COMPUTER TECHNOLOGY

CORE: Computer-controlled operant reinforcement

LORRAINE CHAYER-FARRELL and NELSON L. FREEDMAN
Queen's University at Kingston, Kingston, Ontario, Canada

A Pascal-based system and software is described for computer reinforcement of operant responding (CORE). CORE allows a single microcomputer to control up to eight different work stations simultaneously, on one of several schedules. The user need not be familiar with Pascal, although modifications of the programs require a good working knowledge. The system is menu-driven and interacts with the user in a question-and-answer format. The user can select a variety of experimental parameters, for example, session and reinforcement duration, delay, and type of reinforcement schedule. During run-time, CORE provides frequently updated on-screen graphics (Cumulative Records or Interresponse Time Distributions) and counter display. Permanently stored data and information from other counters may also be displayed or printed from the disk. Presently, three programs are implemented for a variety of related reinforcement schedules (e.g., fixed ratio, variable ratio, fixed interval, variable interval, alternation).

Design and implementation of software systems to control operant schedules has progressed from a single system controlling one work station to more general systems with several built-in schedules simultaneously controlling several work stations while allowing user control of experimental parameters (OPN, Emmet-Oglesby, Spencer, & Arnoult, 1982; Spencer & Emmett-Oglesby, 1985; FREDL, Kaplan, 1985). Although systems to control operant responding for brain stimulation have been described previously (Campbell, Evans, & Gallistel, 1985; Phelps & Lewis, 1982), they were often designed for special purposes and do not satisfy general needs. An interactive system was developed to meet general requirements for Computer-control of Operant responding on a variety of Reinforcement schedules (CORE). Although the reinforcement medium chosen is brain stimulation, the system can easily be modified to operate various other reinforcement mechanisms (e.g., food hopper, liquid dispenser, syringe drive).

DESIGN DECISIONS

Four general requirements were considered important in the design of CORE: (1) capability to run several work stations simultaneously, (2) user selection of experimen-

tal parameters, (3) continuous on-screen data display, and (4) selection among several built-in schedules. The software was designed to be user friendly since CORE is intended for experimenters with no programming expertise, but with a basic familiarity with computer operating procedures.

Some of the same basic design considerations were described by Kaplan (1985), who developed a Pascal-based system, FREDL. Rather than implement each novel conditioning environment, as in FREDL, however, we opted to implement a few specific schedules of particular interest. To avoid user error, experimentally chosen parameters are not altered after the schedule is initiated without terminating and restarting. Rather than using fixed-rate interrupts for real-time control of events, as in FREDL, each significant event calls appropriate procedures within the repeating main control loop. This method was chosen over fixed-rate interrupts, because responses occurring randomly in time are latched in the custom interface until acted on by the computer, making it unnecessary to update timers and counters when stations are idle. Instead, stop and start times are stored temporarily in memory and counters are updated as events occur. We decided to use two timing systems. One rough timer interrupts the system clock at the end of each pass of the main control loop for timing session duration, displaying elapsed time on screen, and plotting. The other precision timer on the Tecmar board (see Hardware) is used for timing with 1-msec accuracy significant events occurring within the main control loop.

The programming language chosen to implement CORE is Turbo Pascal (Version 3.0, Borland, Inc., 1985). This decision was made for the same reasons suggested by Kaplan (1985). It is a structured language, which makes

The work was supported by Natural Sciences and Engineering Research Council Grant A6306 to Nelson L. Freedman. We thank R. Eikelboom, D. J. K. Mewhort, and B. Butler, Queen's University, for critically reading the manuscript; A. Ouimet and D. Hunniford in our electronics shop for custom building the interface and technical assistance; H. Kaplan, Addiction Research Foundation, Toronto, for consultation on preliminary design of the system; and G. Finley, University of Alberta, for a prepublication version of the TECIO.PAS.

