Cognitive Modelling for Psychology Students: The Evaluation of a Pragmatic Approach to Computer Programming for Non-Programmers

Trevor D. Collins
Institute of Educational Technology
The Open University, Milton Keynes, UK
e-mail: T.D.Collins@open.ac.uk
phone: +44 1908 655731

Pat Fung
Institute of Educational Technology
The Open University, Milton Keynes, UK
e-mail: P.Fung@open.ac.uk
phone: +44 1908 652418

Abstract

Cognitive modelling is a common component of undergraduate psychology degree courses. However, the problem with cognitive modelling is that students must first learn how to program a computer before they are able to practice cognitive modelling and thereby, appreciate for themselves the contribution cognitive modelling makes to psychology. Learning to program a computer is a non-trivial task, particularly for people from non-technical backgrounds. Overcoming the steep learning curve associated with learning traditional programming languages, such as Prolog or Lisp, is a difficult problem currently facing psychology education. “Hank” is a visual programming language that has been proposed as a solution to the programming overhead associated with teaching cognitive modelling. This paper discusses a programming walkthrough evaluation of the supporting material and modelling exercises used by The Open University for teaching cognitive modelling with Hank. This work is part of an ongoing evaluation project of Hank involving further theoretical and empirical studies. The key issues raised in this paper are; the consistent use of appropriate terminology, and the sufficiency, level and pace of the explanations provided.

1 Introduction

Cognitive modelling is an important aspect of cognitive psychology. Building computer-based models of cognitive theories requires the model-builder to specify every process in detail, thereby enforcing a strong test on the adequacy of the theory. Cognitive scientists typically use Prolog or Lisp to build cognitive models [Eysenck and Keane, 1995]. Although some cognitive modelling courses have attempted to give psychology students first hand experience of cognitive modelling (see [Ford, 1987] and [Scott and Nicholson, 1991]), learning to program a computer using Prolog or Lisp is a particularly difficult task (see [Fung et al., 1990] and [Taylor, 1990]). As a result, many cognitive psychology courses only consider the underlying theory and historical developments in cognitive modelling. However, the perceived pedagogical benefits of practical cognitive modelling are:

1. the model-builder has to study the theory in detail in order to build an accurate model,
2. the practical experience of building an accurate model serves to identify and correct the model-builder’s misconceptions of the theory, as well as reinforce their understanding, and
3. once built, the model can be used to investigate the failings and limitations of the theory.

In order to give psychology students an insight into cognitive modelling, without first having to become skilled Prolog or Lisp programmers, a visual programming language called “Hank” has been developed by Paul Mulholland and Stuart Watt at The Open University’s Knowledge Media Institute (see
http://kmi.open.ac.uk/projects/hank/). As reported in [Mulholland and Watt, 1998], the design rationale behind the Hank programming language has drawn heavily from the areas of end-user programming, programming by demonstration and software visualization.

The Programming Walkthrough method described in [Lewis and Rieman, 1993], and the Cognitive Dimensions method described in [Green and Blackwell, 1998] are being used to evaluate Hank. As well as an empirical evaluation comprising of observational studies, and interviews with the course tutors and students, this paper presents a programming walkthrough evaluation [Lewis and Rieman, 1993] of the supporting materials used to introduce psychology students to cognitive modelling with Hank.

This work forms part of an evaluation project to verify the effectiveness of Hank with respect to its initial set of design objectives, validate Hank’s design rationale, and investigate the pedagogical benefits of cognitive modeling for undergraduate psychology students. Section 2 presents the design objectives of Hank, a description of the language, and an overview of the computer-based modeling environment. An overview of the programming walkthrough evaluation is given in Section 3, with a description of the methodology used and the resulting outcomes. Section 4 concludes this paper with a summary of the issues raised by the programming walkthrough evaluations and their educational implications for psychology.

2 Hank: A Cognitive Modelling Language for Psychology Students

“Hank” is a visual programming language designed for psychology students [Mulholland and Watt, 1998]. Specifically (in the first instance), for psychology students of The Open University, a distance-learning university based in the United Kingdom. Five key objectives (some specific to distant learning) were set prior to the design of Hank; the language should be specifically aimed at psychology students, usable by non-programmers, usable in groups, clearly visualize process, and be usable on paper.

