

Cognitive task graphs and executable user models for better hypertext

Yin Leng Theng, Cecile Rigny, Harold Thimbleby & Matthew Jones

*School of Computing Science, Middlesex University
Bounds Green Road, London N11 2NQ
Tel: +44 181 362 5000
Email: {y.theng, c.rigny, h.thimbleby, m.jones}@mdx.ac.uk*

Abstract

Authoring hypertext systems is complex and difficult. To help designers build quality hypertext systems, designers should understand users' behaviour and the common tasks they perform when navigating hypertext systems, and, further, designers should build in accurate representations of user models. This is uncontentious wisdom, but so far, not easy. This paper shows that common tasks can be broken down into simpler subtasks, taking into consideration users' reasoning and learning. The results of this cognitive task-based approach can then be used to design the interface and functionality of a hypertext system. In particular we show that task trees can be automated, leading to 'executable user models.' Executable user models can be used efficiently as substitute users to support design and evaluation purposes. We discuss the task browsing as a concrete example.

Keywords

Task graph, hypertext, browsing, information search, seeking references, recall, navigation, cognitive user model

1. Introduction

Building effective and well-structured hypertext systems is a difficult and complex process — due to the richness of associative links that exist among nodes, and to the combinatorial explosion of design possibilities. Designers have to cope with a vast range of potential structures to build a hypertext system, and with a large number of choices to create links. When designers are themselves 'lost' in how to manage the complexity of the design process, producing well-structured hypertext systems becomes a matter of chance. It is, therefore, not surprising that few good hypertext systems are created. This, of course, has inadvertently generated a set of problems associated with navigation issues within hypertext systems, of which the 'lost in hyperspace' problem is

but one example. How can designers be helped in creating well-structured, user-centred hypertext systems, appropriate to the users' tasks?

2. Need to understand users and the tasks they perform

To help designers build well-structured hypertext systems, we need to know users and their needs by analysing the common tasks they want to perform or try to perform. In fact, the importance of understanding users and their tasks has never been in question, at least in the research community: ranging from Hansen's "know the user" (Hansen, 1971), to Lewis and Rieman's "task-centred design" (Lewis and Rieman, 1995). By trying to make sense of what users should do or what they actually do, designers should stand a better chance of producing user-centred hypertext systems that will meet users' needs more effectively. However, it is also well-known that designers often design for themselves unless they are trained to realise that people are diverse, and that users are unlikely to be like them (Thimbleby, 1990; Landauer, 1995). It is also imperative that designers build into hypertext systems an accurate enough representation of user models when performing these tasks, taking into consideration users' reasoning and learning processes (Newman and Lamming, 1995).

Unfortunately the conventional exhortations do not help make much impact on hypertext design problems. The hypertext design space is astronomically large (Thimbleby, 1995). Iterative design, that usually helps improve systems, has problems in that users involved in the process experience too little of a system to help make significant design contributions, and, secondly, once most hypertext systems have reached the point of user testing, commercial pressures would have them shipped and sold to uncritical customers.

In this paper, we present ways of understanding users' behaviour and navigation issues in hypertext by giving them more structure, and using this structure for improved hypertext design, thus ameliorating the usability problems more effectively.

3. Browsing in hypertext

To develop an accurate understanding of users, it is essential that representative tasks which provide reasonably complete coverage of a system's intended functionality are chosen (Lewis and Rieman, 1995). Therefore, to identify the representative tasks in hypertext, we considered the kinds of support hypertext systems generally provide: applications support (e.g., tutorial and educational needs); support for collaboration among a number of individuals and production of on-line manuals; functional support (e.g., browsing and authoring); and cognitive support (e.g., reading, annotating, collaborating and learning). Based on the kinds of support provided by hypertext systems, we identified four representative tasks users want to perform (or try to perform) when using hypertext. These tasks are browsing, information search, seeking references, and recall.

As an illustration, we have selected one out of these four: the task we call *browsing* to illustrate how cognitive task graphs can be built. (Cognitive task graphs for the other tasks can be developed similarly.) Generally, browsing refers to reading and navigating in hypertext without a definite or explicit goal for the user to accomplish. However, in this paper, we include focussed browsing, where users have an idea of what they want to do to accomplish a goal. They may explore the system's interface to discover actions useful in helping them accomplish their current goal. Based on a match between what they are trying to do and the interface's description of actions, users then select actions they think will help them accomplish their current goal. In deciding what action to do next, users will then try to understand system responses based on the action they have just performed.

