In Proceedings of ICCM - 2007- Eighth International Conference on Cognitive Modeling. 79 - 84. MI: Ann Arbor.

ACT-R Models of Cognitive Control in the Abstract Decision Making Task

Daniel Dickison (danieldickison@cmu.edu)
Niels A. Taatgen (taatgen@cmu.edu)
Department of Psychology, Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15201

Abstract

This paper discusses a method of modeling individual differences in cognitive control by developing models that differ in control structure. Such a strategy for modeling behavior is necessitated when tasks are complex and individual differences in performance vary on many measures of performance. For these tasks, merely adjusting a parameter to fit various groups of subjects may be impractical or impossible. Two such models for the Abstract Decision Making task are implemented in ACT-R to fit the performance of high and low-control subjects in the first experiment. These models are then used to accurately predict performance on a second experiment that involves novel "games" unrepresented in the prior experiment.

Keywords: Individual differences; cognitive control; ACT-R

Introduction

Computational models of cognition are often developed to fit the average performance on a task by a population. While this method suffices when describing phenomena that are observed widely, it purposefully ignores individual differences. One way to model individual differences is to propose some parameter – such as working memory capacity – that varies among individuals and observe how changes in that parameter affect the model's performance (Daily, Lovett & Reder, 2001; Taatgen, 2002; Rehling, Demiral, Lebiere, Lovett, & Reder, 2003; Chuderski, Stettner & Orzechowski, 2006).

However, when tasks are more complex, manipulating isolated parameters to fit individual differences becomes more challenging. For these types of tasks, it is possible to assume that individuals adopt different control structures that manifest themselves as different problem solving strategies. The different degrees of cognitive control can be characterized as the amount of top-down control that an individual exerts when completing a task, as opposed to behavior being primarily driven by bottom-up processes (Taatgen, 2007). This paper explores the development of distinct models using the ACT-R architecture (Anderson, 2007) that differ in control structure to describe individual performance on the Abstract Decision Making (ADM) task (Joslyn & Hunt, 1998).

Abstract Decision Making Task

ADM is a task developed by Joslyn and Hunt (1998) designed to predict individuals' performance on various real-world tasks that require decision making under time pressure. The task was used in a battery of tests designed to show individual differences in cognitive control.

ADM involves 5-minute long "games" that require subjects to classify objects into 1 of 4 bins according to their attributes. Both objects and bins have 3 attributes: size, color and shape. Only objects that match a particular bin's attributes are allowed in the bin. While objects always have 3 concrete attributes (e.g. small, red, circle), bins may specify any subset of the attributes (e.g. all circles). The number of attributes that a bin omits (i.e. the number of those that are wildcards) will be referred to as its *generality*. Thus, an "all circles" bin has a generality of 2 because it does not specify size or color. The generality of a game is defined as the greatest generality of its 4 bins.

Each game consists of 4 different bins that subjects study for as long as they want prior to starting the game. Because reviewing bins during the game is a slow and hence costly action, it is in the subject's interest to memorize the attributes of the bins. During the game, an object becomes available every 15 seconds with a pop-up notification that the subject must dismiss before continuing. None of the attributes of objects are immediately visible even when they become available. To reveal one attribute of an available object, the subject must query it by typing in the appropriate commands. To assign an object to a bin, the subject must type in a different set of commands. The subject receives points if an assignment is correct and loses points if it is incorrect. The magnitude of reward or penalty is inversely proportional to the generality of the bin – that is, more specific bins award more points.

Because there is a time limit and objects are presented quicker than most subjects can classify them, there is a time pressure urging subjects to act as quickly as possible. To analyze performance on the ADM task, we specified several measures that could be computed per subject for each game. The most obvious is the *score* measure, which is identical to the cumulative points that the subject earns for assigning objects to bins. For analysis, we normalized scores to a proportion of total possible points in a given game. Because penalties are assessed for incorrect assignments, negative scores are possible. A major problem with the score measure is that various factors play into determining the final score – including accuracy, speed, and whether subjects assigned objects to the more lucrative specific bins. To help tease those factors apart, *idealness* was defined as the proportion of correct assignments that were ideal. For any given object, there can be more than 1 bin to which it can be assigned, with 1 bin being the most specific and lucrative. *Idealness* is the proportion of correct assignments where the subject chose the best bin. Finally, *queries* was

defined as the average number of attribute queries a subject made of an object before attempting to assign it for the first time. This can range from 0 meaning the subject never queried any attributes – quite a suboptimal strategy – to 3 meaning the subject always queried every attribute. Values greater than 3 are also possible if the subject queried an attribute more than once.

