POPEYE: A PRODUCTION RULE-BASED MODEL OF MULTITASK SUPERVISORY CONTROL (POPCORN)

James T. Townsend, Helena Kadlec, and Barry H. Kantowitz
Purdue University
West Lafayette, Indiana

Abstract

Recent studies of relationships between subjective ratings of mental workload, performance, and human operator and task characteristics have indicated that these relationships are quite complex. In order to study the various relationships and place subjective mental workload within a theoretical framework, we developed a production system model for the performance component of the complex supervisory task called POPCORN. The production system model is represented by a hierarchial structure of goals and subgoals, and the information flow is controlled by a set of condition-action rules. The implementation of this production system, called POPEYE, generates computer simulated data under different task difficulty conditions which are comparable to those of human operators performing the task. This model is the performance aspect of an overall dynamic psychological model which we are developing to examine and quantify relationships between performance and psychological aspects in a complex environment.

Introduction

With increased automation in the working environment, physical demands of tasks have, in many situations, become secondary to mental or psychological demands. Automation has changed the role of the operator from one of direct control to one where the operator primarily monitors and schedules multiple tasks. This has resulted in complex systems which place greater demands on the operator's information processing capabilities. In these situations it is often assumed that performance on tasks is mediated by the allocation of processing resources which are limited (ref. 1). Mental workload is then operationally defined in relation to the overall ability of the human processing system to process information and generate responses as the task demands change (ref. 2).

Human factors and cognitive psychologists have recently begun to investigate potential variables contributing to mental workload using a variety of methods. Since mental processes are not directly observable, they are often inferred from the operator's performance or physiological measures. Alternatively, estimates of mental workload may be obtained directly from the operator's subjective judgments of the workload imposed by the task. Because of its high face validity, the latter approach of obtaining subjective ratings of workload has become widely used in human factors research.

The relationships between the performance measures and subjective ratings of workload, however, are not clear and sometimes the measures do not correlate as task demands change. In addition, many results have been accumulating (see, e.g., ref. 3 for a review), without a coherent theory to bring the observations together. Consequently, a more unified approach, which would embed the various aspects of this research area such as would be provided by a modeling approach, could clarify the relationships between performance and subjective workload measures. Our model of the complex task POPCORN, which will be described in the next section, is an attempt at this approach.

Relationships between the task type and task difficulty on the one hand, and subjective workload ratings and performance measures on the other are complex. Results seem to depend on the task itself, as well as how and when the workload manipulation is accomplished (refs. 4 and 5). Other task characteristics, e.g., task priority and reference task (ref. 6), also play a role. Most important, however, is the result of the latter study which shows that performance and workload ratings do not correlate under all conditions. Finally, while task characteristics certainly affect workload, recent investigations also seem to suggest that operator characteristics may affect not only performance but also workload ratings, at least under certain conditions (refs. 7, 8 and 9).

The considerations that are involved in examining subjective workload, some of which were briefly discussed above, underscore the importance of modeling, since from a practical, as well as a scientific view, it seems extremely important to be able to identify and quantify these various factors contributing to subjective mental workload. That is why we feel that a model, which would represent the performance as well as the psychological aspects of the operator in a dynamic way, could prove very useful in this area of research. With a working model, we could elucidate the relationships between workload (as well as other psychological) and performance measures in a quantitative way as various task characteristics are manipulated. One such possible dynamic model is shown in Fig. 1.

We began by modeling the performance component of the task. In particular, we developed a production system model of POPCORN, utilizing some of the production systems ideas developed by Newell (ref. 10) and later elaborated by John R. Anderson (refs. 11 and 12). Production systems have been useful in modeling various cognitive skills, such as general problem-solving (ref. 12) and a computer text editing task (ref. 13). Our production system will be presented following a brief description of the POPCORN task.

Description of the POPCORN Task

A complex task, called POPCORN, was recently developed at NASA by Sandra Hart for studying psychological variables that may contribute to the experience of workload. This task simulates a relatively complex automated system where the operator is responsible primarily for decision-making and the scheduling of the different components of the task in order to maximize the score in a minimum amount of time.

The POPCORN task is implemented on the IBM PC AT, and the operator interacts with it via a mouse. The complexity of a particular simulation can be manipulated primarily by the number of functions available to the operator, and ranges from level 1 (least complex) to level 5 (most complex). To begin the modeling, we chose level 2 since it has only six of the twelve functions available and thus is easier to model, yet it is psychologically interesting since some decision strategies must be employed.

The monitor display, as it appears for a level 2 scenario, is shown in Fig. 2. The larger boxes along the bottom of the display are the task boxes, with the smaller boxes beneath them used to select the different tasks. There are five task boxes, each of which will contain a task of a different type, and one penalty box which has no lid. The boxes along the right hand side are the functions used to operate on the tasks. At the second level of complexity, the functions OPEN, CLOSE, STUFF, Y->G, R->Y and SEE are available. The OPEN function opens the task box, while the CLOSE function closes it. The STUFF function is used to replace all the individual "kernels" of the task that have popped out back into their task box. The other three functions are used for kernels that have changed their state (i.e., color or visibility) in the warning zone (see below).

The scenario would proceed as follows. At specified times the task boxes are filled with the "tasks"; each task is a group of identical "kernels", the five different tasks being represented by kernels of different symbols, # - + = and *. The kernels can be released from their particular task box by first selecting that task (by moving the mouse to the smaller box underneath the task box and clicking the mouse), followed by clicking the mouse in the OPEN function box. Once the task box is open, the kernels "pop out", one at a time, and float in an upward direction at a predetermined speed specified by the experimenter. Each click of the PERFORM function (lower right hand corner of the display and available at all levels of complexity) renders one kernel of that task done, whereby the kernel disappears from the screen and the score is incremented. Only popped kernels may be performed, and only one at a time.

