The Strategic Use of Memory for Frequency and Recency in Search Control

Andrew Howes (HowesA@cardiff.ac.uk)
School of Psychology, Cardiff University, Cardiff, CF10 3YG, Wales, UK

Stephen J. Payne (PayneS@cardiff.ac.uk)
School of Psychology, Cardiff University, Cardiff, CF10 3YG, Wales, UK

Abstract

A requirement of an information processing account of human problem solving is that it includes a mechanism by which people remember which goals and operators have been evaluated and which still need to be evaluated. One might expect that these are issues of such fundamental importance that they must have been solved or at least addressed by the two architectural accounts of cognition (Soar and ACT-R), but in fact it is an issue that is glossed in both. We identify two problems: (1) Soar and ACT-R guarantee information about goals, and (2) ACT-R combines measures of frequency and recency into a single representation of activation. In this paper we report a model of how people search simple binary trees. The model demonstrates the cognitive plausibility of a search algorithm that is supported by a memory system that delivers independent estimates of frequency and recency.

Introduction

A requirement of an information processing account of human problem solving is that it includes a mechanism by which people remember which goals and operators have been evaluated and which still need to be evaluated. Whether the task is the Tower of Hanoii, a waterjugs problem, a world-wide web search problem or a spatial navigation task, a person engaged in search examines the consequences of applying an operator to a state by trying it out and perceiving to which state it, and subsequent operators, lead. At some point in the future, the person may, through backup, or because of loops, find themselves in a visited state. Recognition that the state has already been visited and/or that the operator has already been applied to this state, will in the long-term help prune the search space and thereby constrain the effort spent on attaining the goal. This constraint has been used in a number of models of human problem solving (Atwood & Polson, 1976; Jeffries, Polson., Razran, & Atwood, 1977; Anderson, 1993; Howes, 1994). Atwood & Polson's model of human performance on the waterjugs problem, built up a representation of the 'familiarity' of states that was factored into the operator selection process. The more familiar an operator then the less likely it was to be selected.

One might expect that these are issues of such fundamental importance that they must have been solved or at least addressed by the two substantial architectural accounts of cognition (ACT-R, Anderson, 1998; Soar, Newell, 1990), but in fact it is an issue that is glossed in both. In Soar, the architecture automatically ensures that operators that have already been applied to a particular state in pursuit of a particular goal (on the goal stack) on a particular trial will not be reselected. In ACT-R the goal stack has privileged status. Items posted on the stack are not subject to the constraints of memory, i.e. they do not have decaying activation and cannot therefore be forgotten (Altman and Trafton, 1999).

Another resource for supporting decisions about which operator to apply is memory for previous attempts at a goal (either successful or failed). If a goal has been achieved prior to the current attempt then memories that indicate that an operator is familiar may be taken as evidence that it is more likely to lead to the goal than an unfamiliar operator (Payne, Richardson, Howes, 2000). However, an issue for the problem solver is how to determine the source of the familiarity. If the source is the current trial then the operator should be rejected, if it is a previous trial then perhaps it should be selected.

Payne, Richardson, Howes (2000) investigated the role of familiarity (Jacoby, 1991) in controlling interactive search. They tested the hypothesis (Aasman & Akyurek, 1992; Howes, 1994) that people help control search merely by recognising the actions that have been tried before and found that the familiarity of items could affect decisions about which item to select. Moreover familiarity was used strategically. When participants had information indicating that familiarity would be more likely to indicate that an operator would lead to the goal, they were more likely to use familiarity to guide selection.

Again, one might expect that this issue would have been addressed in architectural theories of cognition. However, while Soar's chunking mechanism is flexible, the issue of whether it can provide a mechanism for representing the episodic familiarity of an operator has only recently started to be explored (Altmann and John, 1999). The situation for ACT-R is more complex.

