Problem Solving Between Action-Selection and Action-Completion in a Simple Domain

Gareth E. Miles (gareth.miles@southwales.ac.uk)
School of Psychology, University of South Wales, Trefforest, CF37 1DL, UK

Stephen J. Payne (s.j.payne@bath.ac.uk)
Department of Computer Science, University of Bath, Bath, BA2 7AY

Abstract

Experiment 1 demonstrates that problem solving knowledge can be applied while a move is in progress in certain Tower of London (ToL) problems. A two-stage move process is often delayed in the second stage when participants have been misled by similarity to a previous problem. We suggest this is indicative of misgivings about the chosen move caused by on-going analysis of the move that is being made. Experiment 2 swapped the stages of the two-stage process and again reported more hesitancy in the second stage when participants had been misled. We conclude that it is desirable for models of problem solving to evolve so that they can apply the same learned problem solving knowledge both before a move is selected and while the move is being made. We then describe a model of ToL problem solving that fulfills these criteria and has been computationally-implemented within an embodied cognitive architecture.

Keywords: Problem Solving, Embodied Cognition, Production Systems

Introduction

The current paper tries to distinguish between two accounts of how a move (or action) is selected and executed during solving a problem. The first account holds that knowledge about how to solve a problem is used to decide the move and then this move is simply executed. A description of the problem acts as input to the problem solving process (here we term this a *situation-only* decision process). On this basis an action is selected and then executed. The second account suggests that problem solving occurs after the move has been selected as well as beforehand. In this account problem solving also occurs after an action is selected, immediately prior to and during the execution of said action. The input to this problem solving processes is not only a description of the problem but also a description of the intended action (we term this a *situation-action* decision process).

Problem solving after a move has been selected is rare in current psychological theories, for example this does not typically occur in ACT-r models (see Anderson, 2007). While theories of problem solving based solely on *situation-only* problem solving have the advantage of simplicity, they lack the power to use existing problem solving knowledge to evaluate ongoing or imminent actions. Theories featuring *situation-action* problem solving must necessarily feature some *situation-only* problem solving in order to derive an action for consideration. However, this *situation-only* problem solving does not necessarily have to be complex – indeed an algorithm that simply picks an action at random that hasn't been considered before might be sufficient. Some recent theories, particularly those exploring embodied problem solving, favour a combination of simple situation-only problem solving prior to action selection followed by more complex *situation-action* problem solving afterward (e.g. Miles, 2009, 2011; Schuboltz, 2007), the latter is often based on the mental simulation of the results of the action.

The differences between the two accounts are important because each implies a different form for knowledge about solving problems. *Situation-action* problem solving representations potentially could replace much of the need for *situation-only* problem solving representations. A key element of *situation-action* problem solving knowledge is that it typically either encourages or discourages an already selected action. By contrast *situation-only* problem solving knowledge is concerned with suggesting an action. Once *situation-action* problem solving knowledge is added to a theory of problem solving then there is less emphasis on the need to select the correct action first. An unsuitable action can be selected then rejected using *situation-action* knowledge. Indeed, a situation-action account suggests that people often cycle through a series of possible actions at any given stage of solving a problem, allowing *situation-action* knowledge to confirm or deny each action.

Hence the two accounts of problem solving are fundamentally different. The traditional model suggests a single decision point, followed by action. The *situation-action* position suggests a series of tentative decisions regarding possible actions followed by evidence gathering while that action is held in mind. It is notable that the second position is much more temporally scalable than the first. For example it is awkward to model speed-accuracy-tradeoffs in the first style of problem solving, while the ability to vary the amount of time spent gathering evidence is implicit in the second proposal.

Empirical Support for Situation-Action Knowledge

Currently there is only indirect empirical support for the existence of *situation-action* knowledge. A necessary pre-condition of this knowledge is the ability to represent an action without necessarily executing that action.

Neuroscience has provided evidence of a representational role for parts of the brain associated with action, for example the premotor cortex (e.g. Decety et al., 1994). The existence of mirror neurons that are activated both by performing an action and observing an action (Gallese, Fadiga, Fogasse, & Rizzolatti, 1996) suggests that the motor areas of the brain are involved in thinking as well as doing. Over the last 15 years a large body of work has pointed to the conclusion that motor areas of the brain are used for representational roles as well as for executing actions (see Barsalou, 2009, p. 1285, for a brief review).

