the expressive potential of computer languages only extends over the syntactical dimension of computer operations. This contrasts sharply with the expressive potential of natural languages which extend over the aesthetic, metaphoric, artistic, affective, and moral domains. Why can't these various languages co-exist? The answer boils down to this: computer languages are part of a technological framework, which, when applied to a number of problems, delegitimizes other frameworks. We are then left with a very restricted mental landscape.

General Summary

I have examined the three major approaches to the use of computers in education and found serious limitations with each approach. The drill-and-practice approach was shown to embody a deterministic, behavioral technology that turned learning into a systematically-designed and quality-controlled form of work. Although drill-and-practice courseware programs were only intended to supplement instruction, they in fact introduced a technological framework in the classroom culture that mitigated against non-behavioral educational goals. Computerized tutorial programs were shown to extend the behavioral and technological approach to learning even further. That is, in tutorial courseware programs, interactions were still shaped by an external agent's intentions in order to maximize the learner's performance-gains and still constrained by computable algorithms. Furthermore, the human learner was still treated as a means towards someone else's ends and only given a form of pseudo-control in the interaction. Most seriously, computerized tutorial interactions pre-empted personal intellectual agency and ultimately inner-directed learning. Finally, the use of computer programming and simulations in education was shown to limit the learner's mental landscape to objective, quatitative, and procedural "intellectual tools." This left the learner with an under-developed intellectual agency within the qualitative, dialectical, and experiential domains of natural and social events.

Each of the approaches described above may have some short-term gain associated with them, but taken together, they represent a shift towards technologizing education. Drill-and-practice courseware programs alter the nature of sub-skill acquisition, tutorial courseware programs restrict the full range of personal intellectual agency, and computer programming and simulations delegitimize non-technological ways of learning and thinking about problems. Taken together, is this worth the price?

END NOTES

[1] Some scholars have suggested that the prevalence of drill-and-practice in our schools is not incidental. The hidden curriculum of drill-and-practice courseware, they claim, prepares a certain segment of the student population for similar types of computer-controlled jobs in society (Apple, 1979; Olds, 1982). However, such an argument involves socio-political evidence and is beyond the scope of this paper.

[2] It should be pointed out that viewing humans as information processors is a very useful assumption for research in educational psychology. Cognition can then be viewed as computation and a theory of cognitive processes can be developed (Pylyshyn, 1980). However, the goal of educational practice is not scientific theory-construction.

BIBLIOGRAPHY

Alderman, D.L. (1978). Evaluation of the TICCIT Computer-Assisted Instructional System in the Community College. Princeton, N.J.: Educational Testing Service.

Amarel, M. (1983). The classroom: An instructional setting for teachers, students, and the computer. In A.C. Wilkinson (Ed.), Classroom Computers and Cognitive Science. New York: Academic Press.

Apple, M.W. (1975). The adequacy of systems management procedures in education. In R.H. Smith (Ed.), Regaining Educational Leadership. New York: John Wiley & Sons.

Apple, M.W. (1979). Ideology and Curriculum. London: Routeledge & Kegan Paul.

Apple, M.W. (1982). Education and Power. London: Routeledge & Kegan Paul.

Barr, R., & Dreeben, R. (1978). Instruction in classrooms. In L. Shulman (Ed.), Review of Research in Education (Vol. 5). Ithaca, IL: F.E. Peacock.

Becker, H.J. (1983). School Uses of Microcomputers: Reports from a National Survey. Nos. 1-5. Baltimore, MD: Center for Social Organization of Schools, The Johns Hopkins University.

Benne, K.D. (1975). Technology and community: Conflicting bases of educational authority. In W. Feinberg & H. Rosemont, Jr. (Eds.), Work, Technology, and Education: Dissenting Essays in the Intellectual Foundations of American Education.

Urbana, IL: University of Illinois Press.

Berliner, D. (1982). Viewing the teacher as manager. Education Digest, 47, 20-23.

Bitter, G.S. (1982/83). The road to computer literacy. Electronic Learning. Sep., 60-63; Oct., 34-68; Nov.-Dec., 41-91; Jan, 40-48; Feb., 54-60.