The design rationale behind Hank is described in [Mulholland and Watt, 1998]. A brief overview of the main constructs of the language, along with a description of the computer-based environment are given here.

2.1 The Hank Language

Hank has three main programming constructs: “fact cards,” “questions” and “instruction cards.” Fact cards adopt a tabular structure for representing factual knowledge, instruction cards represent procedural knowledge, and questions are used to query fact cards and instruction cards.

Fact cards are made up of a title on the top row, column label(s) referring to components on the second row, and data row(s) representing individual facts in the remaining rows (see Figure 1, left).

Questions are used to query information by looking for matching fact cards, or instruction cards, in the Hank database. Questions are made up of a title on the top row, column label(s) on the second row, and a single data row on the third row (see Figure 1, right). Variables, delimited by question marks (e.g. ?day?), can be used within the data row of a question. When a match is found for a query the result is reported as the “Status” of the system - successful matches report a status of “OK,” unsuccessful matches report a status of “Fail.” In addition to the system’s status, the value of any instantiated variables is also reported. The example question
shown on the right in Figure 1 returns \( ?\text{day} = \text{Monday} \) and \( \text{Status} = \text{OK} \).

*citation*

*Instruction cards* are used in Hank to represent procedures (see Figure 2 for an example). The top section of an instruction card is referred to as the instruction card’s “wish box.” The wish box is made up of three rows; the title on the first row, the component column label(s) on the second row, and the variables or values used to pass information into and out of the procedure on the third row. The lower section of the instruction card is referred to as the “process box.” The process box describes the process involved in carrying out the procedure using a set of questions connected together using OK and Fail links. According to the status of each question, the system follows either an OK or Fail link. If no applicable link is found after a question has been answered, the system (i.e., question processor) halts and returns the value of any variables used in the original question as well as the final status.

By drawing on the use of simple visual representations, such as tables to represent data in fact cards and directed links to illustrate the control flow of the program within instruction cards, the language itself is intended to be easier to use for psychology students than more formal textual languages such as Prolog or Lisp.

### 2.2 The Computer Environment

The above three programming constructs can be used on paper to write paper-based cognitive models - this is particularly useful for Open University cognitive psychology students working individually, at home and without access to a computer. However, the students also attend a one week residential school to complete additional group-based practical work that they are unable to do at home. During their residential school, students work in groups of 4 (approx.) to complete two practical projects which they select from four areas of the course: memory, language, problem solving, or cognitive modelling. In the cognitive modelling project, the computer-based version of Hank is available for the students to build more complex models than they were able to tackle individually using the language on paper.

The computer environment for Hank includes a graphical environment in which the user (i.e., the students) can write Hank programs, an automated question processor to run their programs, and a bidirectional tracer for illustrating the program’s execution. Three windows; the “Program Window,” “Control Panel” and “Workspace,” are available under the “Windows” menu for switching between each of these three tasks. Figure 3 shows a screen view containing an example of each window.

### 3 Evaluating Hank

This paper describes part of an attempt to evaluate Hank, validate the adopted design rationale, and investigate the pedagogical benefits of cognitive modelling for psychology students. The methodology described in this section refers to the programming walkthrough evaluation. The other methods being applied in the Hank evaluation project include the Cognitive Dimensions method described in [Green and Blackwell, 1998], and a number of empirical evaluations comprising of observational studies and interviews with the course tutors and students. The *cognitive dimensions* framework is a broad brush evaluation technique in which a number of dimensions are identified to investigate the cognitive load associated with a specific task. The *empirical studies*
are centered around the students’ cognitive modelling projects.

3.1 The Programming Walkthrough Method

The *programming walkthrough* evaluation method is a task centered approach used to predict user difficulties. The complete task to be carried out by the user is analysed step by step, in an attempt to identify any possible sources of error in the interface and the supporting materials, such as ambiguities, inconsistencies, and insufficient or poorly structured guidance. Following the approach reported in [Lewis et al., 1998], a programming walkthrough analysis was carried-out to evaluate the introductory course material, self-assessment exercises, and assignments of The Open University’s introductory cognitive modelling project.

The programming walkthrough method [Lewis and Riemer, 1993] has also been applied to examine the sorts of models the students will be expected to produce at residential school using the computer-based modelling environment. At residential school students can attempt to model one of four theories; semantic primitives, conceptual hierarchies, mean-ends problem solving, and production systems. By investigating the models the students would be expected to produce, a number of suggestions have been made as to the type of guiding knowledge the students should be supplied with by the residential school tutors.