If representations in the motor areas of the brain are available to other areas of the brain, then logically these other areas will be able to make use of these representations when deciding what to do. It is exactly this logic that supports the existence of *situation-action* knowledge. Simply the problem solving parts of the brain are aware of the situation, they are also aware of the action that has been represented in the premotor cortex and related areas. It makes sense for the problem solving areas of the brain to make use of this knowledge regarding the conjunction of situation and action to either spur the motor areas into that action or pull them back from the brink of making an error.

The paper begins by reporting two Experiments that looked for evidence of continuing problem solving in the final stages of move selection in the Tower of London (variant) problem. We then show how these data can be computationally modeled using *situation-action* knowledge.

Experiment 1

The paradigm used in both Experiments reported here works by biasing the selection of a first move in a given Tower of London problem. The bias occurs because participants have earlier solved a problem that is either superficially similar or the same as the target problem. In the repeat condition the bias supports the correct first move (of two possible moves), while in the false-analogy condition the bias supports the incorrect first move (again of two possible moves). These conditions are contrasted with problems where there is no bias.

Method

Design: The Experiment was presented in two blocks, firstly of 3-disk problems then of 4-disk problems. Each block featured two training problems, a one minute pause, then two target problems. Each target problem had an inverse version, which although superficially similar has a different optimal first move. In each block one of the target problems was the same as one of the training problems (*repeat* condition) and one target problem was the inverse of the other training problem (*false-analogy* condition). Comparisons were planned between these conditions and the unbiased performance on the final training problem (*novel* condition).

Participants: Twenty-eight undergraduates participated in Experiment 1, each received either 30 minutes course credit or £2.

Materials and apparatus: The ToL problems were presented on a desktop computer. Participants responded using a mouse. To move a disk the participants had to click on the disk they wanted to move and then click on the peg they wished to move it to. At the top of the screen the goal state was shown while the current state was interacted with in the main area of the screen. Disks were shown in different colours and each was labeled with a different letter.

Procedure: At the beginning of each of the two blocks participants were first presented with an orientation task. The orientation task required six moves from a flat start state, with no goal displayed. Instructions were then displayed on the screen for a minimum of 30 seconds.

For 10 seconds prior to all training and target problems, the goal state for the problem was presented, on top of, and obscuring, the initial configuration for the problem. During this period a miniature representation of the start state was shown near the top of the screen, but all the disks in this representation were blocked grey, preventing participants from beginning a solution to the problem. This part of the procedure was designed to act as a cue to the related training problem. This display was then removed revealing the interface.

During the training phase problems the participant was only allowed the number of moves in the optimal solution to a problem. Once they had made this number of moves (and the goal state was not reached) a panel appeared (for 3 seconds) obscuring the problem, with "Try Again!" displayed in large letters. The problem was then reset to its original start state. This restriction was critical in ensuring all participants learned the same correct solution for each training problem (each problem had only one optimal solution path).

Timed lockouts were used between problems and between blocks. Participants were locked out for 30 seconds between all consecutive problems. The pause between training and target phases was one minute. There was also a minute lockout between blocks.

Prior to each target problem in a block a hint was given during the last 15 seconds of the lockout time. In the 3-disk block it was phrased as follows "The next problem will be the same as one you have already done." while in the 4-disk block it was "Note: You will have already solved the next problem". This change of phrasing was designed to increase the salience of the hint in the 4-disk block.

Results

All latency data were log transformed for analysis; the raw data are summarised in Table 1. Comparisons were made between i) false-analogy condition and the repeat condition, ii) false-analogy condition and the final training problem (novel condition). The later comparison is subject to order effects, but the order effects (one would expect improved performance with practice) run counter to the predicted effects of condition (false-analogy < novel).

The 3-disk problems

No significant effects of condition were found in measures of 3-disk problem solving. Most participants were still learning the basic methods needed to solve the ToL during this block, and this may have disrupted performance on the experimental conditions.