In the following sections, we describe how the task *browsing* in hypertext is broken down into simpler, unit tasks using cognitive task graphs. We explain the rationale behind this task analysis approach. We then describe how cognitive task graphs are validated and refined by firstly, conducting video protocols with real users, and secondly, using computerised users in the form of executable user models. The results of our analysis are then used to make principled recommendations for the design of a prototype hypertext system.

4. Building a cognitive task graph

The concept of a task is central to user-centred design. Many task analysis (TA) techniques have been developed that focus on different aspects of tasks, such as the ease of learning a task, the knowledge users require in order to accomplish a task, or the task structure. These different aspects generally refer to the cognition, practice or logic of the task (Payne and Green, 1989). Therefore, to break the task *browsing* into unit tasks, we need to represent it in terms of how it is usually done (practice), the flow of actions (logic) and the cognitive processes (cognition).

To gather inputs for building the cognitive task graph for *browsing*, we performed the following activities: started with a rough cut of the task graph for *browsing* based on known facts about *browsing* and our experience with hypertext systems; and conducted video protocols to observe two "typical" users browsing a hypertext system. By "typical," we refer to a user who has general knowledge about navigation with hypertext systems.

Because we wanted to model the practice and logic aspects of the task *browsing*, we adapted the task decomposition (TD) technique to break down the task *browsing* into subtasks users need to have in order to perform browsing, and then describe *browsing* in terms of a flowchart of actions to achieve goals. Goals are the desired state(s) of the system and actions are the lowest level units of behaviour. Figure 1 shows the task

graph for *browsing* and the possible subtasks that are associated to it: (1) starting; (2) getting general information; (3) understanding the system; (4) navigating; (5) finding out what topics or areas are covered in the system; (6) finding a particular topic; and (7) quitting.

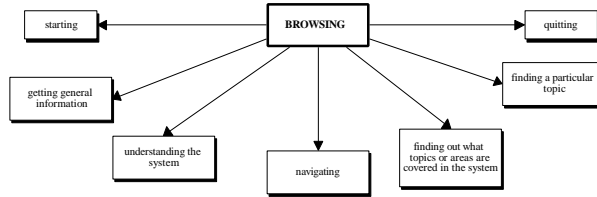


Figure 1. Task graph for browsing

Except for subtasks (4) and (5), the rest of the subtasks are goal-directed, that is, users have a goal in mind when executing them. Each subtask is then further broken down into simpler, executable subtasks or actions. As examples, we have only included subtask graphs for *navigation* (e.g., unfocussed browsing) and *finding a particular topic* (e.g., focussed browsing) as shown in Figures 2 and 3.

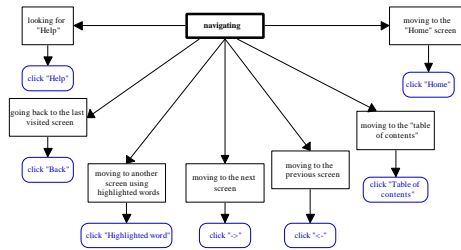


Figure 2. Task graph for navigating

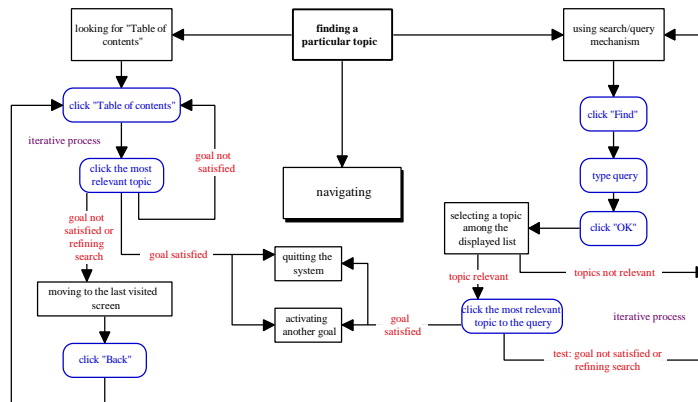


Figure 3. Task graph for finding a particular topic

Although subtasks are interrelated, the order in which they are performed depends on several parameters: users' domain knowledge of hypertext systems; users' domain knowledge of the contents covered in a particular hypertext system; users' intention or goal of using the system; and the context of users' tasks based on situated actions described in (Suchman, 1987). Unlike TD and related techniques that tend to focus on what actually happens, we also build into the task graphs subtasks that describe what *should* happen. We consider this an important inclusion since *browsing* is dynamic and therefore, hypertext systems supporting *browsing* should be adaptive as well as predictive.