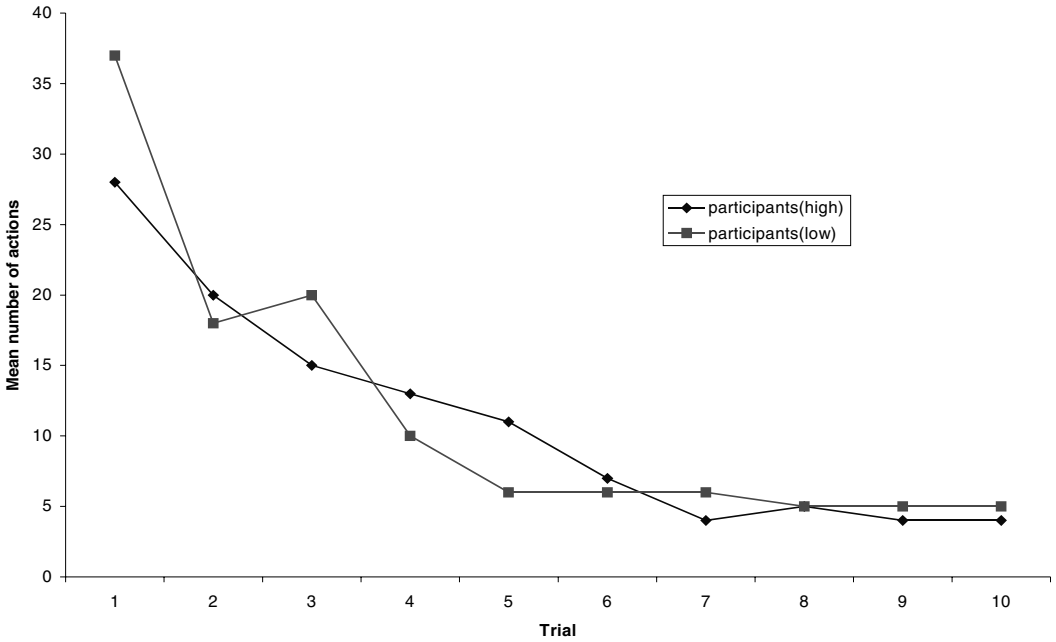

Figure 1: Mean number of actions taken by high and low systematicity participants

In ACT-R, each chunk stored in declarative memory has an activation that is used to determine probability of retrieval. This activation is made up of a base-level activation and an associative activation. Anderson and Lebiere (1998; page 70) state: "... the activation of a chunk is a sum of the base-level activation, reflecting its general usefulness in the past, and an associative activation, reflecting its relevance to the current context," and, "The base-level activation of a chunk represents how recently and frequently it is accessed."

Importantly however, the frequency and recency components of base-level activation are not independently inspectable by the production rules and it is not therefore possible to write ACT-R production rules that make strategic use of frequency and recency information stored as components of chunk activations. It seems unlikely therefore that it is possible to write productions that, for example, prefer the most frequent operators at the expense of the most recent.

A commonly used solution to this in ACT-R models has been to use flags on declarative memory structures. A flag is added to operators that have been applied on this trial and then all flags are wiped at the end of the trial (Anderson's model of navigation 1993; Lebiere, personal communication) leaving no episodic evidence that they had ever been there. While, this is an extra-architectural mechanism that, unsurprisingly, is not claimed as part of the theory, its use undermines the claim that models constructed in ACT-R are subject to a principled set of memory constraints.

In this paper we report a computational level model of how people search simple binary trees. The model makes strategic use of frequency and recency information and demonstrates the cognitive plausibility of a search algorithm that is supported by a memory system that delivers independent estimates of frequency and recency.

Task

In a series of studies to be reported elsewhere we observed participants searching simple binary word-mazes. Each maze was a binary tree structure with a depth of 5 nodes. At each choice point participants were presented with three buttons on a computer display. Two buttons at the top and bottom of the right of the screen were labelled with different words (perhaps 'gun' and 'pistol') and the other button, on the left of the screen, was labelled 'back'. Selection of one of the two buttons on the right changed the current state to a state nearer to the leaves of the tree and selection of 'back' moved the state to a node nearer the root of the tree. Participants were asked to search for a leaf node with a given label (a random word).

Observations:

- All participants were able to complete these search tasks.
- Three strategies were used:
  - **Random search** with a forward bias. Participants selected either the top or the bottom button on the right of node X, searched the subtree and then on returning to X selected the other button.