Table 1: Number of optimal first moves (from 28), first click latency (secs), and second click latency (secs) in Experiment 1

Condition	No. of Optimal 1st moves	1st Click latency (SD)	2nd Click Latency (SD)
Repeat	28	5.18(1.66)	1.33(1.57)
Novel	26	6.22(1.66)	1.39(1.54)
False Analog.	12	7.35(1.83)	1.94(1.84)

The 4-disk problems

Significantly more optimal moves were made under the repeat condition than the false-analogy condition, 28/28 vs. 12/28, $p < .001$. A within-participant T-test found no significant differences between the false-analogy and repeat conditions on measures of the time taken to initiate the first move ($t < 1$). However participants took less time to complete a move in the repeat condition than they did in the false-analogy condition, $t(27) = 5.86$, $p < .001$.

In the novel condition participants succeeding 26 times on 28 first attempts at the final training problem, this compares to 12 times from 28 attempts of false-analogy target problems, $p < .001$. There was no significant difference between the novel condition and the false-analogy condition selection on time taken to initiate the first move by clicking a disk ($t < 1$) but it took longer for participants to click on the location the disk was to go to in the false-analogy condition, $t(27) = 2.99$, $p < .01$.

Of the 28 participants, two made an error in the 'novel' condition. The remaining 26 were split into those that made an error on the subsequent false-analogy condition and those that did not; groups error (N=14) and correct (N=12). These data were analysed in 2 x 2 mixed design ANOVA on disk destination click latency, with condition (false-analogy Vs. novel) as a within participant factor and error group (error Vs. correct) as a between participant factor. There were no interactions with, or main effects of group (all $F < 1$). This analysis suggests that hesitancy over the move being made was present both in those who did make the correct move those who didn't.

These results are consistent with the hypothesis that participants were engaged in problem solving during the final stages of completing the move. The hesitancy seen in the final stages of the move in the false-analogy condition is best explained as second thoughts about a move that has previously been decided upon. Despite these second thoughts, the original move is at least sometimes completed, but sometimes an alternative move is chosen (as indicated by the lack of differences in hesitancy between those who made errors and those that made the correct move in the false-analogy condition). This suggests that problem solving knowledge is being used after the first move has been decided upon and initiated.

Our theoretical account assumes that the participants have decided on the move they want to make prior to clicking the disk. Certainly the relative distribution of latency between first click and second click supports this idea. However to demonstrate that the move has been decided upon prior to the first click, Experiment 2 reversed the order of actions needed to make the move, with the destination selected first and the disk selected second. In the second stage of the move only one disk (the top disk) could be selected, the decision about where to move it having already been made.

Experiment 2

As well as changing the order in which the actions needed to complete a move were carried out, Experiment 2 attempted to improve on several elements of the design of Experiment 1. Crucially, only the comparison between the novel condition and the false-analogy condition was explored. It was felt that this comparison best captured the impact of the false-analogy manipulation.

Though the novel condition replaced the repeat condition, Experiment 2 used the same basic design as Experiment 1, with the exception that the two Experimental blocks now used 4-disk and 5-disk problems. Prior to this, participants completed a training block of 3-disk problems that facilitated the learning of the main principles of solving Tower of London problems. While the order of actions needed to move a disk were changed, other aspects of the interface remained unchanged.

Method

Participants: Sixty-four undergraduates took part in the Experiment, each received 30 minutes credit toward their course requirement.

Apparatus: The apparatus and software was the same as it had been in Experiment 1. In all stages of the Experiment the method for moving the disks was altered. Now the participant had to click on the location they wanted the disk to go to. When this was done the peg they had pointed to was highlighted (turned from black to yellow). At this stage the participant then clicked on the disk they wanted to move to this peg. If the next click was not on a disk that could be legally moved to the highlighted peg then the highlighting on the chosen peg was removed, thus allowing participants to change their mind on the desired move.

A set of 5-disk problems was introduced for Experiment 2. It was reasoned that these would be sensitive to our Experimental manipulations in the same way as the 4-disk problems were in Experiment 1 (and the 'too simple' 3-disk problems were not). Each problem again had an inverse counterpart that was used in the false-analogy condition. The use of different problems was balanced across the Experimental conditions for both 4-disk and 5-disk problems.

