COMPUTER TECHNOLOGY

Computerizing the activities of subjects for psychological research at a large university¹

B. JAMES STARR,² STATE UNIVERSITY OF NEW YORK AT BUFFALO, Buffalo, New York 14214

Many large schools require students in introductory psychology courses to participate as Ss in a number of experiments. S-POOL, the program described here, computerizes the bookkeeping for such endeavors. The program is described in general terms and system-specific features are discussed. S-POOL may serve as a model for programs that might be developed for other subject-pool systems. The description and flowchart presented here may also aid in program development.

The State University of New York at Buffalo (SUNY/B) requires that students in introductory psychology courses participate in a number of psychological experiments. Keeping track of such research participation by ordinary means can be very inefficient. Thus, clerical workers devote large blocks of time to keeping records regarding the students' fulfillment of research obligations, and administrative errors often occur in the process. In addition, important information is often lacking, e.g., the number of man-hours remaining in the S pool at any particular time. Indeed, important research can come to an abrupt termination due to sudden lack of Ss. It is also difficult to set up research priorities since Ss are often assigned to, or sign up for, an experiment on a first-come first-served basis. Such an informal method may be sufficient to handle small scale research programs, but the method becomes progressively more inadequate as an institution grows. In order to improve the system used at SUNY/B (i.e., eliminating the deficiencies of the then-current method and adding new conveniences), it was decided that the entire bookkeeping process should be computerized. S-POOL (Starr, 1969), the program to be described here, is a direct outgrowth of that decision.

DESCRIPTION OF THE PROGRAM

S-POOL was written for the particular subject-pool system in operation at SUNY/B. Since the program was designed for a specific system, it can best serve as a model for other systems and should probably not be adopted as a general-purpose subject-pool program. Generally, programs as large and complex as S-POOL are not easily modified by programmers who are unfamiliar with the

specific program to be modified. In addition, the program presented here was written in sections and thus contains a minor inefficiency in the composite program. This will be discussed below.

The program, written in Fortran IV, credits students with participation (or failure to participate after having signed up) in a particular experiment. All experiments are number-coded, and S-POOL stores the student number, 12

Fig. 1. A simplified flowchart of program, S-POOL.

letters of his name, the number of credits earned, and the reference numbers for up to nine experiments for each S. The program also generates information concerning the current status of the subject pool, including: total number of Ss, the potential number of subject-hours, the number of subject-hours already used, and the number of subject-hours remaining. The constant updating function of the program is served by two auxiliary tapes; one contains the old record, and the other stores the new, updated pool.

Figure 1 contains a simplified flowchart of the basic operations of the S-POOL program. The three input phases (options) of the program are shown. Phase 1 reads in the course enrollment. Phase 2 assigns experimental credits. Phase 3 is used to delete Ss who have dropped the course and to add new students (e.g., late registrants) to the subject pool. The data cards vary for each input phase.

Three output options are also included in the program. All of these provide the information regarding the status of the subject pool. One of the output options prints all individual records by section. Another option prints only the records of those Ss who are delinquent in fulfilling their subject-pool requirements. The third option suppresses the printing of individual records.

S-POOL stores records for up to 2,300 students in five sections. Four of the sections may contain as many as 475 students; the remaining section may contain up to 400 students. The information regarding the current status of the subject pool is printed out on a "per section" basis. Multiple (up to nine) copies of the output are easily obtained and may be used to apprise the instructors and students of the progress made toward completion of the experimental requirements.

The program was originally written for use on an IBM 7044 computer. More recently, it has been used on the CDC 6400 system. An average credit (Phase 2) run on the CDC 6400 takes about 36 sec. These runs are usually large and are run a few times during any semester. Other input runs are generally run only once during a semester. S-POOL includes a number of built-in error messages to help the user diagnose the most common mistakes.

