Heuristics for Mindtools*

Harold Thimbleby
Stirling University
Scotland, FK9 4LA

Abstract

An analogy is made between printing technology and hypertext. Printing is seen as a 'mindtool,' but hypertext's potential as a mindtool has not yet been realised. However, heuristics, both conceptual and particular, are readily available to lift hypertext and other computer-based systems from their present inexcusable condescension of the user's mind to being true mindtools.

The Chinese started using printing around AD175 for Confucian texts. They later invented, then abandoned, movable block type; their complex orthography making it a relatively minor intellectual diversion. The full awakening of printing waited till Johannes Gutenberg (who also invented several support technologies: inks, presses and so on) working in a different culture, *c.* 1440–50. In just the following fifty years, the technology of printing became a *mindtool*, rather than just another way of making books:

- The author now wrote not for a particular dedicatee or patron, but for the public. Printing gave a single author many readers, and the author's income now came from many readers unknown to himself. Note that Latin books had an enormous international market.
- When texts had been copied by hand, pagination (itself a major invention) varied from copy to copy, but now the same information in any copy of a book could be found in the same place on the same page; hence a teacher could say to his students, "look at page 4." Knowledge tools, such as tables of contents, indices and so on became practical propositions.
- Diagrams became usable; prior to printing, manual transcriptions of drawings made them unreliable. Petrus Ramus exploited the reliable duplication of diagrams to popularise his logic and analytical methods.
- Books ceased to be meant to be read aloud from beginning to end, but could be read in any order and browsed.
- Information became cheap; readers could annotate their own copies of books. Knowledge could be questioned independently of the economic considerations of having the knowledge at all!

---

* From: *Cognitive Tools for Learning*, P. A. M. Kommers, D. H. Jonassen and T. Mayes eds. , NATO ASI volume **F81**, 1992, pp161–168, Springer-Verlag.

- Aldus Manutius brought out a series of cheap ‘pocket’ books (which were widely pirated). Books represented, for the first time, accessible, durable, portable and *personal* information.
- Personal knowledge—‘education’—became not a matter based on rote use of aural memory, but of flexible visual and spatial abilities.
- William Caxton in England, and his contemporaries on the continent, published all they could of their national literature. In the process they helped define their own languages.

Printing was spread with missionary zeal*—but as can be imagined from the enormous political impact (merely hinted at in the list above) many attempts were made to suppress and control the new technology.

Such observations are reminiscent of the current assessment of hypertext, even down to the problems of copyright and piracy. Is hypertext poised to become a new mindtool, rather than just another way of packaging information? Or is it to become a read-only, highly duplicated mass media: multimedia CD-roms for domestic television sets?

It seems that hypertext may well be *poised*, but it certainly is *not yet* a mindtool. For it treats the user as a passive recipient of information, whether users work alone or collaboratively (though work is in progress—see: Witten, Thimbleby, Greenberg & Coulouris, in press). Indeed, it would be no exaggeration to say that hypertext and other contemporary computer-based systems treat their users as mere peripherals, perhaps random–access peripherals, but as peripherals nonetheless. Not even as computers, capable of inferences. Not even as human, but, at best, rather like discs: a ready sink of facts.

The purpose of this article is to claim not just that users are treated as less than computers (which is dehumanising and exploitative) but that techniques are available to do better, much better. We will argue that the recognised problem of “getting lost in hypertext” (Nielsen, 1990a) is, in large part, due to the condescension of hypertext designers—an attitude shared with all other computer systems designers (Thimbleby, 1989). We will argue that treating the user at least as well as we would treat a computer is a powerful heuristic, indeed, a mindtool for design—an issue which is taken up elsewhere for user interfaces in general, rather than just hypertext (Thimbleby, 1990). Indeed, Runciman and Hammond (1986) convincingly argue that user interface insights can be obtained directly by ‘programming the user.’