Address correspondence to Nelson L. Freedman, Psychology Department, Queen's University, Kingston K7L 3N6, Canada.

programs understandable and easier to follow. BASIC programs require many implicit and explicit GOTO statements, a characteristic that makes development and understanding of the program difficult. Pascal can accommodate more complex data structures than can BASIC or FORTRAN. Such data structures are the backbone of CORE. The Turbo compiler, although nonoptimizing, produces native code at a reasonable speed and efficiency. Assembler, a more time- and space-efficient language, proves to be disadvantageous when implementing large programs (with source code of considerably greater magnitude), and, since good assembler programs are hard to write and debug, skilled programmers would be required for any variations or changes to the programs. The Turbo system also has an easy-to-use graphics package (Turbo Graphix Toolbox Version 1.0, Borland, Inc., 1985), which has the necessary functions and procedures for screen display and updating of cumulative records or interresponse time (IRT) distributions. These features, plus the capability of creating and storing compiled versions of the programs, which are generally much faster than programs interpreted with BASIC, were the deciding factors in our choosing Turbo Pascal.

MAIN SOFTWARE FEATURES

The software features provide continuous on-screen data and graphics display with frequent updating; choice between two methods for selecting reinforcement duration—animal-determined (i.e., determined by time on manipulandum) or fixed user-specified time in milliseconds; and simultaneous control of independent reinforcement parameters within a common reinforcement schedule for multiple stations. In addition, the software provides a rerun procedure to eliminate repetitious parameter entry, provision for checking and reentering parameters, early termination or abortion of a schedule without loss of data, automatic recording of data, and easy step-by-step menu format.

HARDWARE

The CORE system hardware components are a Columbia MPC Computer with 640K of memory capacity (or any IBM-compatible personal computer with a minimum of 256K of memory), two diskette drives, a Panasonic Model 1092 printer for graphics and data hard copies, and a high-resolution TTL monochrome monitor (note: graphics work satisfactorily on a composite video-input monochrome monitor). A PC-Mate Lab Tender board (Tecmar, Inc.) is installed in the computer to provide parallel I/O to control a custom-built interface and an additional AMD9513 clock-counter with a 1-MHz time base. The custom interface consists of a control board used to select one of up to eight I/O boards. We use 21 of the 22 bit busses on the 8255 chip: 8 bits of port A, which are configured as output; 8 bits of port B, which are configured as inputs; and 5 bits of port C, which are con-

figured for board selection signals. Port A has a double function. Initially board selection is accomplished by decoding and latching the output from port A. After the board selection signal has been latched, port A is used for output data to the selected board. Port B is used for input data to the computer from the selected board. Five bits of port C are used for handshaking information between the computer and custom interface. Each I/O board consists of a maximum of eight latched input and eight latched output registers. These registers and boards can be user defined in any combination up to the maximum capacity of each board. Optionally, the input registers can be unlatched through program control. The status of all registers is indicated on two banks of eight light-emitting diodes (LEDs), one bank for input and one for output on each board. A 5-V power supply is used to power the TTL logic on the I/O boards. A separate 24-V power supply is required to drive all external electromechanical devices.

There were several reasons why we opted to build a custom interface as opposed to using commercially available ones. The custom interface provides switch debouncing, which eliminates multiple counts for a single microswitch closure. Inputs are latched so responses will not be missed while control passes from one stage to another during run time. And foremost, the interface is optically isolated from the PC-Mate Lab Tender board in the computer and from all external devices. Thus, damage to the computer from electromechanical transients is avoided. The custom board is not recommended for use with other commercially available devices. For the software to be compatible with other I/O devices, the interface procedures would have to be rewritten.