Procedure: Many elements of the procedure were the same as they were in Experiment 1, though the block structure of the experiment was altered. Initially participants completed a block of four 3-disk problems. Following this block the two Experimental blocks were presented. The first used 4-disk problems in the same basic structure as was used in Experiment 1 (orientation task – two training problems – pause – orientation task – target problems). In each Experimental block one of the target problems was Novel and one a false-analogy to a training problem (order of conditions was counterbalanced). A 5-disk block followed the 4-disk block, using the same structure.

Table 2: Number of optimal first moves (from 128), first click latency (secs), and second click latency (secs) in Experiment 2

Condition	No. of Optimal 1st moves	1st Click latency (SD)	2nd Click Latency (SD)
Novel	104	5.35(1.71)	1.17(1.66)
False analog.	86	4.73(1.77)	1.34(1.85)

Results and Discussion

Descriptive data for Experiment 2 are shown in Table 2. We combined data from the two Experimental blocks with latency data log transformed. There were significantly fewer correct first moves in the false-analogy condition in comparison to the novel condition (proportionally .67 vs. .81 respectively), $p < .05$. Comparisons on latency measures were made using a data set reduced by two, as two of the data points in the 5-disk block showed zero values for first-click latency (126 paired comparisons remained). This was due to participants clicking prior to the interface becoming active causing a zero to be recorded for first click latency. The effects on the counter-balancing of the Experiment were thought to be minimal. The expected simple effect, i.e. false-analogy slower, was found in the second stage latency, i.e. disk-selection, $t(125) = 2.00$, $p < .05$. There was no significant difference in the time taken to initiate the move.

The data confirm that problem solving knowledge is being applied after a move has been decided upon in the Tower of London. We argue in the next section that these data and those from Experiment 1 are best accommodated by a cognitive architecture that primarily uses *situation-action* knowledge to solve problems.

Modeling Problem Solving Following Action Selection

Problem solving knowledge has often been modeled in production system architectures, a tradition with its origins in Newell & Simon's (1972) seminal book Human Problem Solving. Recently the ACT-r cognitive architecture has been used to produce production system accounts of problem solving. In traditional problem solving accounts, *situation-only* knowledge is represented in the following format: IF situation THEN action.

The model presented (TOL-GLAM) here is coded in the Glamorgan Problem Solver (GLAM-PS) architecture. This is notable because it doesn't use amodal representation and doesn't have a dedicated mechanism for processing goals (see Miles, 2011). TOL-GLAM is thus an example of an embodied account of problem solving in the Tower of London, with emphasis placed on representation in the motor and perceptual systems used to complete the task. In terms of the representation of knowledge about solving the ToL, much of what TOL-GLAM knows is stored in the format: IF situation AND action THEN inhibit/activate action. This knowledge verifies the appropriateness of an already selected action, rather than specifying what action should be taken in a particular situation.

The TOL-GLAM Model

In the GLAM-PS architecture there are modules dedicated to visual perception, ocular movement and motor actions. There are also modules dealing with other functions, for example auditory perception, speech production and bodily movement. Each module has its own production memory, working memory and production matching bottleneck.

Executive control within a GLAM-PS model emerges from the interaction of distributed subsystems (a similar idea was explored by Barnard, 1991). This control is based on each module's ability to see what is happening in all the other modules. So a production in the motor action module can match to working memory representations in other modules as well as working memory representations in the motor module itself.

Situation-Only Problem Solving in TOL-GLAM

There are examples of *situation-only* and *situation-action* problem solving knowledge in TOL-GLAM. In the two Experiments the first move is restricted to two possibilities. The disk that is to be moved is always known (as it is on top of all the other disks in a tower configuration), the only question is the disks destination.