As the kernel moves up the screen, it may be performed as long as it has not crossed the warning line. Once the kernel crosses the warning line, it can change its state to one of the warning states (which was predetermined by the experimenter). The "normal" state of the kernel is green. In the warning zone, it can change to either yellow, red, or invisible. As the changed kernel moves up through the warning zone, it can still be performed for points if its state is first returned to green by pressing the appropriate sequence of functions. When the kernel is returned to its green state it must first be performed before the next kernel can be operated on. These warning states are one of the ways of penalizing the operator for lagging behind. If the kernel is not performed in time, it moves to the top of the screen where it disappears and goes to the graveyard. An optional penalty for each dead kernel can be imposed by subtracting points from the score for each dead kernel.

If there is another task scheduled to enter into a task box which still has some (or all) kernels in it, the operator is given a 20 second warning by a red flashing bar under that task box. If the kernels in the task box are not done within that 20 second warning, the unperformed kernels are sent to the penalty box. There the kernels lose their identity, and since the penalty box has no lid, they begin to exit as soon as they arrive there. The points for performing these kernels are no longer obtainable; however, performing them does avoid the penalty for dead kernels.

The object of the simulation task is to obtain as many points as possible in the least amount of time. Often, therefore, the scenarios can be performed faster and more efficiently if two or more tasks are worked on simultaneously, by alternating between them. The higher levels include progressively more functions which allow the operator a wider range of options and strategies. These will not be described here since they are not included in the model at the present time. As an operator plays POPCORN, the functions and the times at which those functions are performed are stored in a response file by POPCORN.

In addition to the complexity level and also within each complexity level, the difficulty of each POPCORN scenario can be manipulated by four major variables: 1) the number of kernels in each task, 2) the total number of tasks, 3) the task schedule (i.e., the schedule of the arrival times of the tasks; a massed schedule results when all tasks arrive simultaneously, while different arrival times result in a staggered schedule), and 4) the speed of the kernels' movement. These variables will be used to examine the effects of environmental factors on the performance of POPCORN, and later to study the influences of the psychological variables of the model. We next describe the production system for the performance component of the POPCORN task.

Production System Model of POPCORN Performance

Performance of POPCORN lends itself to a production system approach since it can be readily interpreted as a hierarchy of goals and subgoals. The hierarchial goal structure is presented in Fig. 3 and the corresponding productions controlling the flow of control of the system are given in Table 1.

There are two main branches in the system. The first branch (productions P1 to P13) consists of the strategy selection that an experienced operator may engage in to prepare for playing POPCORN. Prior to the task, the operator is given a brief description of the upcoming task, called the flight plan. The flight plan provides information about the number of tasks to be done, the number of kernels in each task, the arrival schedule (massed or staggered), the speed with which the kernels move, the rewards/penalties for performed/dead kernels, and the state of the kernels in the warning zone. Based on this information and the operator's experience, (s)he can form an initial opinion about the difficulty of the upcoming scenario and decide, perhaps tentatively, on an initial strategy. The second branch (productions P14 to P44) is the production system of the actual performance of the POPCORN task. It should be emphasized that the operator is not bound in any way to use the initial strategy once (s)he starts playing. The playing strategy can be re-evaluated at any time if it is not conforming to the proper execution of the task. A demonstration of the production system follows.

The performance of the POPCORN task begins with the goal to 'play POPCORN'. Since the flight plan is the first thing to appear on the monitor, production P1 applies and the new goal becomes to 'choose an initial strategy'. If the flight plan has not yet been read and processed by the operator, production P3 applies and the goal becomes to 'read the flight plan'. Production P4 is the only one that applies here, and the operator reads the flight plan, stores the levels of the variables pertaining to the scenario (e.g., whether the speed of the kernels is slow, moderate or fast, whether the arrival schedule is massed or staggered, etc.) in working memory (WM), brings into WM the weights of these variables from long-term memory (LTM), and initializes the variable DIFF (difficulty) to zero and VARIABLE to 1. These latter two variables will be used in calculating the perceived difficulty of the scenario on which the strategy will subsequently be based.

The weights of the flight plan variables pertain to the importance of each variable in contributing to the difficulty of the scenario. For example, the speed with which the kernels move may contribute more to determining the difficulty than the number of kernels in each task, and will thus have a greater weight. Our pilot work indicates that the speed variable is the most important variable in determining the perceived difficulty of a scenario. These weights are parameters of an operator which get updated, or tuned, based on the operator's experience. Table 2 shows a possible way of breaking down each variable into its levels, which are the independent variables of our studies by which we manipulate the difficulty or complexity of the environment. An example of the calculation of the perceived difficulty is also presented in Table 2. For illustrative purposes, the parameter values are chosen such that the DIFF variable lies between 0 and 10.

Once the flight plan is read but the strategy has not yet been chosen, production P5 applies and the goal becomes to 'weigh the variables' of the flight plan which are now stored in WM by P4. Production P7 calculates the perceived difficulty (DIFF) of the scenario in a manner analogous to the example shown in Table 2. When all the variables have been calculated into the DIFFiculty score, P8 makes the new goal to 'pick one strategy Si' (i = 1, 2, ..., 5). Here, depending on the result of the DIFF score, one of productions P9 to P13 will apply and a strategy is chosen.