- **Systematic search**. Participants always selected the top button on the first visit to node X, searched the subtree, and then on return to X selected the bottom button.
- **Memory-based search**. On trials after the first participants generally attempted to remember the correct path.

- On trials after the first, participants flexibly interleaved search based on memory for previous trials with, when memory for previous trials failed, either systematic or random search.
- None of the participants perseverated, i.e. they did not repeatedly search the same incorrect subtree more than a handful of times.
- With practice (about 4 trials) all participants were able to follow the correct path with relatively few errors (Figure 1).
- Those participants who used a systematic strategy were significantly more efficient than those who did not. On the first trial the variation in the performance of the systematic participants was less than the variation in the performance of the random participants. (Unfortunately, the statistically significant difference in efficiency between the use of the two strategies is not reflected in Figure 1. This is because some participants using the random strategy can, luckily, find the goal with relatively few actions.)

Model

The first model that we built relied on a single activation-based measure that combined both frequency and recency information. This model could perform the first trial of a task by avoiding operators with high activation (those inferred to have been selected recently or frequently). However, on subsequent trials, a strategy of preferring operators with a higher activation (i.e. the ones used most recently on the previous trial or the ones used most frequently over trials) proved to be fragile. Activation may be high either because an operator was selected many times incorrectly or because it was selected more recently (i.e. closer to the achievement of the goal). Worse, if an error is made because an algorithm prefers highly active operators, then the algorithm may perseverate ad infinitum on incorrect selections.

The model that we focus on in this paper, is an extension of a proposal by Payne, Richardson, Howes (2000). It relies on the separate and strategic use of information about the frequency and recency of operator usage. The model is not based on assumptions about the structure of memory, rather it is based on assumptions about what information memory can deliver. The heuristics that define the search algorithm rely on the following functions for acquiring information from memory:

- **F = frequency(I)** - returns an estimate of the frequency F of item I.
- **X = most_recent(P)** - Instantiates pattern P to its most recent occurrence. (e.g. most_recent(op) would bind X to the most recently tried operator). Only one value can be returned for a particular P.
- **F = freq_before(I, E)** - returns the frequency F of I before the most recent occurence of event E. (e.g. to give the frequency of an item I before the selection of the current goal.)
- **F = freq_after(I, E)** - returns the frequency F of I after event E.

In order to simulate a lack of reliability in the information returned by these functions, frequency and recency information decayed from memory stochastically. Also, false positives were randomly generated in answer to queries about whether operators had been applied on this trial. In the Payne, Richardson, and Howes (2000) experiment, false positives occurred when participants were forced to make a decision about whether or not they had applied an operator before. In fact, participants may have only seen the operator and not applied it. The functions that determined the rate of decay and false positives are not important for our current purposes.

The purpose of introducing the errors was not to capture some quantitative aspect of the data but instead to ensure that the search algorithm was robust given the return of incorrect information from memory. Most importantly the algorithm should not perseverate implausibly even when degraded information is returned from memory.

The heuristics work by adding to a preference value for each operator proposed. There are three sets of heuristics: those that switch algorithm (or strategy); those that control systematic search; and those that control frequency-based search.

Given goal G, operator Op and a preference constant V, the rules for each algorithm are described below. The rules depend on memory encodings of the frequencies and recencies of associations, in general between G and Op, but for clarity. a short-hand has been used to describe the rule conditions, which does not refer to the association per se, but instead just to Op. Each rule proposes an addition (plus) or a subtraction (minus) to the current value of the preference for Op. The rules are described in a pseudo-code where variables are represented with capitals. The symbol '=' indicates a test of equality. If the test has a variable on either side and the variable is not already bound then the test will result in binding. The variables TOP and BOTTOM are respectively bound to the top and bottom forward menu selections.

Rules 1 to 5 describe the memory-based algorithm. This algorithm is used if the model has a memory

indicating that the goal has been achieved before. Rules 6 and 7 describe the random algorithm. Rules 8 to 10 describe the systematic algorithm. (A particular instantiation of the model uses either the random or the systematic rules but not both.) Finally, rule 11 switches to the memory algorithm and rule 12 restarts a search in the case of apparent exhaustion (this is described further below).