There are presently four sound-deadened work stations, which the computer monitors and controls through the interface in any combination programmed through the software. Each station contains a Plexiglas box with a house-light and interchangeable manipulanda at each end. The manipulanda consist of a lever with a microswitch and a nose poke hole with photo-detector, set in a piece of brushed translucent Plexiglas. Each manipulandum has an associated cue light that can be turned on or off through software control. All stations have a detachable stainless-steel spring-encased electrode for brain stimulation experiments. The electrode is attached to an AirFlight gold contact swivel. Levels of constant-current stimulation are manually set on separate sine-wave and square-wave stimulators for each station (built in our electronics shop, based on a design by Mundl, 1982). Initiation and termination of stimulation through the electrodes is controlled by the software and interface. Through the interface, the computer detects the activation of either manipulandum and directs delivery of reinforcement to the appropriate station on any reinforcement schedule, including alternation between manipulanda. A simple wiring change at the interface allows the operation of other electromechanical reinforcement devices, such as a food hopper, a dipper, or a syringe drive. Reinforcement can also be activated by the experimenter through a shaping box, whose but-

tons are wired in parallel with the manipulanda at each station. A set of procedures and functions (TECIO.A), originally written in C and assembler for addressing the Tecmar board parallel ports and clocks, have been described (Finley, 1985). A more recent unpublished version (TECIO.PAS) has been made available to us (G. Finley, personal communication, October 7, 1986) and incorporated for addressing the Tecmar I/O ports and clock in Turbo Pascal. For the execution of Turbo Graphix software, the ATI Technologies, Inc., Graphics Solution multi-application video adapter is used. It is a Hercules-compatible monochrome graphic adapter card with 690×200 pixel resolution. Full schematics are available upon request from N.L.F.

SOFTWARE

The software monitors the input devices and controls output devices at each station with a minimum of complex instructions, to reduce sampling time and minimize data loss. This is accomplished through a bitmap. The bitmap is implemented with the following declaration statements:

```
Type  Vocab = (HOUSELIGHT,LEFTLIGHT,RIGHTLIGHT,
               STIMULATOR,LEFTMANIP,RIGHTMANIP);

      Name = string [10];
      Range = 0..8;
      Bytel = 0..256;
      Access = array [1..16,1..8] of Bytel;

Var   Bitmap: Text;
      Device: Name;
```

```
Bit,Board,Station: Range;
Map: Access;
Direction: (input, output);
```

Before starting, the program uses a procedure called "Read_In" to read the bitmap into a data table, "Map," which also checks for appropriate direction labels for each device. If the direction is invalid, then an error message is generated. This table is addressed by all software procedures using the interface. Thus the bitmap and interface can be reorganized without recompilation (see Table 1). The purpose of the bitmap is two-fold. First, use of the bitmap obviates any special knowledge of the interface controlling the I/O devices. Second, direct addressing of I/O bytes by the program is avoided through the use of descriptive and meaningful names for the individual devices. An advantage of indirect addressing is that the software is considerably easier to understand by programmers examining it for the first time. Moreover, by using a bitmap, it is not necessary to assign one board to one station; stations can share bits of the same board, saving board space and money and facilitating occasional use of special-purpose boards. Procedures and functions are used to check the status of, or to activate, a particular device [e.g., check(rightmanip, 1), turn(leftlight, 1, on)]. The bitmap is accessed from within these procedures/functions to retrieve the interface board number and the bit associated with that device.

Three different collections of related schedules have been compiled into three different operating programs that the user may load. The operating programs are menu driven. When the programs are entered initially, a main menu is presented from which a number of options are