The strategies are labeled S1 through S5. Strategy Si denotes that the operator will work on i tasks simultaneously. Thus, for example, when the perceived difficulty is less than 2 (i.e., a very easy scenario), production P9 will apply and the operator chooses to work on all five tasks simultaneously, strategy S5. As the difficulty increases, fewer tasks can be done simultaneously.

When the strategy is chosen, P6 and P2 return the system to the goal to 'play POPCORN' again. This time the conditions of P14 apply and the new goal becomes to 'work on the tasks'. Initially all the task boxes are closed and the kernels cannot get out. Thus if the i task boxes that the operator wants to work on are not open, P16 applies and the goal becomes 'open all i task boxes'.

Since at the second level of complexity only one task can be attended to at any one time, in this production system, task X will refer to the task the operator is currently attending to. (Note that in productions P19, P25, P26, P27 and P35 task X can also include the penalty box; however, opening or closing the penalty box constitutes an error.) From P2 task X has been tagged as the first task to be opened. But task X has not yet been selected thus P18 applies and the goal becomes to 'select task X', which is accomplished by P19 where the mouse is moved to the smaller box under task X and the mouse is clicked. When task X has been selected P20 applies and the goal becomes to 'open task X', which is accomplished by P21. When task X has been opened, but not all i tasks have yet been opened, P22 makes the next task the current task, which is then selected and opened in the same manner. Upon opening all i task boxes, P17 applies and the new goal becomes to 'work on tasks' again. Now the task boxes are open and kernels are popping out, so P23 applies. Here the operator decides which popped kernels will have to be performed first (if $i \geq 2$). It is assumed that the task with the most popped kernels will always be chosen to be operated on first. Now the new goal becomes to 'perform popped kernels' which is where the majority of the actual playing of POPCORN takes place.

The most straightforward way to play is to select task X, if it is not already selected (P25), perform all the popped kernels of that task (P27), then select a new task with the most popped kernels (P26), perform those (P27), select another task with the most popped kernels (P26), perform those (P27), and so on until all popped kernels are done. However, other conditions may arise, particularly in faster scenarios, where the operator has to switch tasks or the order of performing the popping kernels in order to accommodate new incoming tasks without losing points or to take care of kernels that have gone into the warning zone.

If the kernels of the current task have entered the warning zone and changed to yellow, then one of productions P28, P29, or P32 applies depending on further conditions of the scenario. If there are no kernels popping out of any of the other (open) task boxes (i.e., only the current task is left to do at this point) and the scenario is not too difficult, then the operator can process the warning state and P32 applies to make the new goal to 'process the warning state'. This is the most efficient strategy in this case since a minimal amount of time is lost. Production P34 changes the top kernel in the warning zone from yellow to green, and P33 brings the system back to the goal to 'perform popped kernels' where P27 now applies. The sequence of P32, P34, P33, and P27 must be applied for each kernel in the warning zone, thus it is assumed that the warning state can only be efficiently processed in situations where there is enough time and there are no other demands on the operator. The experienced operator knows from past experience in which situations the warning states can be efficiently processed, and some pilot work has supported this assumption. For the other warning states, red or invisible, productions similar to P34 can simply be included in this part of the production system.

If the scenario is too fast (i.e., the DIFF is greater than some critical value which can be thought of as another operator parameter; here 5 is chosen somewhat arbitrarily for illustration), then P29 applies and the operator stuffs the task. This loses some time but prevents the loss of points if there is not enough time to process the warning state.

If the kernels of the current task have entered the warning zone and kernels are also popping out of other tasks, some of which may also be near or entering the warning zone, then P28 applies and the new goal becomes to 'stuff task X' in order to avoid losing them whereby their performance is postponed until later. In this case, if the scenario is too difficult, the best strategy is to stuff the kernels back into their box and close the box (P31) in order to have sufficient time to perform the other popping kernels. If the scenario is relatively easy, then only stuffing the task (P30) may be sufficient to provide enough time to catch up with the other popping kernels.

Another situation where the straightforward sequence of selecting and performing kernels as they pop out (using productions P25, P26, and P27) may be disrupted arises when the 20 second warning flashes under a closed task box signalling the upcoming arrival of a new task in that box. In such a case, the task (called task Y in P35) has not yet been selected, and if the situation permits the processing of an additional task (e.g., when other open tasks are finished, or their kernels are popping slowly and not approaching or inside the warning zone), as judged by the operator, then production P35 applies and that task is selected and then opened (P36). In this situation the new task is incorporated into the ongoing strategy [Si becomes S(i + 1)].

If the scenario is fast and there are already many popping kernels, the operator may elect to stuff, and possibly close, one of the current tasks (P37). In this way the task box with the flashing warning in essence takes the place of one of the current tasks in the strategy, and the performance of the popping kernels can proceed in a "normal" fashion. However, the experienced operator can judge how much time is required to pop and perform the kernels, and may even be able to finish a started task before switching to the new one.

When all the kernels of the i tasks have been performed, then the best strategy is to close at least some of the finished task boxes if more tasks are expected to arrive in those boxes. If the empty task boxes remain open, then the kernels of the newly arrived task will begin to leave as soon as they arrive. Production P39 will apply in this case, and the new goal becomes to 'close (5-i) task boxes'. (Note that closing (5-i) tasks assumes that the strategy Si remains effective; this assumption seems reasonable for an experienced operator.) Again the task to have the close function performed on it must first be selected, if it is not already selected, (P43 and P19) before it can be closed (P41 and P42).

At this point the operator opens the next i tasks which contain kernels (P16) and the game continues in the same manner as above. If the operator has finished all tasks but more are to arrive later, all there is to do is wait (P38) until the new ones arrive. If all tasks are done, productions P15 and P0 end the game.