Because *browsing* in hypertext is a cognitive activity, it is also essential that we take into consideration two aspects of cognition that are important to users browsing hypertext: reasoning and learning. Established TA techniques that are concerned with describing some aspects of the cognitive characteristics of users' tasks include the Model Human Processor, GOMS (Goals, Operations, Methods and Selections rules), Cognitive Complexity Theory, Task Knowledge Structures — there are many proposals. These cognitive TA techniques, however, are difficult to use and implement (Preece et al, 1994). Instead, because reasoning and learning are two important aspects of cognition that do need to be captured, we based our cognitive task graphs on Polson and Lewis' exploratory learning method (Wharton, Rieman and Polson, 1994). We made the assumption that learning is achieved when users are able to understand, induct, and subsequently, build links and concepts, as illustrated by the dotted links in the task graph for *trying out buttons options* (see Figure 4), which is one of the further subtasks associated to the subtask of *understanding the system*.

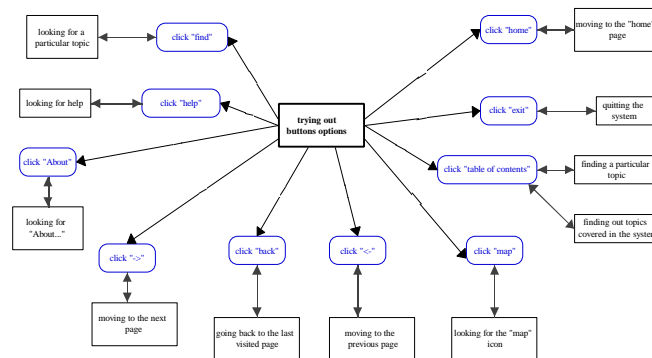


Figure 4. Task graph for trying out buttons options

5. Validating and refining cognitive task graphs

Real users need not be used only to evaluate designs, which is seen by many designers as the sole reason (Johnson, 1992), but also to generate design recommendations. From Rosson *et al.*'s study (1989), user-task analysis was among one of the several

techniques used by designers for ‘idea generation.’ Other techniques involved using external sources such as available literature, meta-strategies such as concentration, design activities such as charts and diagrams, as well as trial and error.

To validate and refine the cognitive task graphs for *browsing*, a second investigation was carried out. We recorded two typical users’ behaviour and actions using *ACM Hypertext-on-Hypertext* (ACM, 1989), a convenient hypertext system implemented in HyperCard, to complete some exercises associated with *browsing*. By conducting a video protocol of the users’ interaction, we were able to interview users after the experiment and ask them to explain step-by-step why certain decisions and actions (e.g., moving to the next screen, going to “Home” screen, moving to the “Table of Contents”) were taken while using the hypertext. We were also interested to find out the frequency with which particular procedures were carried out, the circumstances under which one procedure was used rather than another and the inputs and outputs for each procedure. Users’ feedback helped us understand their reasoning and learning processes when they chose to perform certain actions. The cognitive task graphs were then modified and refined accordingly.

It appears that this approach is very cost effective: an expert’s initial task graph, then protocols of only two users, leads easily to an improved task graph. Clearly, one might scale up this procedure for use in a production environment, but our purpose was to obtain a structure suitable for the user models to be employed in hypertext design. Whether the additional cost of obtaining better task graphs would in fact be worthwhile is a separate issue; we suspect there would be diminishing returns, especially when the delay is considered.

Section 6 presents a cost-effective tool that enables designers to automate the task graphs, thus reducing the use of extensive and time-consuming real user validating and refining them.

6. From task graphs to executable models

The task graph can be considered a program. One of us (Rigny) has developed a cognitive architecture that can emulate the user executing such graphs. CUM-DesTool (Cognitive User Model Design Tool) is a general simplified architecture to build operational cognitive user models which enable designers to understand, predict and simulate users’ behaviour and performance. The tool enables designers to model users’ tasks and knowledge in terms of concepts and relations between these concepts.

CUM-DesTool’s structure refers to ACT and the memory model proposed by Reason (1988) and is intended to enable a rapid and easy implementation of users’ models. It can handle any type of user ranging from novice to expert. The tool enables designers to model users’ tasks and knowledge in terms of concepts and relations between these

concepts. For example, the users' task trees and knowledge are represented with concepts and various links (see example described in Figure 5). The links enable the modelling of users' cognitive processes and reasoning.