Experiments

Two experiments were conducted. The first allowed the generation of models. The same models were used to predict subject performance in the second experiment, the results of which were used to validate the models.

Experiment 1

The task consisted of 2 practice games and 4 real games. Practice games used just 3 bins and had 20 seconds between object presentations. The 4 real games consisted of 2 games each of generality 0 and generality 1. Forty-one people from the Carnegie Mellon University community participated as part of a larger experiment studying individual differences. Models were developed to fit the pattern of data observed in experiment 1.

Experiment 2

The task consisted of just 1 practice game followed by 4 real games. The real games increased in generality from 0 in game 2 to 3 in game 5. Fifty-three subjects participated in experiment 2.

Models

Two ACT-R models were developed to describe subject performance in the ADM task – one for a high-control strategy and one for a low-control strategy. Because it is unclear which aspects of the task give rise to particular subject performances, an attempt was made to model the task as closely as possible to the actual experiment. The model, like subjects, must type in certain commands in response to text prompts to query and assign objects. This is accomplished via ACT-R's perceptual and motor modules, which simulate the amount of time required to process visual stimuli and perform key presses. The amount of time required for the models to process textual prompts was adjusted so that the models classified roughly the same number of objects per game as did subjects. This was a simplification to reflect the amount of time people need to parse a prompt, because the models were able to deduce the current state of the textual interface as soon as the visual stimuli was encoded. One further simplification made in modeling the task was that the new object pop-up notifications were omitted for the model runs.

In both models, the objects are processed in order as they become available. Neither model moves on to a subsequent object until the current object has been successfully assigned to a bin. Each bin is stored in declarative memory. A representation of the current object is kept in the imaginal buffer, which is ACT-R's mechanism for temporarily storing the current problem state. The imaginal buffer representation includes slots for the object attributes that get filled in as they are queried. The imaginal buffer was configured to spread activation to items in declarative memory, so that when deciding which bin to assign an object to, bins that had matching attributes to those of the imaginal buffer representation were more likely to be retrieved than other bins.

The differences between the low-control and high-control models are outlined below.

Low Control Model

This model (Figure 1) makes use of the bottom-up process of expected utility available as a module in ACT-R (Anderson, 2007) to decide at each step whether to query more object attributes or to attempt an assignment. Two exceptions are when no attributes of an object are known, then it will always query, and, conversely, when all attributes are known, it will not query any more. The "query" production rule's utility is held constant while the "retrieve bin" production rule will gain or lose utility based on past successes and failures. When an assignment is correct, a positive reward is propagated backwards through the production rules that had fired leading up to the assignment. As a result, the "retrieve bin" production rule gains utility relative to the "query" rule, thus, in the future the model is more likely to ask fewer questions before assigning. On the other hand, when an incorrect assignment is made, a negative reward (i.e. a penalty) is propagated to the "retrieve bin" rule, leading to more queries before assignment.

When a bin is retrieved, it is checked against the currently known attributes of the object in question. If there is a mismatch, the model reverts to the state where it may attempt another retrieval or query for another attribute. Otherwise, if it is a match or if it *might* be a match, it will attempt to assign.

This model makes incorrect assignments when the retrieved bin specifies attributes that have not yet been revealed from the object. For example, after only having determined that a given object is red, the model may retrieve a bin that will take red circles. Even though the red object may not be a circle, the low control model will still try to assign the object to this bin, possibly resulting in an error. This leads to a corresponding negative reward, thus slightly biasing the model towards querying more in the future.

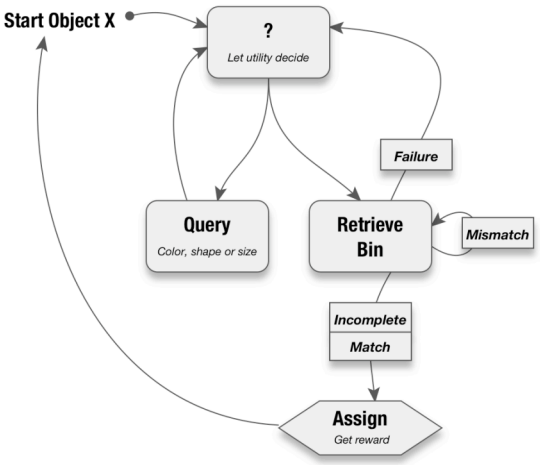


Figure 1: Low control model flow diagram.