The *situation-only* algorithm used by TOL-GLAM begins by generating an action plan for moving a disk. This action plan is represented in the motor module, within a hierarchical structure. An example is given below (with only key attributes shown):

Action_plan1

Type	*Action_plan*
First_element	*disk_click1*
Last_element	*destination_click1*

Disk_click1

Type	*click_on_object*
Location	*diskA_location*
Super_element	*Action_plan1*
Previous_element	*none*
Next_element	*destination_click1*

Destination_click1

Type	*click_on_object*
Location	*Peg2_location*
Super_element	*Action_plan1*
Previous_element	*Disk_click1*
Next_element	*none*

This initial action plan will often not involve the movement of the top disk. Typically TOL-GLAM, will first represent the movement of the bottom disk in the tower to its goal location. This reflects a means-ends analysis of the ToL problem where the bottom disk is prioritized as the biggest difference between the start state and goal state. While the GLAM-PS architecture doesn't feature explicit goal representation, what is happening is that TOL-GLAM is effectively 'subgoaling' the bottom disk. The model then represents a move of the top disk (the first blocking disk) to the peg where the 'subgoaled' disk is not going (in order to remove the block).

An exception to this process occurs when TOL-GLAM recalls a previous problem that was apparently the same as the current problem. In this case the first move that was made successfully in the previous problem is used to determine where the top disk should go.

TOL-GLAM will now have a representation of a potential first move in its motor module. It is at this point that *situation-action* knowledge is used to determine the appropriateness of the action, either increasing its activation till it is executed, or blocking its execution.

Situation-Action Problem Solving in TOL-GLAM

The *situation-action* algorithm used in TOL-GLAM is based upon forward search, and makes use of representational simulation of the results of the move that is being considered.

The process begins after a potential move of the top disk has been represented in the motor module. At this point productions in the visual module are able to match to this motor module representation and simulate the result of this action. In Miles (2011) visually simulated interim stages were utilized in a model of offline algebra problem solving, what the TOL-GLAM model does is very similar, essentially looking ahead to see what the results of the action that is being considered will be.

Simulation of the results of the move involves the creation of a projected representation of the disk being moved in its new location, and the inhibition of the representation of the disk in its current location. Once the move has been simulated in the visual module, the motor module is now able to consider the next move that will be made after the current one. The productions that do this match both to simulated and actual visual representations.

At this stage an action plan will be generated for the second move and typically, any conflict with the first move will often become apparent to TOL-GLAM. This is particularly the case if the first move blocks the ideal second move. A production looks for incompatibilities between the two moves being considered. On the other hand if the first move doesn't block the ideal second move then it is taken as providing evidence that the first move is a good one.

Executing an Action in TOL-GLAM

Key features of the GLAM-PS architecture determine the process of action execution in TOL-GLAM. One of these is the Action-Execution Threshold (AET), a level of activation that must be reached before an action or action plan will be executed. The AET is an important element of GLAM-PS because it allows actions to be represented without necessarily being executed (Miles, 2009). Within the TOL-GLAM model it allows a move to be represented and then evidence gathered about the suitability of the move. There is no limit in GLAM-PS to the number of productions that can match if those productions change an existing representations activation level. This means that in TOL-GLAM the representation of a move can be simultaneously inhibited and activated by competing productions. It is the relative strength of the competing productions that will determine whether the action representation will continue to increase in activation until it surpasses the AET, or be inhibited.

Simulating the Results of Experiment 1 and 2 in TOL-GLAM

To simulate the results of Experiments 1 and 2 TOL-GLAM was setup with productions that represented knowledge gained from previous training problems, which were added to the productions that model normal problem solving in the ToL.

The additional productions, modeling knowledge from specific previous problems, trigger when TOL-GLAM is faced with a problem that has the same initial configuration and goal configuration as the previous problem in terms of number of disks on each peg in each configuration (so an exact match wasn't necessary). The identity of the first disk to be moved must also match. These encodings of solutions from previous problems result in the first move used in the previous problem first being represented in the motor module and subsequently quickly gaining activation.

The performance of the model was tested on simulation runs of the 84 problems from which data were taken for Experiment 1 (28 each in the repeat, novel and false-analogy conditions) and the 256 problems from Experiment 2 (128 novel, 128 false-analogy).

In GLAM-PS each production has a strength value, this strength modifies the impact of the production – so a strong production will increase the activation of an action representation more than a weak one. The strength of the productions modeling knowledge from the previous problem was systematically varied through a single parameter beta, which multiplied the strength of productions activating a representation of the action used previously. The beta values used conformed to a Gaussian distribution, meaning that in some case the influence of previous problems was strong, but in others the influence was weaker. A second parameter theta modified the strength of productions that solved the problems, this was a free parameter with a single value across all simulations runs used to maximize the match between model and data.