Table 1
Contents of Bitmap Addressed by CORE Interface Procedures

Device	Station	Bit	Board	I/O Type	Device	Station	Bit	Board	I/O Type
HOUSELIGHT	1	2	0	O	LEFTMANIP	1	0	0	I
HOUSELIGHT	2	5	0	O	LEFTMANIP	2	2	0	I
HOUSELIGHT	3	2	1	O	LEFTMANIP	3	0	1	I
HOUSELIGHT	4	5	1	O	LEFTMANIP	4	2	1	I
HOUSELIGHT	5	8	8	O	LEFTMANIP	5	8	8	I
HOUSELIGHT	6	8	8	O	LEFTMANIP	6	8	8	I
HOUSELIGHT	7	8	8	O	LEFTMANIP	7	8	8	I
HOUSELIGHT	8	8	8	O	LEFTMANIP	8	8	8	I
LEFTLIGHT	1	0	0	O	RIGHTMANIP	1	1	0	I
LEFTLIGHT	2	3	0	O	RIGHTMANIP	2	3	0	I
LEFTLIGHT	3	0	1	O	RIGHTMANIP	3	1	1	I
LEFTLIGHT	4	3	1	O	RIGHTMANIP	4	3	1	I
LEFTLIGHT	5	8	8	O	RIGHTMANIP	5	8	8	I
LEFTLIGHT	6	8	8	O	RIGHTMANIP	6	8	8	I
LEFTLIGHT	7	8	8	O	RIGHTMANIP	7	8	8	I
LEFTLIGHT	8	8	8	O	RIGHTMANIP	8	8	8	I
RIGHTLIGHT	1	1	0	O	STIMULATOR	1	6	0	O
RIGHTLIGHT	2	4	0	O	STIMULATOR	2	7	0	O
RIGHTLIGHT	3	1	1	O	STIMULATOR	3	6	1	O
RIGHTLIGHT	4	4	1	O	STIMULATOR	4	7	1	O
RIGHTLIGHT	5	8	8	O	STIMULATOR	5	8	8	O
RIGHTLIGHT	6	8	8	O	STIMULATOR	6	8	8	O
RIGHTLIGHT	7	8	8	O	STIMULATOR	7	8	8	O
RIGHTLIGHT	8	8	8	O	STIMULATOR	8	8	8	O

Note—Manip = Manipulandum.

```

THE SIGN ON DATE AND TIME IS:   9/10/1986   13:38:35

***** MAIN MENU *****

EXIT TO SYSTEM,                ----->  A:RUN

SHAPING,                        ----->  B:DESCRIPTION
                                ----->  C:RUN

ALTERNATION,"SHUTTLLING"       ----->  D:DESCRIPTION
                                ----->  E:RUN

ALTERNATION + DELAY,           ----->  F:DESCRIPTION
                                ----->  G:RUN

EXTINCTION,                    ----->  H:DESCRIPTION
                                ----->  I:RUN

DISPLAY DATA FILE,            ----->  J:RUN

PRINT DATA FILE,              ----->  K:RUN

*****
Type Menu Choice by Letter..... Then Press Enter

```

Figure 1. Sample screen of main menu display taken from the ALTerNation program.

available (Figure 1). Main-menu options are: (1) shaping, (2) various reinforcement schedules, (3) extinction, (4) display data file, (5) print data file, and (6) exit to system. Optionally, convenient program descriptions can be selected (Figure 1).

Once the schedule is selected, the operator is prompted to enter the schedule parameters: reinforcement duration (animal determined or fixed); animal identification; reinforcement type (fixed or variable ratio, fixed or variable interval); reinforcement delay time; record of current intensity; record of manipulandum type; and session duration. The actual parameters vary somewhat between operating programs. Consequently, for each operating program, the relevant parameters active at each station are displayed on the screen for confirmation (Figure 2). The parameters are entered sequentially: to make corrections, strike the return key to move the cursor sequentially through the parameter list and reenter the new parameter. When the parameters have been chosen, entered properly, and checked, the operator is presented with a choice of graphics display: cumulative records, or interresponse time (IRT) distributions (see graphics section for details).

After the parameters have been correctly entered, the system is initialized. The initialization procedure zeros the counters and input/output boards, stores the start time, turns on the houselights, and draws and labels the graphic windows. The programs then go into "run" mode. Individual schedules are maintained by a main control loop. The latched input from each station is repeatedly checked to detect whether a manipulandum has been operated. When latched responses are detected, the rat may be reinforced, if schedule requirements are satisfied, and the appropriate counters are updated. Table 2 shows the variety of counters that are currently displayed from the four stations. As an alternate strategy, a revised program will

display all eight stations and allow the user to query the status of the counters with a keyboard interrupt. During idle stages, the elapsed-time display and graphics are updated on the screen.