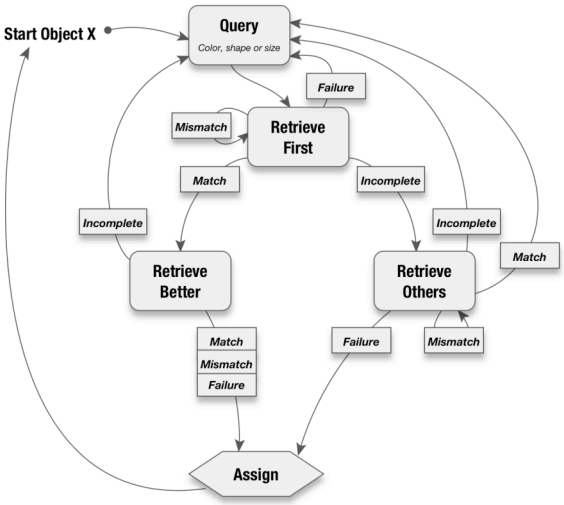


Figure 2: High control model flow diagram.

High Control Model

The high control model (Figure 2) follows a more disciplined strategy by exerting additional top-down control. Instead of relying on the expected utility of querying versus assigning, the model attempts to maximize correctness and points gained by evaluating one bin at a time to see if it will take a given object. It also tries to minimize queries by assigning to a potentially matching bin if no other candidates can be retrieved. Specifically, it will always query first, then, after each query, it will attempt to retrieve a matching bin. If a matching bin is found, it tries to retrieve a better match – if one is found, it assigns the object to the better bin, and if one is not found, it assigns to the original match it had found earlier. If, after the original retrieval, a *maybe* matching bin is retrieved, it attempts to retrieve other bins. If no other matches are found, then it will assign to the first "maybe" bin it had retrieved, on the assumption that the object must fit the only possible bin.

An example will help illustrate the model's logic and how it minimizes the number of queries while maintaining accuracy. Suppose there are just 2 bins, one for small red circles (called A) and another for any small red shape (called B). When the model starts processing an object, it may query the object's shape. Suppose this object is a triangle. The model now tries to retrieve a bin, and retrieves the bin B. The model recognizes that the triangle may fit in this bin, but the color and size could possibly mismatch. The model now moves to the "retrieve others" state and attempts to retrieve other bins. It retrieves bin A, and sees that the triangle is a mismatch for this circles-only bin and discards it. It then fails to retrieve any other bins because there are no other bins for this game. It then concludes that the first bin – bin B – must be the correct bin, and assigns the triangle to the small red shapes bin. Thus, after just 1 query the model was able to correctly classify the object.

This model only makes errors when it fails to retrieve matching bins after first having found a possible candidate. This is, during the "retrieve others" state, it fails to retrieve a "maybe" bin even when one exists in its declarative memory. This outcome results from the fact that bin retrievals are subject to activation decay and noise.

Results and Model Fits

In order to fit the models to subjects, subjects were divided into a high-control group and a low-control group based on certain idealness and queries criteria. Each model was then developed to fit a group of subjects in experiment 1. Apart from the control structures of the models, their speed of processing textual prompts was adjusted so that they classified roughly the same number of objects that subjects classified during each game. Finally, predictions were made for experiment 2 using the exact same models, and their outputs were compared to data observed in experiment 2.

Control Grouping

The criteria for splitting subjects into groups was based on the idealness and queries measures for the games which had generalities of 0 and 1 – that is, games involving no wildcards in bin attributes, and games additionally involving bins with 1 wildcard, respectively. Because in games with generality 0 each object can only be assigned to 1 bin, the idealness measure is necessarily 1.0 for all subjects. With the addition of generality 1 bins, each object can be assigned to either 1 bin (the most specific bin), or to 2 bins. If a subject always assigns to the best bin, idealness will still be 1.0. If, on the other hand, the subject assigns to the more general, less lucrative bin(s), idealness will decrease. Figure 3, shows the distribution of subjects' idealness measures for experiment 1, generality 1, while Figure 4 shows the same measure for the varying game generalities. A somewhat bimodal distribution is evident at generality 1, with many

subjects getting a perfect 1.0 and a cluster of subjects getting a lower value around 0.7 through 0.9. The bimodality is further exemplified in games of increasing generality. These distributions suggest that two distinct strategies were used by subjects, rather than subjects varying on one continuous dimension. We deemed subjects who had idealness greater than 0.9 in the generality 1 games to be potentially high-control, and others as low-control.