INPUT

Parameters

The program employs three defining parameters. The LAST parameter refers to the number of records (cards) being entered into storage on a particular computer run. The IN and IOUT parameters refer, respectively, to the type

Table 1
The Values of the Important IN and IOUT Parameters and the Action Taken by S-POOL for Each

Value	IN (Input Is. . .)	IOUT (Printed Output, by Section, Is. . .)
1	Class enrollment cards	All stored records
2	Credit cards (punched by the user) which give information concerning experimental participation	Only the individual records of those Ss who are delinquent in fulfilling their research obligations
3	Drop-add cards containing information regarding late registrants and students who have dropped the course	Measures concerning the current status of the subject pool (individual records will not be printed)

of input (data) being entered into storage and to the type of output desired. The numeric value of these parameters defines the action of the program as shown in Table 1. The above parameters (and a fourth that allows for multiple copies) are punched on a single card which precedes the data cards on any particular computer run.

The First Run

Class enrollment cards always constitute the data for the first computer run. Here, the initial tape generation occurs. After the first run, the tape will contain the name and student number of each S and blank storage spaces for credits and experiment reference numbers.

Subsequent Runs

Subsequent runs are of two general types: credit runs and drop-add runs. Generally, drop-add runs are infrequent. The credit run records the S's transaction (i.e., research participation) with the subject pool. On occasion, an S may have more than the nine allotted transactions with S-POOL. The program contains storage space for 500 student numbers and 500 experiment reference numbers to accommodate these individuals. Even in these "overflow" conditions, each S will have credits automatically added to his total.

Drop-add runs utilize the class enrollment cards as data. These cards are modified by punching a "1" or a "2" in one of the columns. (The number in this column defines a parameter that instructs the program with regard to how to proceed in Input Phase 3.) The program seeks and deletes information identifying the student when the parameter is equal to two. It adds information identifying the new student for those cards with "1" as the defining parameter.

OUTPUT

The output options have already been described in Table 1. What remains is the description of the output. Typical output from a credit run requesting all stored data

might include: (1) credit card information that could not be read into the record and an appropriate error message indicating why; (2) records, by section, of all students; (3) records of student numbers and experiment reference numbers of all Ss having more than nine transactions with the program; and (4) the data indicating the current status of the subject pool. The error messages are self-explanatory and serve to make the user aware of the source of error on a particular data card (e.g., an incorrect student number).

SYSTEM-SPECIFIC FEATURES OF S-POOL

Class enrollment cards at SUNY/B are routinely generated by the Office of Admissions and Records. The program is designed to use these cards in Input Phase 1 and (with a minor modification) in Input Phase 3. In the body of the program, each section of introductory psychology is designated in a specific way. This serves to reduce the area of computer core that must be searched to place each student properly. Instead of searching all 2,300 locations, the simple expedient of punching the appropriate section letter reduces the area that must be searched to 475 locations. The program is geared to read BCD on Tape Unit 5 and to write BCD on Tape Unit 6. S-POOL reads the old binary record from the auxiliary tape mounted on Unit 3 and writes the updated version on Tape Unit 4. Thus, with each run, the auxiliary tapes are alternated between Units 3 and 4.

Introductory psychology students at SUNY/B are required to earn three experimental credits. Any S who fails to participate without reason, after having committed himself for an experiment, is penalized. If the "credit" column of a data card in Input Phase 2 is blank, one credit is subtracted from S's total. Also, the program when IOUT = 2 destroys the records of all Ss with three or more credits. This option (intended for use at the end of a semester) is designed to preserve and print only the records of students who are delinquent in their experimental obligations.

SOME GENERAL COMMENTS

As was noted earlier, S-POOL was written (and tested) in sections. Because of this, the search operation was not written as a subroutine. Consequently, there are a number of sections of the program concerned with locating the appropriate student. When the various sections of S-POOL were integrated, it was felt that the addition of such a subroutine would not improve (or reduce the size of) the program enough to warrant the additional programming and debugging time. (The program, including comments, contains 480 cards.)

Ideally, the program should have been written in a language better suited to the accounting function served by S-POOL. A general-purpose language such as PL/I or a business-oriented language such as COBOL would have minimized the energy expenditure in the task set for the programmer. Fortran IV was chosen at

SUNY/B *only* because it is the language with which most of the graduate students in our department were conversant. It had been our hope that any of these students could be placed in charge of S-POOL (i.e., programming and debugging further modifications and running the program). It is, of course, quite a hardship to modify a large, complex program written by another programmer. Furthermore, S-POOL was quickly modified to its present (finalized) form. The author would thus urge that any further efforts involve one of the other languages that is more compatible with the purposes of the program.