A schedule can be terminated by one of two methods: (1) normal exit when the preselected session duration times out, or (2) user aborts the session at any time without loss of data by striking the "Q" key. When a session is terminated, all data are temporarily stored in memory and can later be permanently stored in a data file. Upon session termination, the screen is cleared and only the graphic windows are displayed. At this time the user is prompted for a hard copy of the graphic windows. This method was chosen, since storage of graphic screens requires too much disk memory. Thus, the completed graphics display may be viewed at the end of the session, but, unless a hard copy is made, no other permanent record is provided in memory. The final stage in the program is the submenu, which again offers the operator a choice of procedures: store data permanently into the data file, display data file, print data file, rerun same schedule (with or without a parameter change), return to main menu, or quit to system.

GRAPHICS

A central feature of CORE is a selection of on-screen graphic displays to monitor experimental data from all reinforcement schedules. The user may choose to display Cumulative Record or an Interresponse Time (IRT) distribution (Figure 3). The graphics system avoids the necessity for expensive cumulative recorders or plotters. Moreover, individual response data need not be stored for later analysis and graphic display. The choice of cumulative records plots points representing cumulated responses for each station approximately every 10 sec

INTERRESPONSE TIME DISTRIBUTIONS

MAXIMUM OF 20 BINS PER STATION,
WITH 350 MSECS PER BIN INCREMENT

CUMULATIVE RESPONSE CURVES

THE RESET VALUE FOR THE GRAPHS WAS 30

Figure 3. Sample on-screen graphic displays available in CORE. Top: Inter-response time histograms for four work stations with selectable bin size. Numbers above the bars indicate reset frequency; numbers beneath indicate the count in each bin. Bottom: Cumulative response records for four work stations. Dots representing the number of responses are printed on the screen every 10 sec. The counters and cumulative record displays reset after 50 responses (user selectable).

100% under all conditions. In general, accuracy was greater than 98% at strobe frequencies up to 6 pps with one, two, or three stations operating. At higher frequencies or with more stations, accuracy was reduced. Accuracy under these conditions was improved somewhat when plotting was suspended. Thus, under worst-case conditions, 4 animals would have to respond simultaneously at frequencies faster than 6 pps to affect system accuracy.

CORE PROGRAMS

Three CORE programs are described below. These programs share a variety of common features, including shaping of the rat, maintenance of the schedule for a chosen duration, and extinction of responding when required. Although the programs share common features, details of each program are outlined below.

VARINT (Variable Ratio and INTerval schedules)

This program uses one manipulandum on the right side of each station. The shaping procedure is designed so that every response on a manipulandum is reinforced and plotted. The two major schedules in this program are: (1) fixed/variable ratio schedule and (2) fixed/variable interval schedule. Under both schedules, the user enters a ratio or interval value. If fixed, the value remains constant throughout the schedule. If variable, the value is the average ratio/interval. From this value, random number tables are established for each station (1,500 random numbers). The values in the tables are randomly chosen between 1 and $(2 \times \text{average ratio or interval}) - 1$. New values are taken from this table, one after the other, as each ratio/interval is satisfied. When all values have been exhausted, the pointer is reset to the beginning. Although precompiled lists, such as these, use more memory, they eliminate calculation or generation of random numbers in real time. When the schedules are in operation, the control program accumulates responses (or time) until the ratio/interval is satisfied. When it has been satisfied, the animal is reinforced, response counters are updated, and the ratio/interval value is reset (if fixed) or chosen from the random table (if variable). Reinforced and nonreinforced responses are combined on the data plots. The extinction procedure records and plots all responses, but they are not reinforced.

ALT(ALTernation)

This program monitors two manipulanda (right, left) and controls right and left cue lights. During the shaping procedure, both lights are on initially. When the animal responds, the corresponding light is turned off for the duration of the reinforcement, and then turned on again. All responses are plotted.

Two major schedules are available in this program: alternation schedule, and alternation with delay (of reinforcement). In both schedules, the cue light on one side

is selected randomly and turned on. If the animal responds on the cued side, it is reinforced; otherwise, the response is counted, but the animal is not reinforced. Response initiation on the cued manipulandum terminates the cue light for the duration of the reinforcement. On termination of reinforcement, the opposite cue light is turned on. Both schedules are basically the same, except that the latter has a user-specified delay (in milliseconds) between the response and reinforcement onset. Responses on the non-cued side are counted and recorded separately, but only alternation responses are plotted.