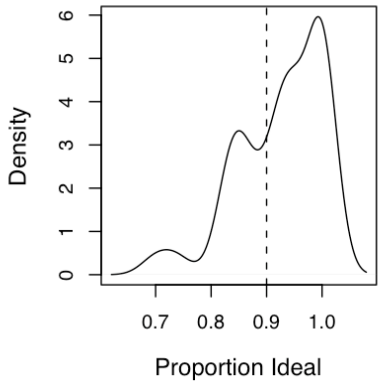


Figure 3: Distribution of experiment 1 subjects' *idealness* measure for games of generality 1 as a density plot. The dashed line shows the cutoff for high-control classification on the idealness measure.

In addition to the idealness criterion, a criterion based on the number of queries was used for control group classification. The motivation for adding another criterion is that using the idealness criterion alone, subjects would be classified as high-control if they simply attempted to assign objects to the most lucrative bins first, and, failing that, try the more general bins. This strategy maximizes idealness while minimizing the number of queries to the detriment of overall score. In order to filter out such strategy takers from high control classification, we required subjects to show an increase in queries when playing generality 1 games as compared to when playing generality 0 games. Because of the configuration of bins, more queries are required to find the ideal bin when there are wildcards involved.

In experiment 1, 27 subjects out of 41 were classified as high control on the idealness criterion, with 25 of those also matching the queries criterion for high control. For experiment 2, these numbers were 24 out of 53 for the idealness criterion and 18 of those matching the queries criterion also.

Model Comparison

Having grouped subjects thus, the outputs produced by the models were compared to the observed data. Figure 5 shows the overall score, number of queries, and proportion ideal for each game generality in experiment 1. The two practice games are excluded from analysis, and the 2 games of each generality are collapsed.

Of note is that, regarding idealness, the low control model reflects the degree to which subjects decrease idealness in generality 1 games, while the high control model maintains a near-perfect mark. Furthermore, the high control model increases the number of queries as it moves to the more general games as do subjects. This is noteworthy because the high control model is designed to minimize the number of queries, but it is sensitive to the fact that more queries are necessary for high performance, a feat made possible by its complex control structure.

The group classifications are good predictors of overall score in experiment 1, producing a main effect of group on score, $F(1, 160) = 52.6$, $p < 0.01$, in addition to a main effect of game generality on score, $F(1, 160) = 20.4$, $p < 0.01$.

Figure 6 shows the same measures plotted for experiment 2. Note that in this experiment, there was 1 game each of the 4 levels of generality, and there was only 1 practice game as opposed to 2. That may explain the lower overall performance of subjects in this experiment compared to experiment 1.

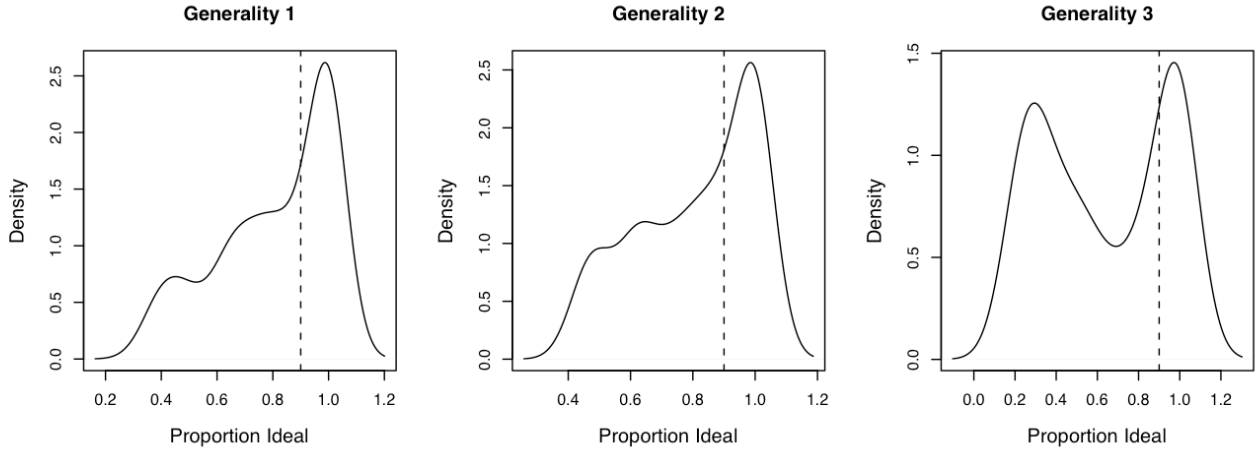
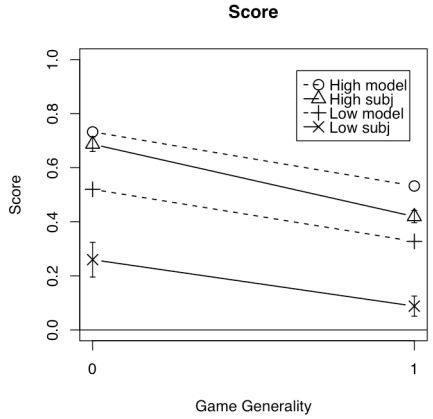
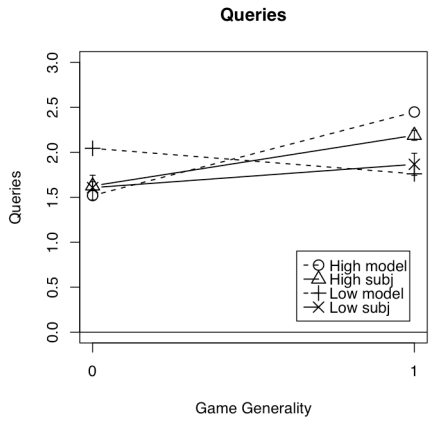