There is only space to describe some of these rules here. We will focus on those for the systematic algorithm. Rule 8 says, if the most recent algorithm is systematic and the operator (Op) being evaluated is a forward operator at the top of the screen, and the most recent of the previously applied operators (R) was not a 'back' operator THEN add V to the preference for Op. Rule 9 is similar to rule 8 but adds a preference for the forward operator at the *bottom* of the screen if the previous operator was a backup, Lastly, rule 10 prefers the back operator when the bottom operator has been tried on this trial and the most recent previous operator was also a back.

1. IF most_recent(algorithm) = A,
A = use_memory,
forward(Op) = true,
freq_after(Op,A) = 0,
freq_before(achieved(G), A) = FG,
freq_before(Op, A) = FO
THEN P becomes plus(1 / (1 + abs(FO – FG) * V))

2. IF most_recent(algorithm) = use_memory,
forward(Op) = true,
freq_before(fail(Op), now) = FN
THEN P becomes minus(FN * V).

3. IF most_recent(algorithm) = A,
A = use_memory,
forward(Op) = true,
freq_after(Op, A) = 0,
freq_before(Op, achieved(G)) = OF,
THEN P becomes plus(OF * V).

4. IF most_recent(algorithm)=A, A= use_memory,
forward(Op) = true,
freq_after(Op, A) = 0,
THEN P becomes plus(V).

5. IF most_recent(algorithm)=A, A= use_memory,
back(Op) = true,
freq_after(TOP, A) > 0,
freq_after(BOTTOM, A) > 0,
THEN P becomes plus(V).

6. IF most_recent(algorithm) = A, A = random,
forward(Op) = true,
freq_after(Op, A) = 0,
THEN P becomes plus(V).

7. IF most_recent(algorithm) = A, A = random,
back(Op) = true,
freq_after(TOP, A) > 0,
freq_after(BOTTOM, A) > 0,
THEN P becomes plus(V).

8. IF most_recent(algorithm) = systematic,
forward(Op) = true, top(Op) = true,
most_recent(op) = R, not(back(R) = true),
THEN P becomes plus(V).

9. IF most_recent(algorithm) = A, A = systematic,
forward(Op) = true, bottom(Op) = true,
most_recent(operator) = R,
back(R) = true,
freq_after(Op, A) = 0,
THEN P becomes plus(V).

10. IF most_recent(algorithm) = A, A = systematic,
back(Op) = true,
most_recent(operator) = R,
back(R) = true,
freq_after(BOTTOM, A) > 0,
THEN P becomes plus(V).

11. IF most_recent(algorithm) = A, A = none,
freq_before(achieved(G), A) > 0,
Op = algorithm(use_memory),
THEN P becomes plus(3*V).

12. IF current_node = root ,
most_recent(algorithm) = A ,
Op = algorithm(A),
freq_after(TOP, A) > 0,
freq_after(BOTTOM, A) > 0,
THEN P becomes plus(3*V).

The last algorithm switching rule (rule 12) plays a crucial role. Occasionally the problem solver will return to the root node without having found the goal. This will happen if the search was incomplete (i.e. some subtree remained unsearched) due to inadequate information from memory (a false positive). In this situation rule 12 restarts the search. In the model this is operationalised as the operator for the current algorithm is reapplied. The time at which the most recent algorithm operator was applied is used by the other rules to judge whether memories for operator applications were part of the current trial or previous trials.

Results

For particular rates of memory decay and false positives, the model was run 40 times on each of the 4 tasks performed by participants. The resulting mean performance for three decay rates is shown in Figure 2.