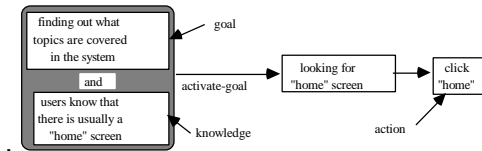


Figure 5. Description of the modelling process

A first version of the tool has been implemented in Common LISP. Crucially, using LISP allows the model to be embedded easily within other LISP programs, such as a general purpose hypertext development environment. There can be multiple instantiations of the cognitive models (randomised) and therefore we get the advantages of very fast, multiple user evaluation. The multi-module structure gives the tool some interesting properties:

1. It allows an *explicit representation* of the users' knowledge and cognitive processes which can support communication between designers and users, and therefore help designers to elicit users' needs.
2. It allows an *easy implementation* of a user model because there is a single representation structure for knowledge and tasks, and a single mechanism for reasoning.
3. *Various types of users* can be modelled by modifying few parameters. Designers are able to rapidly simulate types of potential users' behaviour from a generic user model. Such simulations are very useful to validate/test prototypes of the future system.
4. Measures of users' *cognitive workload*, *performance* and *satisfaction* can be identified and evaluated automatically.

The user's activity, e.g. when browsing hypertext systems, is for the most part goal-oriented. Figure 6 describes the cognitive processes which were incorporated into the user model. Users can identify a set of actions to achieve their goal and then select one action from this set of actions. If the initial goal is satisfied after performing for instance "action 1", a new goal can be activated, or the user can stop browsing the hypertext system. If the goal is not satisfied, users can perform another action from the initial set of actions" or can be disrupted from their initial goal and then, a new goal can be activated. The overall strategy which determines which action will be selected or whether there is disruption from the initial goal or not depends mainly on: users' general knowledge in hypertext; users' knowledge in the contents of the hypertext system they are browsing; and the context in which the user's activity is performed. The first two points determine users' profile. The third point refers to the situated actions described by Suchman (1987).

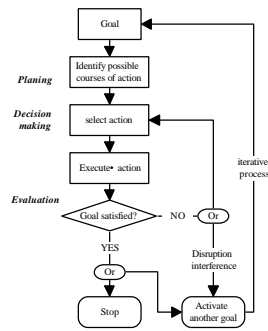


Figure 6. The cognitive model

Figure 7 shows the general interface of the executable cognitive user model for browsing provided by CUM-DesTool. The alphabetical list of concepts that have been implemented is displayed. When a goal is “set to true,” for instance the “browsing” goal, the corresponding module of the simulation of the user’s reasoning and activity is triggered. The trace of the reasoning mechanisms is stored and can be used to help designers understanding users’ behaviour.

Several measures have been identified, including: (a) the number of cognitive steps performed by the user to achieve one particular goal; (b) the time spent by the user per goal; (c) the number of documents browsed by the user per goal, (d) the number of visited hypertext nodes, (e) the number of failures, goals which have not been reached, etc. **These measures can be automatically performed by the cognitive user model.** They enable designers to correlate subjective parameters, such as ease-of-use, with interaction metrics for various types of user. They yield useful insights and raise questions to be addressed by designers.

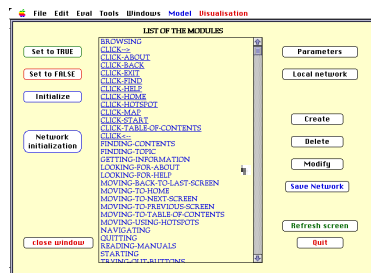


Figure 7. Example model description screenshot

7. Contributions to design

There are two aspects to TA (Johnson, 1992): gathering information and analysing users’ tasks (described in Sections 4 & 5); and the generation of task models (described in Section 6) and task scenarios into running prototypes, in order that the outputs of the user-task analysis support a mapping and checking operation between user tasks and

proposed design. In this section, we will describe how we used the information obtained from the cognitive task graphs for *browsing*, in conjunction with design principles and guidelines, to develop a design model, a prototype hypertext system. Design models can, of course, be represented in a variety of other forms: formal specifications, informal specifications, screen drawings, storyboards and simulations (Johnson, 1992).

Considerations for the design of the prototype hypertext system include: how best to support and improve the various aspects of the task *browsing*; how to support the way users could carry out actions while *browsing* hypertext; identify how much information should be provided on the screen at any one time, as well as give recommendations and suggestions on how to present information on the screen.