3.2 The Outcomes of the Programming Walkthrough

The outcomes of the programming walkthrough evaluation relate to the development of the introductory course materials produced for the students’ introductory modelling project and the models that may be generated during the residential school’s modelling project. These findings, along with some early empirical support, are summarized in this subsection. The empirical results of the introductory modelling project and the results of the cognitive dimensions evaluation will be presented at the conference, along with a summary and preliminary analysis of the data collected on the modelling projects carried-out at the residential school.

3.2.1 Introductory Course Material

A draft version of The Open University’s supporting materials on cognitive modelling was evaluated using the programming walkthrough approach described in [Lewis et al., 1998]. The following points were made by the
programmed walkthrough:

- **Initial Programming Support** - When first asked to build a model what items should the students be supplied with? A blank page or a blank fact card? If blank fact cards are supplied how many columns should they contain?

- **Layout of Examples** - The example questions given in the course material gave the Hank representation followed by the question in English, followed by the response given in Hank. It was proposed that the question should first be written in English, then the Hank representation and response, followed by an explanation of the mapping from English into Hank notation and the meaning of, and reasoning for, the given response.

- **Self-Assessment Questions** - The supporting materials include self-assessment exercises. It was recommended that the answers given to the assessment questions and their accompanying explanations should include a discussion of the alternative answers that the students could be expected to produce.

- **Consistent Terminology** - The terminology used to introduce Hank was chosen to be non-technical, in order to match the expected background knowledge of its intended users (i.e. psychology undergraduates). However, some of the terminology was used inconsistently. For example; the terms “wish box” and “ask box” were both used to refer to an instruction card’s wish box.

- **Sufficient Explanations** - Some concerns were raised over the sufficiency of the explanations given for computer programming concepts. The introduction of variables within the supporting materials was deliberately made at a non-technical level, however concerns were raised over the sufficiency of the explanation given. The matching process used in the introductory exercises only uses the first possible match found for a query, the consequences of multiple matches within a fact card are not addressed in the materials, again some concern was raised over the sufficiency of the given explanation.

- **Explicit Exercise Questions** - It was suggested that the student assignments should more closely follow the tasks described in the corresponding sections of the course materials. As part of the cognitive modelling assignment, the students are required to build a model of schema theory that illustrates schema based retrieval processes. However, the students were not explicitly told to build an instruction card to apply a schema.

- **Wording** - It was felt that the wording used in the assignment questions could be keyed better to the examples given in the course material, in order to help identify which piece of schema representation is required. For example; rather than referring to the schema values as “slots” they could be identified as specific or default values.

- **Nested Instruction Cards** - The schema model requires the students to query an instruction card within another instruction card. Although the students are told that the questions used in the process box of an instruction card can query both fact cards and instruction cards, they are not given any examples in the course material. It was suggested that the students be given an example to illustrate nested processes.

- **Student Explanations** - Some of the student assignment questions required the students to provide the answers for a series of Hank questions. It was felt that in addition to answering the questions, the students should also provide an explanation of their answer. Otherwise, in some cases the students could guess the answer without fully understanding the program involved. For example; when asked to identify the status returned for a given question.

3.2.2 **Cognitive Modelling Projects**

Several models have been developed for the four suggested theories that the students are expected to attempt at residential school. The programming walkthrough evaluation of these models identified the complexity associated with each step of the modelling task. The findings have been particularly useful for identifying the anticipated problem areas, guidance required and potential learning outcome of the cognitive modelling exercise. Although these are specific to each theory and each modelling approach, the following generic points were raised:
• **Additional Hank Primitives** - Additional primitives to the simple "Ask question" introduced in the course material are available in Hank's computer environment. For example: "Ask All questions" ask for all the solutions to a query i.e. not just the first match, "Say questions" ask Hank to write a text message, "Same questions" compare two or more values, and "Add to Fact Card questions" add and remove data rows in fact cards. Some of these primitives are used in the residential school projects. The residential school tutors must ensure that a sufficient set of additional primitives are introduced to the students.

• **Level of Computer Feedback** - The cause of failure within Hank is not reported. Hank may fail to find a matching fact card or instruction card to a query, or it may fail to find a specific data row within an fact card. The user is not given any feedback on the cause of failure. For the purposes of debugging a program it was felt that giving the cause of failure would be beneficial to the user.