S-POOL allocates three storage locations to each student for experiment reference numbers. A "packing" feature allows three experiment reference numbers to be stored and retrieved, in a relatively simple manner, at each location. This feature together with the "overflow" feature are important sections for rendering the

program compatible with other systems. They would, for example, prove of value to schools operating under a system where points are added to the final grade for each experiment in which S has participated.

REFERENCE

STARR, B. J. S-POOL: A program for keeping track of participation in psychological experiments. *Behavioral Science*, 1969, 14, 252 (CPA 328).

NOTES

1. The development of this program was partially supported by National Institute of Dental Research Training Grant 1T1-DE-170. In addition, help and computer time were accorded by the State University of New York at Buffalo Computing Center, which is partially supported by National Institute of Health Grant FR-00126 and by National Science Foundation Grant GP-7318.

2. The author would like to thank Dr. Roy Lachman for helpful suggestions made during the development of the program and Dr. Erwin M. Segal for his critical reading of the manuscript.

Variants of basic black computer interfaces¹

ROBERT S. McLEAN,²
CARNEGIE-MELLON UNIVERSITY,
Pittsburgh, Pennsylvania 15213

An input interface for connecting psychological apparatus to a process-control computer is described. It permits connection of response switches to eliminate contact bounce and to provide interrupts on ongoing transitions of any data bit.

Uttal (1968) has described a number of basic interfaces between control computers and psychological experimentation. This note mentions a variation of the basic input interface that has been implemented at the Computer Controlled Psychology Laboratory (CCPL) at Carnegie-Mellon University; the intent is to make available an alternative set of techniques to be considered by the researcher about to install a control computer. The logic symbols employed by Uttal are used here, and the reader is referred to the original paper for discussion.

RESPONSE SWITCHES

Uttal describes a design for an input interface that allows S's operations of buttons, switches, and similar manipulanda to be sensed by the computer. One of the basic problems inherent in switch closures is that they often produce "contact bounce"—the momentary generation of

rapid on and off indications at the instant that they are being turned on or off. This problem was recognized, and a switch filter was utilized in Uttal's example. The CCPL solution to this was to require that each switch provide both a set and a reset output, and that the input flip-flop be switched accordingly.

Each switch used by the S (or E) is a double-throw type (Fig. 1). The common contact is connected to the logic level that activates direct inputs to flip-flops (reset-set flip-flop). In this way, the flip-flop "follows" the state of the switch. Since the electrical noise produced by most switches consists of rapid making and breaking of the connection due to mechanical contact of the moving parts and not to alteration of the two states of the switch, this provides for a stable switching arrangement. When the switch is thrown "on," the flip-flop assumes the "on" state at the first indication of "on" from the switch, the switch having previously broken the "off" connection early in its mechanical travel from "off" to "on." Successive pulses on the "on" side have no further effect since no "off" pulse has reset the flip-flop in the meantime. Turning the switch "off" results in a similar process: The flip-flop remains "on" until the first indication of the "off" setting from the switch. Further "off" pulses have no further effect since no "on" pulses have occurred in the meantime.

This also has the advantage that it overcomes some of the problems associated with long wires from experimental laboratory to computer. Induced noise and crosstalk among the conductors of the cable carrying signals to the laboratory and computer may cause spurious data. In this input method, one data line is always grounded except during the interval of switching. If crosstalk continues to be a problem, it may be necessary to provide additional pull-up voltage to the input lines. This is usually accomplished by adding a resistor of appropriate value between the flip-flop inputs and the "1" logic voltage, as indicated in Fig. 2.

INPUT INTERRUPTS

The problem of determining the occurrence and timing of events in the experiment via the computer input circuitry is sometimes difficult. A basic decision must be made concerning the conditions under which the computer is to be able to determine that something of interest is happening in the experiment.

Fig. 1. An input buffer bit and a S's response switch.