There are two aspects of design to which the cognitive task graphs directly contribute:

- *Design interface* is concerned with the visual appearance and screen layout of the user interface. This may include the design of pop-up menus, windows, icons, buttons, text fields, command names. It is used as a channel of communication through which the state of hypertext, accessible functions and feedback are presented to the user. Though cognitive task graphs may not give sufficient detail for complete screen layout or interaction style, they provide an initial input and guide for user-interface design.
- *Functionality* has to do with the performance and capability of the system. This may include navigational buttons, table of contents, home page, building links for learning.

Figure 8 shows our interpretation of the design of a simple prototype hypertext system produced from the cognitive task graphs. For example, if a user wants to perform *browsing*, then a pop-up menu is shown that allows the user to select any of the subtasks associated to *browsing* as described in Section 4.

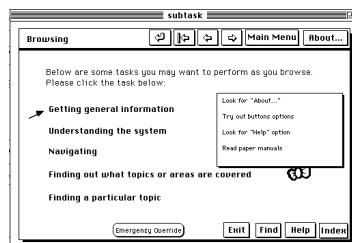


Figure 8. Interface design of a simple prototype hypertext system

8. Conclusions

We have shown developing cognitive task graphs for common tasks in hypertext that users' needs is straightforward. We showed that experts' initial task graphs are easily refined with cheap studies with actual users. We then showed that the resultant graphs

can be interpreted by cognitive modelling tools. Based on the analysis of users' needs, we have implemented a prototype hypertext system.

The cognitive modelling tool, being an executable program, is itself executable from a hypertext system as if it was a user. Though we did not show this, it is standard LISP programming to achieve it. Not only does this allow automatic evaluation, it allows very large scale evaluation by introducing randomness into the model. Obviously such evaluation does not substitute for human evaluation, but it is cheaper, more comprehensive, and can identify critical incidents rapidly.

Subsequent work will involve building different cognitive user models for different user types (e.g., novice, intermediate, experienced).

Acknowledgements

We thank the users for taking part in the experiments. CUM-DesTool was built by Cécile Rigny as part of her PhD work sponsored by the French Ministry of Defence. The analysis of hypertext design issues was done by Yin Leng Theng as part of her PhD work sponsored by Middlesex University. The authors would like to thank Dr Ann Blandford and Dr Mark Addison for their helpful comments on earlier versions of this paper.

References

ACM (1989), "ACM Hypertext-on-Hypertext," ACM Press Database and Electronic Products Series.

Hansen, W (1971), "User engineering principles for interactive systems," *AFIPS Conference Proceedings*, **39**, AFIPS Press, pp. 523–532.

Johnson, P (1992), *Human Computer Interaction: Psychology, Task Analysis and Software Engineering*, McGraw-Hill.

Landauer, T K (1995), *The trouble with computers: Usefulness, usability and productivity*, MIT Press.

Lewis, C and Rieman, J (1995), "Getting to know users and their tasks," In Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (Eds) *Human-Computer Interaction: Toward the Year 2000*, Morgan Kaufmann Publishers (U.S.A.), pp. 122–127.

Newman, W M and Lamming, M G (1995), *Interactive System Design*, Addison-Wesley.

Payne, S and Green, T R G (1989), "Task-action Grammar: The model and its developments. In *Task Analysis for Human-Computer Interaction*, Ellis Horwood.

Preece, J, Rogers, Y, Sharp, H, Benyon, D, Holland, S and Carey, T (1994), *Human-Computer Interaction*, Addison-Wesley.

Reason J (1988). Framework models of human performance and error: a consumer guide. In L.P. Goodstein, H.B. Andersen & S.E. Olsen (Eds.), *Tasks, errors & mental models*. pp. 35–49, Taylor & Francis: London.

Rosson, M B, Mass, S, and Kellogg, W A (1989), “The designer as user: building requirements for design tools from design practice,” *Communications of the ACM*, **31**(11), pp. 1289–1298.

Suchman, L A (1987), *Plans and situated actions: The problems of human-machine communication*, Cambridge University Press.

Thimbleby, H (1995), “Hypertext authoring without getting lost,” *HCI’95 Adjunct Proceedings*, pp. 118–124.

Thimbleby, H (1990), *User Interface Design*, ACM Press (U.S.A.).

Wharton C., Rieman J. & Polson P. (1994). The Cognitive Walkthrough method: a practitioner’s guide. In J. Nielsen and R. Mack, Eds. *Usability Inspection Methods*. pp. 105–140. Wiley: New York.