• **Guiding Knowledge** - It was proposed that the students be provided with a detailed description of the stages involved in building their model. For example, in the case of means-ends problem solving; the initial situation, goal situation, and constraints of the problem must be specified along with a complete set of anticipated situations. The anticipated situations (i.e. stages) should be ordered such that the students are faced with evenly paced steps through the modelling process, that build on the students previous knowledge, rather than steep or un-even jumps in complexity for which the students would be unprepared.

### 3.2.3 Subsequent Action

From the feedback given to the course team members developing the materials for the introductory cognitive modelling project, several changes were made to the course material and its associated assignment. The decision was made to ask the students to draw their own cards rather than providing a set of blank templates. The layout of the Hank examples was changed so that questions written in English preceded the Hank questions. The text was corrected following the recommendations made regarding the self-assessment questions and terminology used in the course material. No changes were made as a result of the concerns raised with regard to the sufficiency of the description of variables or the matching process. The authors of the course materials were cautious to ensure the introductory material remained sufficiently straight-forward for distance-learning students and suggested that a more complete description of variables and the matching process could be provided for the students at residential school.

The recommendations made with regard to the assignment were also taken on board during the subsequent development of the assignment questions. The wording used in the assignment questions was reviewed to more closely match the course materials. An example of the use of a nested instruction card (a schema instruction card) was made within the course materials. The structure of the schema modelling question was matched to the structure and pace of the course materials. And, the students were asked to explain as well as present their programs’ answers.

The suggestions made as a result of examining some of the potential residential school models have been put forward to the authors of the course material. The supporting materials for the residential school tutors are currently under development and are due to be presented to the residential school tutors in June.

### 3.3 Early Empirical Support

During recent preparations for the empirical studies, a pilot study has been run involving three colleagues with previous experience of cognitive modelling with Prolog. These individuals were asked to; read through the revised supporting materials that have recently been given to the students, attempt the introductory assignment, and during a one day pilot study, work as a group with Hank to build a model similar to one that the psychology students would be expected to produce at residential school.

The feedback given with regard to the introductory cognitive modelling project was, in general, very positive. A number of favourable comments were made with regard to the layout, structure, style and pace of the course materials. As well as the visual nature of the Hank language itself and the support given by the workspace for tracing through a program’s execution.
However, serious concerns were noted over the introduction of variables: “They haven’t clearly explained what’s happening underneath really about variables, so I don’t think a student coming to it would really follow what’s happening” and “I got confused around two variable names referring to the same thing.” Although the first comment reflects a point raised in the programming walkthrough, like the programming walkthrough evaluations it may well be tainted by the participant’s previous experiences of variables. The above comments emphasise the need to further validate the students understanding of variables, however, as the authors of the course material point out - psychology students may not be concerned by the underlying operations of the computer.

As for the assignment questions; four new questions were added prior to the question previously reported in Section 3.2.1, which has also been re-written. Again the majority of the comments made were very favourable, particularly for the structure and format of the question on schema theory. Two minor points were raised; one over the grammatical structure of a question, and a second over the choice of an non-intuitive query in part of another question. These two points are important as they may well be a source of confusion for some students. The reason given by the course team for their inclusion in the assignment, is to test the students’ understanding of the representation and to ensure that the students are not simply guessing the answers.

4 Conclusion

This evaluation project has already had a positive influence on the cognitive modelling projects undertaken by undergraduate psychology students at The Open University: The findings of the programming walkthrough evaluations have resulted in changes being made to the introductory course materials (see Subsection 3.2.3), and the feedback given in the pilot study (see Subsection 3.3) indicates that these changes have improved the quality of the course materials.

Acknowledgments

This project would not be possible without the close collaboration of the Hank design team, namely; Paul Mulholland and Stuart Watt, and the other members of the cognitive psychology course team, namely; Sandy Aikenhead, Judy Green, Peter Naish and Ingrid Slack. The programming walkthroughs presented in this paper were carried out with the help of Clayton Lewis from the University of Colorado, during a 3 month visit at The Open University, the authors are particularly grateful to Clayton for his invaluable advice and support. This project has been funded by the research committee of The Open University’s Institute of Educational Technology.

References