For a concrete example, consider HyperCard (Apple, 1987). Like all hypertext systems, HyperCard presents the user with a collection of information nodes (*cards* in HyperCard’s terminology) that may be linked arbitrarily. The nodes typically contain both textual and pictorial information, and in some cases animation, sound or speech. In HyperCard the user can view a single card at any one time, though this restriction has recently be relaxed slightly. The user interacts with HyperCard by, for example, pressing so-called *buttons* which cause other cards to be displayed. Thus the user *navigates* around a graph (represented by the cards and their links); the argument being that the generalisation from the sequential ordering implicit in a conventional book to an arbitrary graph can be exploited to the user’s advantage. However, in practice, it turns out that the user rapidly gets lost or disorientated.

Readers of early books had to read them sequentially, partly because they had no index or table of contents, partly because such aids would not have helped because the author did not structure his text to conform, and partly because the texts adopted entirely different strategies to the problem (for example, poetry). Also, annotating a book (e.g., highlighting points of interest) would have been unattractive because of the high value and almost irreplaceable nature of the book itself. Likewise, a modern reader of a hypertext is reduced to local browsing—hill climbing, having no overall view or plan of his activity—partly because hypertexts have no appropriate indices, partly because hypertext authors do not yet exploit the medium optimally with respect to the aids available. Also, for technological and proprietary reasons, many hypertexts are read-only, and the

* An interesting discussion of the sixteenth century man Petrus Ramus and his books can be found in (Ong, 1974).

user's annotation is simply impossible. (We will return to the question of writable hypertexts below.)

Most discussions of this navigation problem assume that the cause lies mainly in special nature of the user's abilities and the failure of hypertext systems to properly address these cognitive and perceptual issues. Undoubtedly psychological mismatch between the design and the user is serious; for example, cards have a homogeneous appearance, are seen in isolation and give the user few redundant cues. It has widely been suggested that maps and other navigation aids (Monk, 1989) are recruited. Nielsen (1990b) provides a review of the area.

A hypertext is a graph, and its user embarks on a graph-theoretic problem, of traversal, search or optimisation as the case may be. It is therefore instructive to consider the abstract computational issues as such. Briefly, one would consider the properties of the graph, and what algorithms or heuristics would be appropriate for the anticipated task constraints. An abstract data type would be developed, equipped with appropriate operators. Graph-theoretic problems are widespread and have a considerable literature; the design procedure outlined here is *routine* when writing any computer program. Indeed, a program developed in an *ad hoc* manner would be unreliable and professionally unacceptable.

Now consider a typical HyperCard application and the operators available to the user. In addition to the task-specific links between cards (represented visually as buttons), HyperCard provides several generally available navigational aids whereby certain keyboard combinations or menu items provide navigation by moving: to first, last, previous, or next cards; to most recently visited cards ('back'); and to any one of an idiosyncratic choice of cards ('recent'). See figures 1 and 2 respectively.

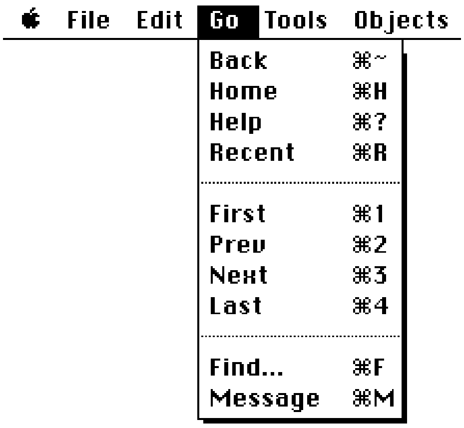

Figure 1. HyperCard's 'Go' menu*

* This paper was written about HyperCard version 1, and the figures are derived from it. The discussion, however, applies equally to HyperCard version 2.

