QUICK—A Preliminary Report

LUCE, T. G., S. FLETCHER, R. N. HURST and T. FREDERICK
Departments of Biological and Computer Sciences[1]
Purdue University, West Lafayette, Indiana 47907

Introduction

Virtually all computer assisted instructional systems developed to date have been based on the assumed need for real-time interaction. In fact, Balogh and Purdum (1) believe the total response time should be less than 2 seconds. However, efforts to develop interactive CAI systems to run on existing multi-purpose computing facilities (PICLS (4) and PLANIT (3)) have not been cost effective nor able to service large numbers of users. The main approach to cost effective CAI to date has been the development of large scale, dedicated hardware systems for use by a large number of users (e.g. TICCIT (6) and PLATO (2)). This paper reports on an alternative approach to these problems which should prove to be less costly in terms of capital expense and man-effort.

Silvern and Silvern (5) believe that the term CAI should only be used for "learning situations in which a computer contains a stored instructional program designed to inform, guide, control and test the student until a prescribed level of proficiency is reached." They further believe that CAI must have two way communications between the computer and a human "in which there is a stimulus-response-feedback relationship producing learning." Their definition does not require real-time interaction or that the computer be the only source of information. Early efforts in CAI at Stanford (7) and other places, while interactive, did not use the computer as the primary source of information, but used it to control presentation of audio and visual stimuli. The computer itself was also used to evaluate answers and make decisions as to what the student should do next.

Methodology

QUICK, a "quasi-interactive CAI system" is an attempt to capitalize on this latter feature of many CAI systems while at the same time expanding the response-feedback portion of the cycle enough to allow the more cost-efficient batch processing of answers. Students will be given tasks which may include off-line courseware along with a series of questions based on his past history record. Normally a student would spend anywhere from 15 minutes to several days completing the assigned material. Upon completion of the materials, the questions are answered and submitted to the system as a batch job. Input to QUICK can be in any existing mode such as punched cards, optically read cards, or a remote terminal used to create files for batch processing.

---

[1] This work was supported in part by the Parents' Fund for Instructional Development and Innovative Teaching Grant No. 6674-68-1398.

QUICK currently consists of three main programs and a series of data files. The main program is a driver for content files. It issues tasks, analyzes answers and does the decision making. A second program, which is run periodically, takes update records from the main program and updates the sequential student record file.[2] The third program permits primitive authoring capabilities, again in a batch mode.

Course materials are organized into units called tasks. Each task can exist at one, two or three levels called; low, middle and high. The levels may indicate the relative difficulty of the materials but this is not necessary. Each task level would normally contain textual information, or instructions and a series of one or more questions. Each question in turn may have one or more parts. Associated with each question at each level may be a series of help items of three different types. Type one gives more specific information relative to the question. The second type is for examples and the third is used for references. All help items for a question are shared by all parts of that question with the author specifying which help items are to be displayed when specific parts are missed.

At the present time the author may select one of three response modes for each task. The first mode allows the student only one chance to answer his question and no feedback is given about the answers he misses. This would normally be an exam mode. The second mode allows the student to retry questions which he has missed but does not give him any help items. The last mode allows the student to retry questions or parts of questions and does give him help items if any have been specified.

In association with this, the author can specify the maximum number of retries along with an expected number of retries. By manipulation of variables, various effects can be achieved. For example, by selecting answer mode three with maximum tries equal to one, it is possible to give an exam with no retries while still giving the student feedback on the items he has missed.

Multiple choice answers, or other answers of one character (letter or number) are the only type of answer currently allowed. For each question, the author may currently select one of three different answer processors. The first processor grades each part of each question separately. A second processor is designed to handle sequential questions where the answer to part nine, for example, is dependent on all previous parts. This processor ignores all remaining parts as soon as the answer to any part is found to be in error. The student would receive the help items, if there are any, for the part he missed along with a message telling him that the subsequent answers for the question were ignored. The third processor allows answers to parts to be

[2] Two versions of these programs will eventually exist side by side. One version will be highly transportable with documentation on where to make it locally optimized. The second version will be the optimized version for the specific site. Sequential student record files would be one example of a transportable feature which could be optimized by using random files.

entered in any order. This processor can be used for the type of question where the student is to select from a list of answers all answers which meet some critera. The student may or may not be informed of the number of correct answers and it is not necessary for him to enter them in any given order.