Bloom, B.S. (1971). Mastery learning. In J.H. Block (Ed.), Mastery Learning: Theory and Practice. New York: Holt, Reinhart, & Winston.

Bloom, B.S. (1976). Human Characteristics and School Learning. New York: McGraw-Hill.

Bolvin, J.O. (1982). Classroom organization. In H.M. Mitzel (Ed.), Encyclopedia of Educational Research. 5th Edition, Vol. 1. New York: MacMillan. pp. 265-274.

Bork, A. (1980a). Interactive learning. In R. Taylor (Ed.), The Computer in the School: Tutor, Tool, and Tutee. New York: Teachers College Press. pp. 53-66.

Bork, A. (1980b). Preparing student-computer dialogs: Advice to teachers. In R. Taylor (Ed.), The Computer in the School: Tutor, Tool, and Tutee. New York: Teachers College Press. pp. 15-52.

Bork, A. & Franklin, S.D. (1979). The role of personal computer systems in education. AEDS Journal, Fall.

Boyd, G.M. (1983a). Education and miseducation by computer. In Megarry et al. (Eds.), World Yearbook of Education, 1982/83: Computers and Education, pp. 50-54.

Boyd, G.M. (1983b). Four ways of providing computer-assisted learning and their probable im, cts. Computers and Education, 6, 305-310.

Broughton, J.M. (1984). Computer literacy as political socialization. Paper presented at the annual meeting of the American Educational Research Association. New Orleans, April 23-27.

Brown, J.S., & Burton, R.R. (1979). Diagnostic model for procedural bugs in basic mathematical skills. Cognitive Science, 2, 155-192.

Bruner, J.S. (1975). Language as an instrument of thought. In A. Davis (Ed.), Problems of Language and Learning. London: Heineman. pp. 61-80.

Bunderson, V. (1981). Courseware. In H.F. O'Neil (Ed.), Computer-Assisted Instruction: A State of the Art Assessment. New York: Academic Press.

Carlson, D. (1982). 'Updating' individualism and the work ethic: Corporate logic in the classroom. Curriculum Inquiry, 12(2), 125-160.

Coburn, P., Kelman, P., Roberts, W., Snyder, T., Watt, D., & Weiner, C. (1982). Practical Guide to Computers in Education. Reading, MA: Addison-Wesley.

Critchfield, M. (1979). Beyond CAI: Computers as personal intellectual tools. Educational Technology, 19(10), 18-25.

Cronbach, L.J., & Snow, R.E. (1977). Aptitudes and Instructional Methods. New York: Irvington.

Cuffaro, H.K. (1984). Microcomputers in education: Why is earlier better? _Teachers College Record_, _85_(4), 559-568.

Dennis, J.R. (1979a). Tutorial instruction on a computer. _Illinois Series on Educational Applications of Computers_. Vol. 6e. Urbana, IL: College of Education, The University of Illinois.

Dennis, J.R. (1979b). The question-episode: Building block of teaching with a computer. _The Illinois Series on Educational Applications of Computers_, Vol. 4e. Urbana, IL: College of Education, The University of Illinois.

DiSessa, A. (1982). Unlearning Aristotelian physics: A study of knowledge-based learning. _Cognitive Science_, _6_(1), 37-75.

Dreyfus, H.H. (1979). _What Computers Can't Do_ (2nd Ed.). New York: Harper & Row.

Dreyfus, H.H. & Dreyfus, S.E. (1984). Putting computers in their proper place: Intuition in the classroom. _Teachers College Record_, _85_(4), 578-601.

Ellul, J. (1980). _The Technological System_. New York: Continuum.

Ershov, A.P. (1972). Aesthetics and the human factor in programming. _Datamation_, _18_(7), 62-67.

Ershov, A.P. (1981). Programming: The second literacy. In R. Lewis, & E.D. Tagg (Eds.), _Computers in Education: Proceedings of the 3rd IFIP World Conference_. Lausanne, Switzerland, July.

Franklin, B. (1974). _The Curriculum Field and the Problem of Social Control, 1918-1938. A Study in Critical Theory_. Unpublished Ph.D. Dissertation. Madison, WI: University of Wisconsin.

Franklin, B. (1982). The social efficiency movement reconsidered:

Curriculum change in Minneapolis, 1917-1950. _Curriculum Inquiry_, _12_, 9-33.