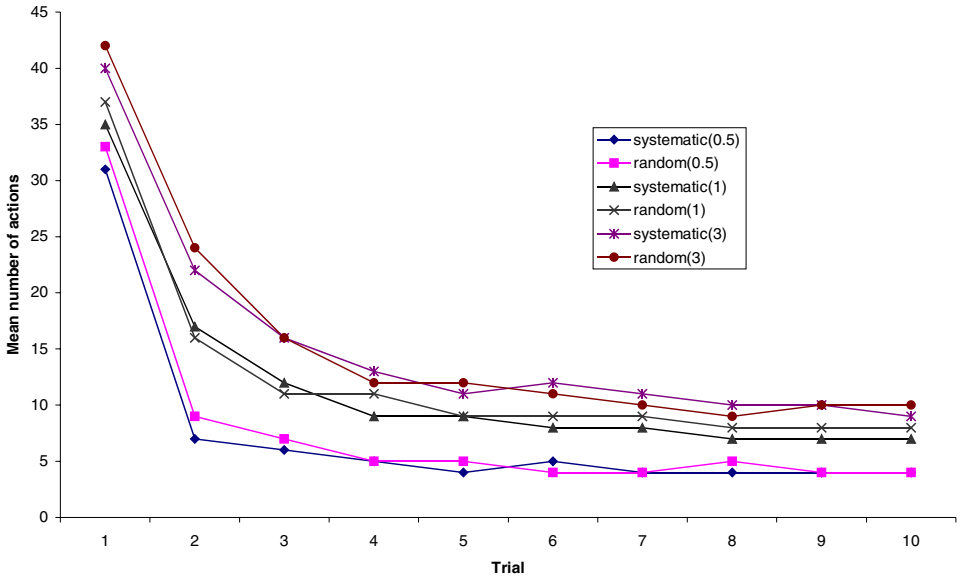

Figure 2: Mean number of actions taken by model given increasingly unreliable information from memory

The participants' mean performance is within the bounds of the best and worst model performance illustrated in Figure 2. We have not attempted to fit the model precisely, rather in accordance with Roberts and Pashler (2000) we explored the range of its behaviour.

Importantly, the model did not perseverate. Regardless of errors made during search, it always recovered and eventually found the goal. Also, as the decay rate increased the model was still able to learn the task. A large number of errors in the first trial did not on average incapacitate the learning over subsequent trials.

The gradual improvement in practice after the first trial was a result of a search algorithm (rules 1 to 5) that is guided by a combination for memory for previous trials and the current trial. If memory for previous trials proved inadequate then memory for the current trial, as distinguished by relative recency, ensured a reasonably efficient search.

Also, in accordance with the participants behaviour, the systematic algorithm produced more efficient and less varied searches on trial 1.

Discussion

The model reported here demonstrates that aspects of the way in which people search and learn paths through external problem spaces can be captured with heuristic rules that make strategic use of independent estimates of the frequency and recency of previously selected operators. Without access to this information it is impossible to write heuristics that distinguish an operator with high frequency from one that has high recency, and it is therefore a problem to determine whether key events occurred on the current trial or previous trials. The analysis of the model's behaviour under a range of memory decay and false positive conditions reveals that it produces behaviour broadly similar to human performance on a simple search task. Notably, unlike previous activation-based models built by the authors, the model does not perseverate when receiving degraded information from memory. In addition, the mean performance of the model over ten trials consists of a practice curve similar to that of the participants.

However, further investigation revealed that, after the first trial, the model produced a much greater variation in behaviour than the participants in the experiment. This issue is a matter for further investigation, and may well imply the need for some superordinate learning mechanism (perhaps rehearsal or impasse-driven learning).

A superordinate learning mechanism might involve the deliberate encodings of what the correct option is. This is an approach that was explored in Howes (1994), and while it deserves further attention, there are two problems. The first is that there is a dislocation in time between when the items are experienced and when a participant achieves the goal. In previous models the feedback of information about correctness produced recency effects in which lower levels of the tree were learnt first (Howes, 1994). These effects were not observed in our experiments. The second is that deliberate learning only pushes the problem back one level. If people deliberately learn what is correct then when situations change or mistakes are made, they also have to deliberately learn that a different option is

correct. Subsequent competition between different representations of correctness would then have to be resolved, perhaps using exactly the kind of mechanism that we have proposed. Progress will require modelling the range of individual trial data rather than just mean data.

Another possibility that we are investigating is that the long-term learning is based on recency and not frequency. Once the goal has been achieved, then the operator that led to the goal will be the most recently selected operator at any choice. Memory for recency could therefore be used to guide learning. However, under normal assumptions about decay, a recency-based model predicts that choice points at different distances from the goal would be learnt at different rates. Our data (not described above) does not support this prediction.