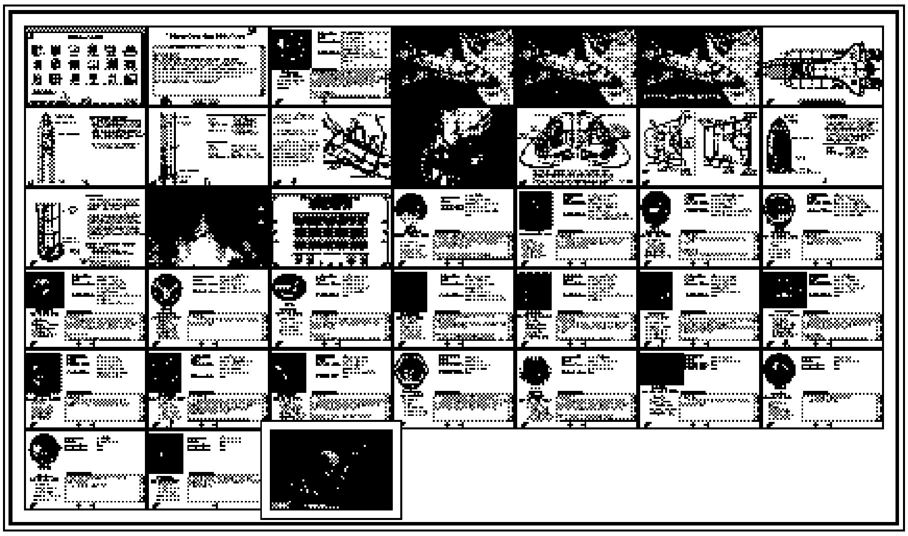

**Figure 2. HyperCard's 'Recent' menu.
(Note that many icons are identical because of lack of resolution.)**

It should be clear that these navigational aids have an arbitrary, and usually misleading, effect on the user's task. For example, the prev (previous) operation shown in figure 1 may provide a link between cards that is not intended to be provided by the specific task. The recent set of cards (at most 36 to 42 of the most recently *first*-visited distinct cards, less if some recently visited cards have been deleted) has no particular computational merit, though it may be justified on perceptual grounds (it provides not necessarily distinct iconic representations of the relevant cards). It is not surprising, then, at least from the computational view, that users experience navigational problems; at best they can use a hill climbing strategy—an approach, regardless of the user's mental limitations, known to be inadmissible in almost all cases, even without the dangers of the user unwittingly getting into cycles.

As a feasibility study, a HyperCard system was developed that could rewrite any other HyperCard system in order to provide the user of that system with a choice of standard search heuristics. HyperCard is programmable, and it is possible to write programs that re-write existing programs in order to change their behaviour: thus it was possible to convert any appropriate HyperCard system from one only providing the default navigational facilities (described above) to one also providing additional strategies. Breadth-first, depth-first, and least cost search heuristics were chosen, but with effort, any AI strategies could have been implemented.

It must be emphasised that the choice of these three simple heuristics was arbitrary, though largely dictated by limitations in the design of HyperTalk, the programming language of HyperCard. For instance, A* is a more appropriate heuristic for many sorts of hypertext problems (A* takes advantage of the fact that the user is rarely interested in the actual path taken to a solution: that is, most hypertext graphs commute), but in HyperTalk there is no way to implement A* in a general fashion without prior knowledge of the particular application and its structure. A survey of heuristics can be found in (Pearl, 1984) and standard texts in AI (such as Winston, 1984).

Cost-based heuristics (in our case, least cost) require a measure, an objective function. Again, the choice was arbitrary even though for any particular hypertext application more appropriate measures could readily be devised. In the present case, cost was based on the number of unused within-hypertext transitions, in other words, effectively the number of untried buttons navigating within the current hypertext document. (Some buttons might provide alternative routes, and this was taken account of; as was the fact that some buttons might have no navigational effect, and some might take the user out of the current hypertext.)

In task-oriented terms, this particular cost function might be appropriate for systematic search based on either of the following user questions: "Where is the card whose exploration I can most quickly complete?" (i.e., with least cost) and "Where is the card that provides me with the greatest choice?" (i.e., with greatest cost). Both alternatives were supported in the study. Note that these questions, exactly as they stand, could usefully be asked by a designer (hypertext author) systematically testing out his hypertext.