Friere, P. (1973). _Education for Critical Consciousness_. New York: Seabury Press.

Gagne, R.M., Wagner, W., & Rojas (1981). Planning and authoring computer-assisted instruction lessons. _Educational Technology_, _21_, 17-26.

Gardner, H. (1979). Toys with a mind of their own. _Psychology Today_, Nov., 101.

Grady, D. (1983). What every teacher should know about computer simulations. _Learning_, _11_(8), 34-46.

Gray, L. (1983). Teacher's unions and the impact of computer-based technologies. in Megarry _et al_. (Eds.), _World Yearbook of Education, 1982/83: Computers and Education_. pp. 29-41.

Greene, M. (1978). _Landscapes of Learning_. New York: Teachers College Press.

Greene, M. (1983). The literacy that liberates. Tape No. 612-20312. _Association for Supervision and Curriculum Development_. Alexandria, VA.

Hawkins, D. (1974). _The Informed Vision: Essays on Learning and Human Nature_. New York: Agathon Press.

Iverson, K. (1980). Notation as a tool for thought. _Communications of the ACM_, _23_, 444-465.

Jackson, P.W. (1968). _Life in Classrooms_. New York: Holt, Reinhart, and Winston.

Keller, F.S. (1968). Goodbye teacher. _Journal of Applied Behavior_

Analysis, 1(1), 79-89.

Kliebard, H.M. (1971). Bureaucracy and curriculum theory. In V.F. Haubrich (Ed.), Freedom, Bureaucracy, & Schooling. Yearbook of tne Association for Supervision and Curriuclum Development. pp. 74-93.

Kolb, D.A. (1984). Experiential Learning: Experience as the Source of Learning and Development. Englewood Cliffs, NJ: Prentice-Hall.

Kroener, J.D. (Ed.), (1981). The New Liberal Arts: An Exchange of Views. New York: Alfred P. Sloan Foundation.

Krueger, M.W. (1983). Artificial Reality. Reading, MA: Addison-Wesley.

Kuhn, T.S. (1962). The Structure of Scientific Revolutions. Chicago: University of Chicago Press.

Kulik, J.A., Kulik, C-L.C., & Cohen, P.A. (1980). Effectiveness of computer-based college teaching: A meta-analysis of findings. Review of Educational Research, 50, 524-544.

Lester, F.K. (1980). Research on mathematical problem solving. In R.J. Shumway (Ed.), Research in Mathematics Education. Reston, VA: National Council of Teachers of Mathematics.

Luehrmann, A. (1980). Should the computer teach the student or vice versa? In R. Taylor (Ed.), The Computer in the School: Tutor, Tool, Tutse. New York: Teachers College Press. pp. 129-140.

Luehrmann, A. (1981). Computer literacy: What should it be?. Mathematics Teacher. 74(9). 682-686.

Lukes, S. (1973). Individualism. Oxford: Basil Blackwell.

Megarry, J. (1983). Thinking, learning, and educating: The role of the computer. In J. Megarry, E.R.F. Walker, S. Nisbet, & E. Hoyle (Eds.), World Yearbook of Education, 1982/83: Computers and Education. New York: Kogan Page. pp. 15-28.

Messick. S. (Ed.). (1976). Individuality In Learning. San Francisco: Jossey-Bass.

Mowshowitz, A. (1976). The Comquest of Will: Information Processing in Human Affairs. Reading, MA: Addison-Wesley.

Murphey, R.T., & Appel, L.R. (1977). Evaluation of the PLATO IV Computer-Based Educational System in the Community College. Princeton, NJ: Educational Testing Service.

Newell, A., & Simon, H. (1972). Human Problem Solving. Englewood Cliffs, NJ: Prentice-Hall.

Newmann, F., & Oliver, D. (1967). Education and community. Harvard Educational Review, 37(1), 61-106.

Nisbett, R., & Ross, L. (1980). Human Inference: Strategies and Shortcomings of Social Judgement. Englewood Cliffs, NJ: Prentice-Hall.

Noble, D. (1984). Computer literacy and ideology. Teachers College Record, 85(4), 602-614.