POPEYE: Computer Implementation of the Production System

Due to the IBM PC AT system limitations, the computer implementation which we call POPEYE does not perform POPCORN in real time. Rather, it simulates results as if it were playing POPCORN. Each time a 'move' is executed, the scenario, as it appears to POPEYE, is updated. Thus the program keeps track of the running time, as well as the last time that the scenario was updated, and updates the scenario for the time difference. The generated responses are stored in an output file which has the same form as the replay file generated by POPCORN when a human operator is performing the task. Thus the responses generated by POPEYE can be checked by running the POPEYE output file using POPCORN's replay command.

The current version of POPEYE performs the task only under the following task constraints. 1) The schedule of task arrivals must be massed, that is, all five tasks of each set must arrive simultaneously. This was done in order to make the initial programming of POPEYE manageable. 2) The current version can only perform two sets of tasks per scenario, although it will not be a problem to make the program flexible to include any number of sets in the next version. Any warning state can be processed, and there are three different speeds available; 0.3 cm/sec, 0.7 cm/sec, and 1.2 cm/sec.

POPEYE prompts the user for a "difficulty criterion", an integer between 1 and 10. This is an operator parameter corresponding to the criterion value for the DIFF variable in the production system (which was set to 5 in Table 1 for illustration), and is used to determine if a task box should be closed after all kernels in it are done, and also to determine whether a task should be stuffed or kernels in the warning zone processed (for productions P31, and P32). This criterion is used in POPEYE by comparing it to the calculated difficulty (DIFF) of the scenario based on the flight plan variables and weights.

POPEYE also prompts the user for an operator parameter "kernels criterion", an integer between 1 and 4, which is used to determine whether to close a box after it is stuffed. If the number of kernels popped out of another task exceeds this criterion, the current task is stuffed and closed; if not, the task is only stuffed. Finally, the last prompt is for the operator's "mean to move" the mouse. This mean is used to generate an exponentially distributed random number which is added to a constant representing the minimum time between two moves.

In the current version of POPEYE the tasks are performed left to right and consecutively, unless emergency situations arise. Also all popped kernels of a selected task are completed before the next task is selected. In our pilot work, these performance assumptions were fairly well supported.

Game parameters which describe the scenario to be simulated must be provided for POPEYE. These parameters include: 1) the number of task sets to be performed (currently only 2 are allowed); 2) the number of kernels per task (any integer between 1 and 8); 3) the schedule code; and 4) a code for the warning state. These are read from a file by POPEYE. In addition, each operator has his/her own flight plan variable weights, which are stored in a separate file. This file, in a way, represents the long term memory of the operator, and contains the weight for each flight plan variable and the weights for the different levels of each.

We have not yet analyzed the performance of the model statistically, but assessed its performance by viewing the generated results as they were replayed in the actual POPCORN task. The data simulated by POPEYE was virtually indistinguishable from data produced by human operators. Depending on the parameters given, POPEYE can generate data which result in performance that looks either like a well-practiced operator or a beginner.

Future Directions

The next version of POPEYE will aim toward a dynamic interactive model which will include such psychological variables as frustration, motivation, and working memory, as shown in Fig. 1. Throughout the report, some reference was already made to some of these psychological variables, and in fact the current version of POPEYE already contains and uses some of these variables (e.g., working memory), albeit not very formally at this stage of modeling. Thus the extension toward a dynamic psychological model is a very natural consequence of our work so far.

By studying the performance aspects of POPCORN as they change with different psychological manipulations, for example, by increasing the number of frustrating events or errors that the operator experiences, we can examine how these psychological variables contribute not only to the operator's performance but also to his/her experience of the individual aspects thought to underlie workload experience such as time pressure, physical and mental effort, etc. In addition, we can investigate how these individual aspects contribute to an overall experience of workload. In this way, POPEYE can be extremely useful in the investigation of the interactions of these (and possibly other) psychological variables with the performance component of the model and their contribution to the experience of workload.

With the exception of Madni and Lyman (ref. 14), no one to our knowledge has attempted to model mental workload and its relationships with performance and task characteristics. Madni and Lyman's model is an extended Petri net representation by which they attempt to describe and quantify task-imposed workload. However, we are not aware of a computer implementation of their petri net model. Petri nets are similar to production systems in that they are formal models of information flow. Whereas both approaches rely on some matching of conditions to proceed from one state to another, production systems additionally postulate a hierarchial structure of goals which governs the overall behavior. The goal structure seems to be more appropriate to model the goal-directed behavior of human operators.

Thus, the production system approach is a useful and suitable representation of POPCORN performance. It is straightforward, and simply by adding more productions it can be fairly easily expanded to model higher levels of complexity. Also, since an action of an operator at any given time only depends on the current state that he/she finds him/herself in -- that is, the transition from one state to another depends only on the current state and not on any of the previous states -- the production system can be naturally generalized to a state probabilistic model by employing a Markov process approach.

The dynamic model will also be very useful in estimating workload ratings under different environmental conditions. For example, a straightforward estimate of workload may be obtained by simply estimating the absolute number of productions required to complete the task. Alternatively, a more complex and accurate estimate may result from a weighted combination of the productions, where a production with more conditions to be matched or more consequents to be performed may contribute to a greater extent. In summary, we feel that this approach to the modeling of POPCORN and employing the model to predict workload ratings is very useful and holds much promise.

Acknowledgment

This research was supported by Workload and Performance Research Group, NASA-AMES, grant # NAG 2-307, to the first author.