During the extinction procedure, all responses are recorded, no reinforcement is given, and only alternation responses are plotted. The user also has the option to choose extinction with or without alternating light cues (secondary reinforcers).

ALTMANIP (ALTernation with variable MANIPulandum control)

This program combines some of the features of VARINT and ALT. It was written to develop parameters to equate response rates on the different manipulanda. Once again, the animals alternate between two manipulanda with light cues, but right and left manipulanda are controlled by different schedules. The program offers two varieties of alternation: 1) alternation and ratio: one manipulandum is under normal alternation control, and the other is under a ratio control (fixed/variable); 2) alternation and delay: one manipulandum is under normal alternation control, and the other has a specified delay (in milliseconds) between the response and reinforcement onset. The user is prompted for the side to be controlled by the ratio/delay at the time of entry to the schedule. All responses are recorded, but only reinforced alternation responses are plotted. The shaping and extinction procedures are as described in the ALT program.

In all three programs, reinforcement duration is determined by the user during the parameter-entry stage. It can be either a fixed value, in milliseconds, or animal-determined. If it is animal-determined, brain stimulation reinforcement is delivered continuously until the manipulandum is released.

USER EXPERIENCE AND CONCLUSION

The system has been in use for several months. The implementation of on-screen graphics has proved to be a useful feature, providing the user with immediate feedback on the rate, variability, and temporal characteristics of responses. Slight modifications to data display and to parameter entry have been made to accommodate users, increase ease of use, and correct unanticipated problems. Decisions to advance animals to a more complex schedule (e.g., to increase from a VR4 to a VR5) are made more easily. All the programs have a similar structure and use common procedures and functions. Therefore, new schedules and programs may be developed easily and

rapidly by altering the main control loop, and by making minor adjustments to parameter entry and data display. Free-operant avoidance and reward programs using similar procedures are currently being written. Programmers familiar with the Pascal language can easily modify the programs, although more complex procedures, such as multiple-schedule self-administration, would require large-scale reprogramming in Pascal.

AVAILABILITY

A fully commented listing of the programs (on paper or on disk) may be obtained for a \$10 fee to cover shipping and handling costs. Checks should be made out to Queen's University. The programmer is available for consultation. We cannot guarantee results on different systems and interfaces. All requests should be addressed to Nelson Freedman.

REFERENCES

- CAMPBELL, K. A., EVANS, G., & GALLISTEL, C. R. (1985). A microcomputer-based method for physiologically interpretable measurement of the rewarding efficacy of brain stimulation. *Physiology & Behavior*, **35**, 395-403.
- EMMETT-OGLESBY, M. W., SPENCER, D. G., & ARNOULT, D. E. (1982). A TRS-80 based system for the control of behavioral experiments. *Pharmacology, Biochemistry & Behavior*, **17**, 583-587.
- FINLEY, G. (1985). Laboratory functions for the Tecmar Lab Tender and IBM PC. *Behavior Research Methods, Instruments, & Computers*, **17**, 397-398.
- KAPLAN, H. L. (1985). Design decisions in a Pascal-based operant conditioning system. *Behavior Research Methods, Instruments, & Computers*, **17**, 307-318.
- MUNDL, W. J. (1982). Amplitude of constant-current pulse can be preset. *Behavior Research Methods & Instrumentation*, **14**, 459-460.
- PHELPS, R. W., & LEWIS, M. J. (1982). A multifunctional on-line brain stimulation system. *Behavior Research Methods & Instrumentation*, **14**, 323-328.
- SPENCER, D. G., & EMMETT-OGLESBY, M. W. (1985). Parallel processing strategies in the application of microcomputers to the behavioral laboratory. *Behavior Research Methods, Instruments, & Computers*, **17**, 294-300.

(Manuscript received September 18, 1986;
revision accepted for publication March 19, 1987.)