A student may complete a task in one of two ways, by passing the required number of questions or by using up all of his tries. After either condition he moves to the next task. Movement between tasks is based on student requests and author stated prerequisites. Each time the student uses QUICK he may specify the next task and level he wishes to do. Upon completion of the current task, the new task will be given to the student unless he has not completed all prerequisite tasks specified by the author. If prerequisite tasks remain, QUICK will select one for the student. If the student does not specify a task, QUICK will again select a task for him. Student requests for level of instruction are currently honored in all cases. If the student does not specify a level for his next task, one will be selected for him based on the number of tries he used and the expected number of tries specified by the author.

Several important changes are planned for the future. First, QUICK will be modified to allow the issuing of subtasks when the original task is missed. Students are presently asked to retry the same questions. Along with subtasks, it will be possible to have QUICK "construct" questions from lists of parts specified by the author. In the near future the student will be able to request help without supplying an answer. The help items which will be shown will be controlled by author specification. Multiple choice answer processing will also be modified to allow feedback based on specific wrong answers rather than just on wrong answers in general. Later modifications will allow free field input, most likely with keyword and string processors. Still later modifications will include a complex decision making package to control movement between tasks.

Current authoring allows only off line task creation although an interactive preprocessor for authoring is being tested. Future modifications will permit substitution, deletion and editing of tasks. Student record display and manipulation is also in the planning stage.

Every effort is being made to make QUICK very transportable. All programs are written in FORTRAN IV with attention given to differences between IBM and CDC input/output formation specifications. Word size, byte size and character size are all set by a data statement, and may be easily changed for specific machines. Only sequential files are currently used but optimization will probably require the use of random access files (see note on page 439). It has been necessary to use a set of binary logic routines for manipulating information which may not present on all systems. It is likely that these routines will have to be coded in assembly language at each site.

At least three distinct uses can now be envisioned for QUICK. First, QUICK can be used for drill and practice in much the same way as the Stanford drill and practice material. This should prove useful for formative evaluation and practice in almost any course. Secondly, QUICK can function as a self-contained instructional system. It is capable of providing instructional materials, giving feedback to wrong answers and branching to new material as the student requires. Finally, QUICK can incorporate existing or newly developed courseware with its own features to make a well rounded CAI system. This system would include computer control and decision making, human-computer interaction and learning stimuli from computer output to any other audio visual media which are appropriate.

Conclusion

We believe that QUICK will offer a reasonable, cost-effective alternative to some of the traditional uses for CAI. In addition, its transportability and batch processing mode should allow its implementation on many systems which cannot, for whatever reason, support traditional CAI.

Literature Cited

1. BALOGH, RICHARD L. and DON L. PURDUM. 1968. *Computer Assisted Instruction—Feasibility Study.* NSA CR-917.

2. BITZER, D. and D. SHAPERDAS. 1972. "The Design of an Economically Based Educational System." *CERL Report X5.* University of Illinois.

3. FREDERICK, T. J. 1973. "Test Site Evaluation of ICU/PLANIT," Proceedings of *ACM 73.*

4. OLDEHOEF, A. E. and S. D. CONTE. 1971. "Experiments with an Automated Instructional System for Numerical Methods." *Comm. of ACM,* Vol. 14, No. 10.

5. SILVERN, GLORIA M. and LEONARD C. SILERN. 1968. "Computer-Assisted Instruction: Specifications of Attributes for CAI Programs and Programmers" from *Proceedings of 21st National ACM Conference* in *Readings and Outline for "Computer-Assisted Instruction" Seminar.* AM Professional Development Committee.

6. STETLEN, K. J. 1972. "Toward a Market Success for CAI-An Overview of the TICCIT Program." M. 72-73 Report. Mitre Corp.

7. SUPPES, PATRICK and MONA MORNINGSTAR. 1970. "Four Programs in Computer-Assisted Instruction," in *Computer-Assisted Instruction Testing and Guidance,* edit. Wayne H. Holtzman, Harper & Row Publishers, New York.