REFERENCES

1. Navon, D., & Gopher, D.: On the economy of the human-processing system. Psychological Review, 86, 1979, pp. 214-255.

2. Gopher, D., & Braune, R. (1984). On the psychophysics of workload: Why bother with subjective measures? Human Factors, 26, 1984, pp. 519-532.

3. Moray, N.: Subjective mental workload. Human Factors, 24, 1982, pp. 25-40.

4. Yeh, Y.-Y., Wickens, C. D., & Hart, S. G.: The effect of varying task difficulty on subjective workload. Proceedings of the Human Factors Society, 29th annual meeting, 1985, pp. 765-769.

5. Thornton, D. C.: An investigation of the "Von Restorff" phenomenon. Proceedings of the Human Factors Society, 29th annual meeting, 1985, pp. 760-764.

6. Gopher, D., Chillag, N., & Arzi, N.: The psychophysics of workload - a second look at the relationship between subjective measures and performance. Proceedings of the Human Factors Society, 29th annual meeting, 1985, pp. 640-644.

7. Damos, D.: The relation between the Type A behavior pattern, pacing, and subjective workload under single- and dual-task conditions. Human Factors, 27, 1985, pp. 675-680.

8. Damos, D., & Bloem, K.: Type A behavior pattern, multiple-task performance, and subjective estimation of workload. Bulletin of the Psychonomic Society, 23, 1985, pp. 53-56.

9. Salvendy, B. (Ed.): Responses of Type A and Type B individuals performing a supervisory control simulation. Proceedings of the Second International Conference on Human-Computer Interaction, pp. 67-74. The Netherlands: Elsevier Science Publishers, 1986.

10. Newell, A.: Production systems: Models of control structures. In W. G. Chase (Ed.), Visual information processing. New York: Academic Press, 1973.

11. Anderson, J. R.: Language, Memory and Thought. Hillsdale, N.J.: Erlbaum, 1976.

12. Anderson, J. R.: Acquisition of cognitive skill. Psychological Review, 89, 1982, pp. 369-406.

13. Card, K. S., Moran, T. P., & Newell, A.: Computer text-editing: An information-processing analysis of a routine cognitive skill. Cognitive Psychology, 12, 1980, pp. 32-74.

14. Madni, A. M., & Lyman, J.: Model-based estimation and prediction of task-imposed mental workload. Proceedings of the Human Factors Society, 27th annual meeting, 1983, pp. 314-317.

Table 1

Production System for Performing POPCORN

```
P0:  If the goal is to play POPCORN
          and all tasks are finished,
     then pop the goal and END !!!

P1:  If the goal is to play POPCORN
          and the flight plan is presented and not read
          and an initial strategy has not been chosen,
     then the subgoal is to choose the initial strategy.

P2:  If the goal is to choose an initial strategy
          and the strategy has been chosen,
     then tag task X as the first task to begin working on
          and press 'return' on the keyboard,
          and pop the goal.

P3:  If the goal is to choose an initial strategy
          and the flight plan has not been read,
     then the subgoal is to read the flight plan.

P4:  If the goal is to read the flight plan,
     then read the flight plan
          and store the levels of the individual variables
             LEVEL(VARIABLE) in working memory (WM)
          and bring in the weights of the variables
             WEIGHT(VARIABLE) from long-term memory (LTM) to WM
          and initialize DIFF = 0, VARIABLE = 1
          and pop the goal.

P5:  If the goal is to choose an initial strategy
          and the flight plan is read and processed,
     then the subgoal is to weigh the variables.

P6:  If the goal is to weigh the variables
          and the strategy is tagged as chosen,
     then pop the goal.

P7:  If the goal is to weigh the variables
          and VARIABLE < 6,
     then DIFF = DIFF + LEVEL(VARIABLE) * WEIGHT(VARIABLE)
          and VARIABLE = VARIABLE + 1.

P8:  If the goal is to weigh the variables
          and VARIABLE ≥ 6,
     then the subgoal is to pick one strategy Si.
```

Table 1 (con't.)

```
P9:  If the goal is to pick one strategy Si
          and 0 ≤ DIFF < 2,
     then put strategy Si = S5 in WM
          and tag the strategy as chosen
          and pop the goal.

P10: If the goal is to pick one strategy Si
          and 2 ≤ DIFF < 4,
     then put strategy Si = S4 in WM
          and tag the strategy as chosen
          and pop the goal.

P11: If the goal is to pick one strategy Si
          and 4 ≤ DIFF < 6,
     then put strategy Si = S3 in WM
          and tag the strategy as chosen
          and pop the goal.

P12: If the goal is to pick one strategy Si
          and 6 ≤ DIFF < 8,
     then put strategy Si = S2 in WM
          and tag the strategy as chosen
          and pop the goal.

P13: If the goal is to pick one strategy Si
          and 8 ≤ DIFF ≤ 10,
     then put strategy Si = S1 in WM
          and tag the strategy as chosen
          and pop the goal.

P14: If the goal is to play POPCORN
          and the strategy is chosen
          and tasks are available for play,
     then the subgoal is to work on the tasks.

P15: If the goal is to work on the tasks
          and no tasks are available for play
          and no more tasks are expected to arrive,
     then pop the goal.

P16: If the goal is to work on the tasks
          and the strategy is to work on (i) tasks simultaneously
          and (i) tasks with kernels have not been opened,
     then the subgoal is to open (i) task boxes.

P17: If the goal is to open (i) task boxes
          and (i) boxes are open,
     then pop the goal.
```