Another free parameter was AET, the level of activation at which an action was executed in the motor module. Additional parameters included the time taken to initiate problem solving and the time taken to execute a mouse click.

A comparison of the models performance to the latency and accuracy data showed relatively strong fits, $\chi^2 = .72$ and $\chi^2 = .83$ respectively. The matches of model to data do show some differences with the model making fewer errors than participants did in Experiment 1, but more errors than seen in Experiment 2. The timing of the mouse clicks was simulated more closely, partially reflecting the availability of parameters that varied the timing of the models performance.

A key purpose of the model was to simulate the differences in latency seen in the false-analogy condition (as compared to Repeat and Novel problems). In this respect the model is very successful, showing the same differences as participants.

General Discussion

Experiments 1 and 2 provide evidence of problem solving occurring after move-selection in the Tower of London (ToL). The TOL-GLAM model accounts for this data through a mechanism based around *situation-action* knowledge. This knowledge is encoded in TOL-GLAM as production rules that increase or decrease the activation of a particular motor action, depending on the apparent suitability of this action. The delays seen in move completion during the false-analogy condition in Experiment 1 and 2 are explained as TOL-GLAM having 'second thoughts' about the suitability of an already selected move. Our findings are similar to those of Walsh and Anderson (2009) who demonstrated how participants adaptively 'changed their minds' about the best strategy to solve a multiplication problem after a quick initial choice.

The way problem solving knowledge is structured in TOL-GLAM is noteworthy, relatively simple *situation-only* productions suggest a possible action, while more complex *situation-action* productions contain much of the knowledge that TOL-GLAM possess of how to solve ToL problems.

The notion that actions are selected and then evaluated is found in other theories. However in most existing theories this evaluation occurs in a single cycle of the system, and is only necessary if there is a conflict between two or more possible actions. For example in SOAR (Newell, 1990) preference rules are used for conflict resolution, while in ACT-r (Anderson, 2007) the relative utility of actions is considered. In TOL-GLAM, and more generally in the GLAM-PS architecture, the evaluation of an action is a protracted process, typically involving the evaluation of a single action, rather than multiple competing actions.

Although the current research focuses on a simple knowledge-lean domain (ToL), the issue explored is fundamental to understanding human thought. Current accounts (e.g. ACT-r, SOAR) appear to suggest that we think about situations, reason about them and only then select an action. The suggestion here is that much of human thought begins with a possible action, and is followed by reasoning about the suitability of this action for the current situation.

References

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* New York: Oxford University Press.

Barnard, P. J., & Teasdale, J. D. (1991). Interacting cognitive subsystems: A systematic approach to cognitive-affective interaction and change. *Cognition and Emotion, 5*, 1-39.

Barsalou, L. W. (2009). Simulation, situated conceptualization, and prediction. *Philosophical Transactions of the Royal Society B, 364*, 1281-1289.

Decety, J., Perani, D., Jeannerod, M., Bettinardi, V., Tadary, B., Woods, R., Mazziotta, J. C., & Fazio, F. (1994). Mapping motor representations with PET. *Nature, 371,* 600-602.

Gallesse, V., Fadiga, L., Fogassi, L., & Rizzolatti, G. (1996). Action recognition in the premotor cortex. *Brain, 119*, 593-609.

Miles, G. E. (2009). Representing goals modally: A production system model of problem solving in the Tower of London. *Proceedings of the Thirty-First Annual Conference of the Cognitive Science Society*, 469-475. Mahwah, NJ: LEA

Miles, G. E. (2011). 'Look no goals!': A sufficient model of simple algebra problem solving without explicit goal representation. *Proceedings of the Thirty-Third Annual Conference of the Cognitive Science Society*, 2691-2696. Mahwah, NJ: LEA

Newell, A. (1990). *Unified theories of cognition.* Boston: Harvard University Press.

Newell, A., & Simon, H. A. (1972*). Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.

Schuboltz, R. I. (2007). Prediction of external events with our motor system: towards a new framework. Trends in *Cognitive Sciences, 11*, 211-218.

Walsh, M. M., & Anderson, J. R. (2009). The strategic nature of changing your mind. *Cognitive Psychology, 58*. 416-440.