Figure 4: Distributions of experiment 2 subjects' *idealness* measure for games of varying generality.





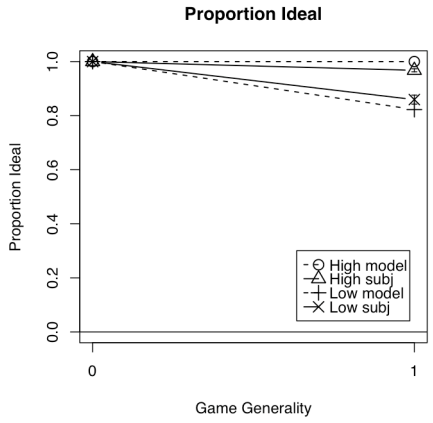
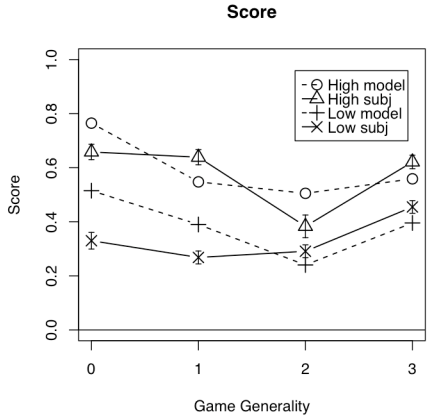
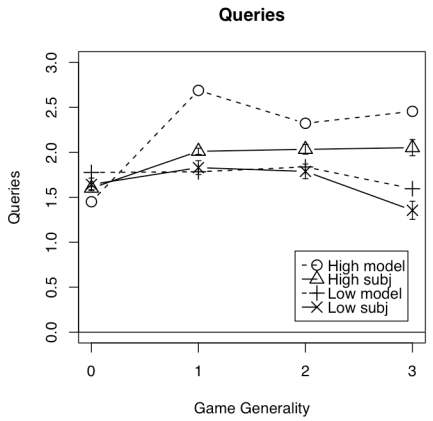


Figure 5: Experiment 1 data (solid lines) and model fits (dashed lines) for high control (open shapes) and low control (crosses). Error bars, shown only for subject data, represent standard error.





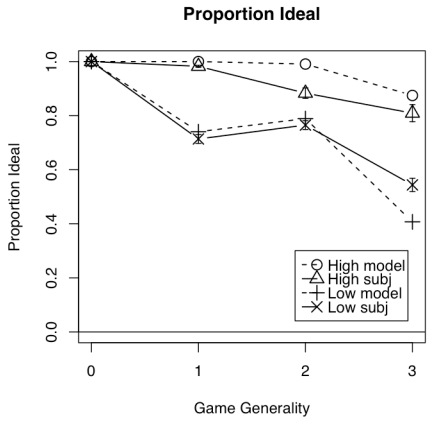


Figure 6: Experiment 2 data and model fits.

For these data, models were not explicitly designed to fit the data but instead were extrapolated from those designed for experiment 1. The fits for game generality 2 and 3, which were nonexistent in experiment 1, show that the models predict subject behavior in the right directions. The most notable deviation is the exaggeration of differences in the number of queries between the high and low control models. The high control subjects maintain the number of queries at