Table 1 (con't.)

```
P18: If the goal is to open (i) task boxes
          and less than (i) boxes have been opened
          and task X is not selected,
     then the subgoal is to select task X.

P19: If the goal is to select task X,
     then move the mouse to task = X
          and click the mouse
          and pop the goal.

P20: If the goal is to open (i) task boxes
          and task X is selected
          and task X is not open,
     then the subgoal is to open task X.

P21: If the goal is to open task X,
     then move the mouse to function = OPEN
          and click the mouse
          and pop the goal.

P22: If the goal is to open (i) task boxes
          and less than (i) boxes have been opened
          and task X is open,
     then tag task X as the next new task (i.e., X = new task).

P23: If the goal is to work on the tasks
          and (i) task boxes are opened
          and kernels are popping out,
     then tag task X = task with the most popped kernels
          and the subgoal is to perform popped kernels.

P24: If the goal is to perform popped kernels
          and all kernels from the open task boxes are finished,
     then pop the goal.

P25: If the goal is to perform popped kernels
          and task X is not selected,
     then the subgoal is to select task X.

P26: If the goal is to perform popped kernels
          and task X is selected
          and task X has no popped kernels
          and task X' is open and has popped kernels,
     then tag X = X'
          and the subgoal is to select task X.
```

Table 1 (con't.)

```
P27: If the goal is to perform popped kernels
          and task X is selected
          and task X has popped kernels
          and the top kernel is green,
     then move the mouse to function = PERFORM
          and click the mouse.

P28: If the goal is to perform popped kernels
          and kernel(s) of task X is (are) in the warning zone
          and other kinds of kernels are also popping,
     then the subgoal is to stuff task X.

P29: If the goal is to perform popped kernels
          and kernels of task X are in the warning zone
          and no other kinds of kernels are popping
          and DIFF > 5,
     then the subgoal is to stuff task X.

P30: If the goal is to stuff task X
          and DIFF < 5,
     then move the mouse to function = STUFF
          and click the mouse
          and pop the goal.

P31: If the goal is to stuff task X
          and DIFF > 5,
     then move the mouse to function = STUFF
          and click the mouse
          and move the mouse to function = CLOSE
          and click the mouse
          and pop the goal.

P32: If the goal is to perform popped kernels
          and the kernels are in the warning zone
          and no other kinds of kernels are popping
          and DIFF < 5,
     then the subgoal is to process the warning state.

P33: If the goal is to process the warning state
          and the top kernel is green (i.e., warning state is
                    processed),
          then pop the goal.

P34: If the goal is to process the warning state
          and the top kernel is yellow,
     then move the mouse to function = Y->G
          and click the mouse
          and pop the goal.
```

Table 1 (con't.)

```
P35: If the goal is to perform popped kernels
          and a 20 sec. warning is flashing under closed task Y
          and other popping kernels are not in or near the
               warning zone
          (and task Y is not selected),
     then tag task X = Y (Y = task with warning flashing)
          and the subgoal is to select task X.

P36: If the goal is to perform popped kernels
          and a 20 sec warning is flashing under task X
          and task X is selected
          and task X is not open,
     then the subgoal is to open task X.

P37: If the goal is to perform popped kernels
          and a 20 sec warning is flashing under task Y
          and kernels of task X are popping "too fast",
     then the subgoal is to stuff task X.

P38: If the goal is to work on the tasks
          and no tasks are available for play
          and more tasks are expected to arrive,
     then wait for the new tasks.

P39: If the goal is to work on the tasks
          and (i) task boxes are opened
          and all kernels of these (i) tasks are finished
          and more tasks are expected to arrive into those boxes,
     then the subgoal is to close (5-i) task boxes.

P40: If the goal is to close (5-i) task boxes
          and (5-i) task boxes are closed,
     then pop the goal.

P41: If the goal is to close (5-i) task boxes
          and (5-i) task boxes are not closed
          and task box X is open (and empty) and selected,
     then the subgoal is to close task X.

P42: If the goal is to close task X,
     then move the mouse to function = CLOSE
          and click the mouse
          and pop the goal.

P43: If the goal is to close (5-i) task boxes
          and (5-i) task boxes are not closed
          and task X is not selected,
     then the subgoal is to select task X.
```

Table 1 (con't.)

```
P44: If the goal is to close (5-i) task boxes
         and (5-i) task boxes are not closed
         and task X is closed,
     then tag task X = new task to close.
```

Table 2

The Variables, Weights, and Levels of the Flight Plan

Variable	Description	Weight	Levels	Weight(level)
1	# of tasks to do	θ_1 (2)*	5 tasks	a_1 (0.25)*
			10 "	a_2 (0.50)
			20 "	a_3 (1.0)
2	# kernels/task	θ_2 (1)	2 kernels	b_1 (0.6)
			4 "	b_2 (0.8)
			8 "	b_3 (1.0)
3	speed of kernels	θ_3 (5)	slow	c_1 (0.1)
			moderate	c_2 (0.5)
			fast	c_3 (1.0)
4	arrival schedule	θ_4 (1)	massed	d_1 (0.8)
			staggered	d_2 (1.0)
5	warning state	θ_5 (1)	none	e_1 (0.0)
			yellow	e_2 (0.5)
			red	e_3 (0.75)
			invisible	e_4 (1.0)

* Note: The numbers in brackets are example values used in the example calculation below.

Example: Suppose the scenario to be played contains 10 tasks each with 4 kernels/task; the speed is moderate, the arrival schedule is staggered, and the kernels turn yellow in the warning zone. Then

$$\begin{aligned} \text{DIFF} &= \theta_1 a_2 + \theta_2 b_2 + \theta_3 c_2 + \theta_4 d_2 + \theta_5 e_2 \\ &= 2(0.5) + 1(0.8) + 5(0.5) + 1(1.0) + 1(0.5) \\ &= 5.8 \end{aligned}$$

=====> choose strategy S3.