Nunan, T. (1983). Countering Educational Design. New York: Nichols Publishing.

Olds, H.F., Jr. (1982). The microcomputer - an environment that teaches: Exploring the hidden curriculum. In Proceedings. The Computer: Extension of the Human Mind. Third Annual Summer

Conference. College of Education. University of Oregon. Eugene, Oregon. pp. 73-85.

O'Neil, H.F. Jr., & Paris, J. (1981). Introduction and overview of computer-based instruction. In H.F. O'Neil, Jr. (Ed.), Computer-Based Instruction: A State of the Art Assessment. New York: Academic Press.

O'Shea, T., & Self, T. (1983). Learning and Teaching with Computers. Englewood Cliffs, NJ: Prentice-Hall.

Papert, S. (1984). Mindstorms: Children, Computers, and Powerful Ideas. New York: Basic Books.

Pea, R.D., & Kurland, D.M. (1983). On the cognitive effects of learning computer programming. Bank Street Technical Report # 18. New York: Bank Street College.

Piaget, J. (1970). Genetic Epistemology. New York: W.W. Norton.

Polanyi, M. (1958). Personal Knowledge: Towards a Post-Critical Philosophy. Chicago; University of Chicago Press.

Polanyi, M. (1966). The Tacit Dimension. Garden City: Doubleday.

Pylyshyn, Z.W. (1980). Computation and cognition: Issues in the foundations of cognitive science. The Behavioral and Brain Sciences, 3, 111-169.

Salisbury, D.F. (1984). Cognitive psychology and its implications for designing drill-and-practice programs for computers. Presented at the annual conference of the American Educational Research Association, New Orleans, April.

Sarason, S.B. (1982). The Culture of the School and the Problem of Change. 2nd Edition. Boston: Allyn & Bacon.

Scheibe, K.E., & Erwin, M. (1979). The computer as alter. Journal of Social Psychology, 108, 103-109.

Scriven, M. (1975). Problems and prospects for individualization. In H. Talmadge (Ed.), Systems of Individualized Instruction. Berkeley, CA: McCutchan. pp. 199-210.

Shneidermann, B. (1980). Software Psychology: Human Factors in Computer and Information Systems. Cambridge, MA: Winthrop.

Shor, I. (1980). Critical Teaching and Everyday Life. Boston: South End Press.

Skinner, B.F. (1968). The Technology of Teaching. Englewood Cliffs, NJ: Prentice-Hall.

Sleeman, D., & Brown, J.S. (1982). Intelligent Tutoring Systems. New York: Academic Press.

Sloan, D. (1984). On raising critical questions about the computer in education. Teachers College Record, 85(4), 539-547.

Sperry, L. (Ed.). (1972). Learning Performance and Individual Differences. Glenview, IL: Scott Foresman.

Stewart, J.T. (1979). Drill-and-practice on a computer. Illinois Series on Educational Applications of Computers. Vol. 7e. Urbana, IL: College of Education, The University of Illinois.

Streibel, M.J. (1983). The educational utility of LOGO. School Science and Mathematics, 83(6), 474-484.

Streibel, M.J. (1985). Beyond computer literacy: Analytical skills, inquiry skills, and personal empowerment. Accepted for publication in the May issue of T.H.E. Journal.

Strike, K.A. (1974). On the expressive potential of behaviorist language. American Educational Research Journal, 11(2), 103-120.

Suppes, P. (1966). The uses of computers in education. Scientific American, 215(3), 206-220.

Talmadge, H. (1975). Systems of Individualized Instruction. Berkeley, CA: McCutchan Publishing.

Taylor, R. (Ed.). (1980). The Computer in the School: Tutor, Tool, Tutee. New York: Teachers College Press.

Turkle, S. (1984). The Second Self: Computers and the Human Spirit. New York: Simon & Shuster.

Weizenbaum, J. (1976). Computer Power and Human Reason. San Francisco: W.H. Freeman.

Whitehead, A.N. (1929). The Aims of Education and Other Essays. New York: Free Press.

Wirth, N. (1976). Algorithms & Data Structures = Programs. Englewood Cliffs, NJ: Prentice-Hall.

Wolcott, H.F. (1977). Teachers Versus Technocrats. Eugene, OR: University of Oregon.