In principle, it may be possible to construct algorithms in ACT-R designed to ensure that during search sufficient episodic information is stored in declarative chunks to enable the kinds of computations that are posited in the model report here (e.g. Altmann and Trafton, 1999). However, regardless of the success of this approach, there will remain an issue about how people obtain information about frequency, and recency. While the concept of activation is well established in psychology, an architecture in which chunks are stored with independent measures of frequency and recency may lead to more parsimonious accounts of problem solving behaviour.

There are a number of models of the cognitive activity that give rise to practice effects, amongst them Logan's (1988) instance model and Rosenbloom and Newell's (1981) chunking model. More recent work has emphasised the strategy specific nature of the practice curves (Delaney, Reder, Staszewski & Ritter, 1998). The model reported here is similar to Logan's in that the practice curve emerges as a result of encodings made from experience with the external environment: however, maze-like tasks are more complex than simple letter arithmetic tasks and it is for this reason that our model requires the combination of frequency and recency dependent control mechanisms that we have described.

In the introduction we claimed that ACT-R's representation of undifferentiated activations was not sufficient to directly support algorithms that capture the behaviour of people engaged in typical search tasks. In contrast, the model that we have reported illustrates the cognitive plausibility of a mechanism that makes strategic use of separate sources of operator recency and frequency during search.

Acknowledgement

Juliet Richardson contributed a great deal to the development of the ideas presented in this paper.

References

Aasman, J. & Akyurek, A. (1992). Flattening goal hierarchies. In J.A.Michon & A.Akyurek (eds.) *Soar: A Cognitive Architecture in Perspective* , 199 -217. Kluwer.

Altmann, E.M. & John, B.E. (1999). Episodic indexing: A model of memory for attention events. *Cognitive Science, 23*(2),117-156.

Altmann, E. M. & Trafton, J. G. (1999). Memory for goals: An architectural perspective. *Proceedings of the twenty first annual conference of th e Cognitive Science Society* (pp. 19-24). Hillsdale, NJ: Erlbaum.

Anderson, J. R. (1993). *Rules of the mind*. Hillsdale, NJ: Erlbaum.

Anderson, J. R. & Lebiere, C. (1998). *The Atomic Components of Thought*. NJ: Erlbaum.

Atwood, M. E. & Polson, P. G. (1976). A process model for water jug problems. *Cognitive Psychology, 8*, 191-216.

Delaney, P. F., Reder, L. M., Staszewski, J. J., & Ritter, F. E. (1998). The strategy-specific nature of improvement: The power law applies by strategy within task. *Psychological Science, 9*, 1-7.

Howes, A. (1994). A model of the acquisition of menu knowledge by exploration. In B. Adelson, S. Dumais, J. Olson (Eds.) *Proceedings of Human Factors in Computing Systems CHI'94* (pp. 445-451), Boston, MA.: ACM Press.

Jacoby, L. L. (1991). A process dissociation framework: Separating automatic from intentional uses of memory. *Journal of Memory and Language, 30*, 513-541.

Jeffries, R. P., Polson, P. G., Razran, L. & Atwood, M. (1977). A process model for missionaries-cannibals and other river-crossing problems. *Cognitive Psychology, 9*, 412-440.

Logan, G. D. (1988). Toward an instance theory of automatization. *Psychological Review, 95*, 492-527.

Newell, A. (1990). *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the power law of practice. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, New Jersey: Erlbaum.

Payne, S. J., Richardson, J. & Howes, A. (2000). Strategic use of familiarity in display-based problem solving. *Journal of Experimental Psychology-Learning Memory and Cognition, 2*, 1685-1701.

Richardson, J., Howes, A., & Payne, S.J. (1998) A cognitive model of the use of familiarity in the acquisition of interactive search skill. In M.A. Gernsbacher & S. J. Derry (Eds.), Proceedings of the 20th Annual Conference of the Cognitive Science Society (p. 1258). Mahwah, NJ: Erlbaum.

Roberts, S., Pashler, H. (2000). How persuasive is a good fit? A comment on theory testing. *Psychological Review, 107*, 358-367.