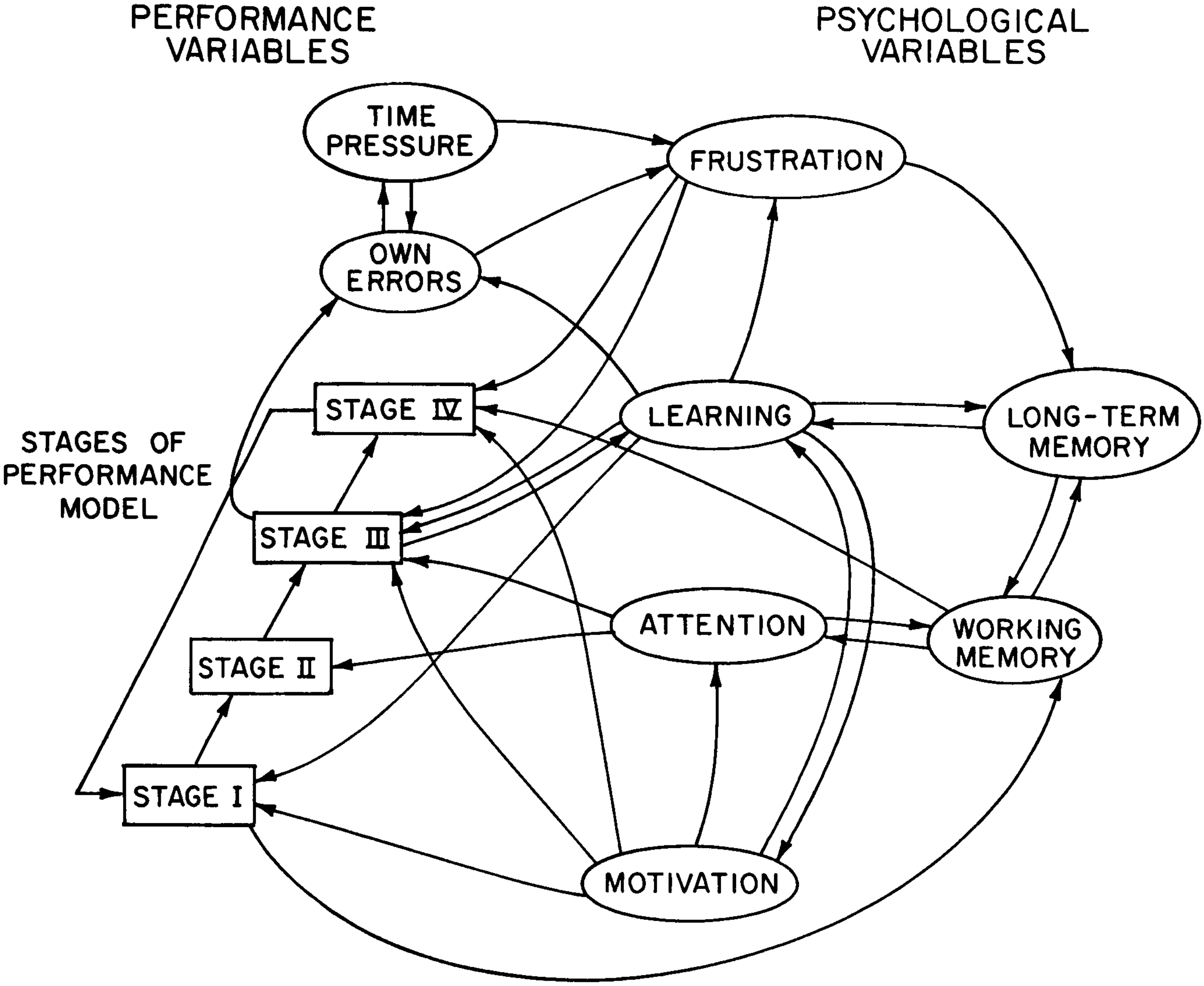

Figure 1. A dynamic psychological model showing the possible reciprocal relationships between the performance component of the model and the psychological variables.

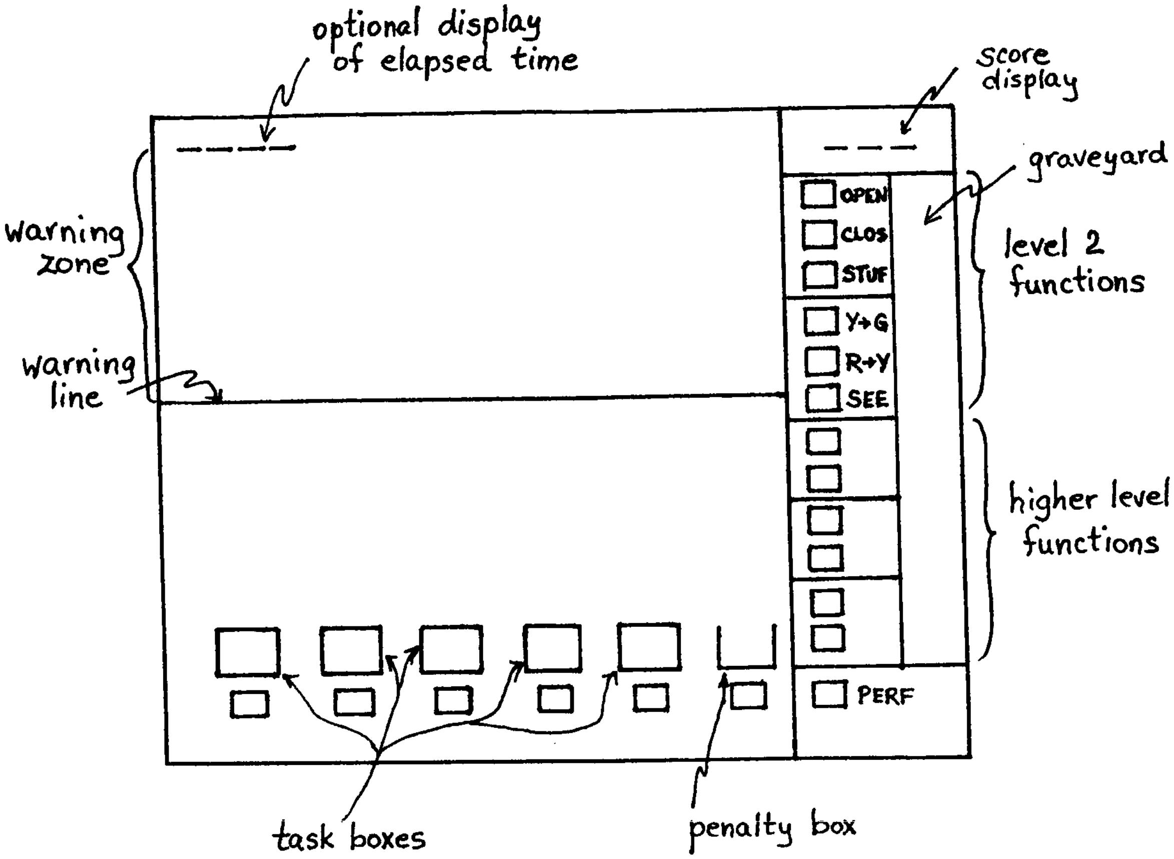

Figure 2. Monitor display of the POPCORN task at the second level of complexity.

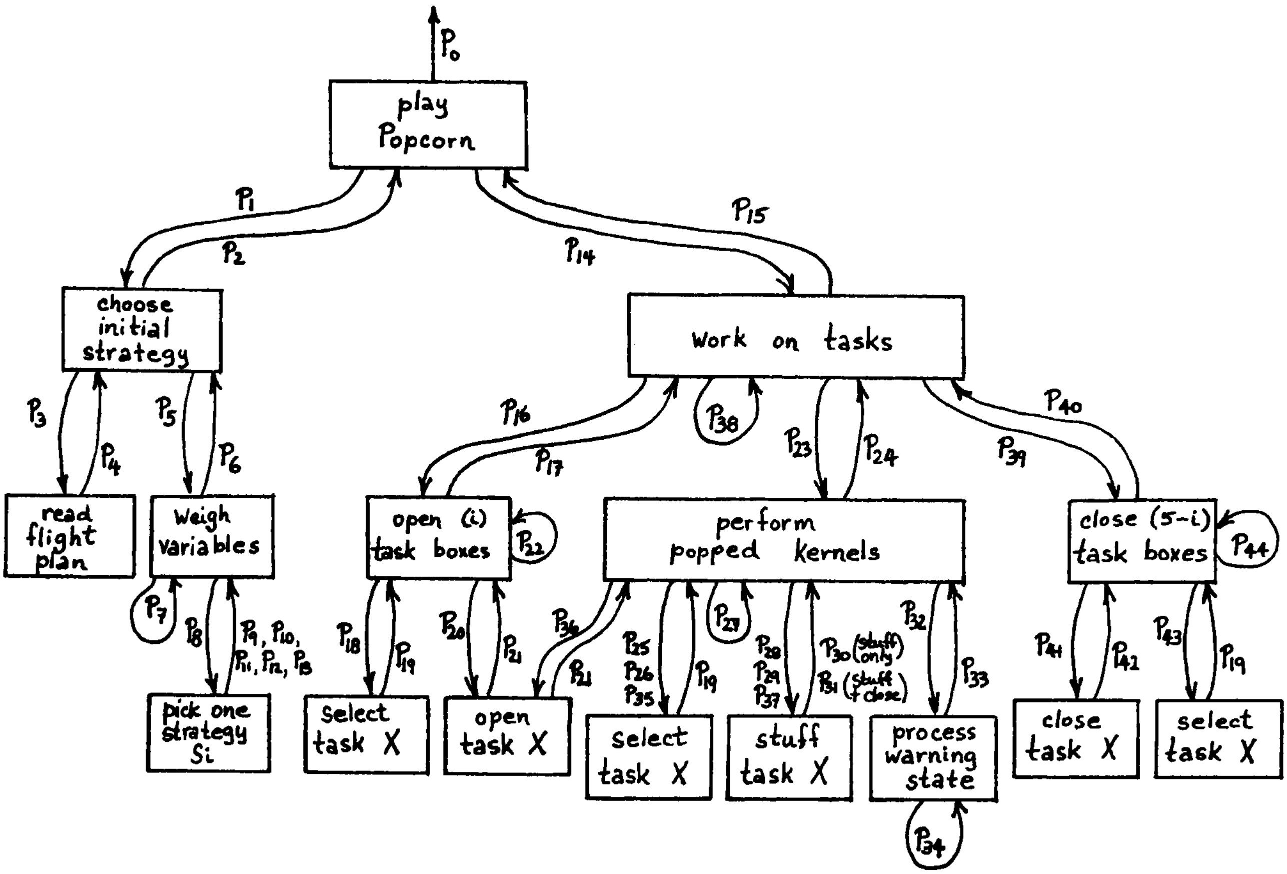

Figure 3. The goal structure for the production system of the performance component of POPCORN.