It is interesting that the measure and other data to support the heuristics are persistent (i.e., stored in the hypertext indefinitely, even between sessions): when a user comes back to a hypertext he can resume his search exactly where he left it—rather like having a sophisticated bookmark. This is an example of reflexive CSCW (Thimbleby, Anderson & Witten, in press). The reflexive CSCW view further suggests that the AI-type ideas developed here for hypertext would also be useful for users of filestores. (Navigation in filestores is less of a problem than in hypertext—but no less amenable to improvement!—first because the filestore structure is generally a tree rather than a graph, and secondly because there are generally many fewer nodes, though each node—file—is larger than a typical hypertext node.) Persistence requires writable memory (to record the user's activity and side-effects on the measures), and this means that hypertexts on CD-rom and CD-I (interactive CDs, read-only optical discs) are unsuitable, particularly so since CDs have a vast storage capacity that exacerbates the user's navigational problems.

A happy side-effect of the system was to provide a 'show interesting buttons' function: to highlight as-yet unused navigational buttons on any card. Since an idiom in HyperCard is to exploit concealed (invisible) buttons, this feature had an obvious user benefit (see figure 3). It would also have been considered an essential feature for systematic search merely from the computational view: its provision alone permits the user to implement standard graph colouring algorithms for systematic search.

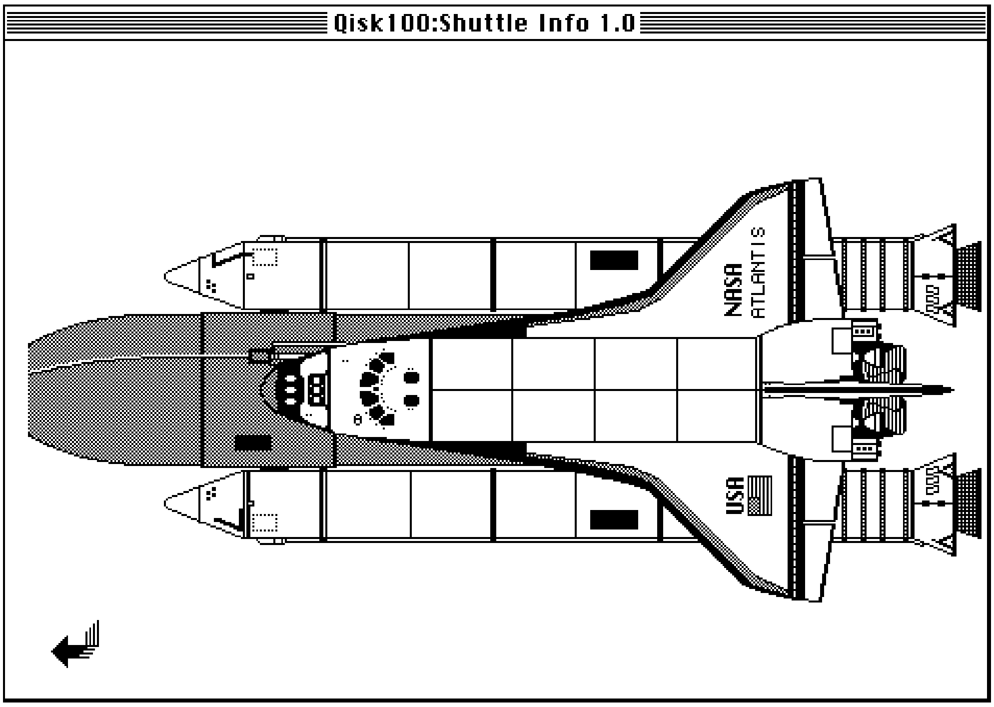

Figure 3. A typical HyperCard card. Where are the buttons? (picture: NASA)

The user interface provided for the new features is rather tedious, depending on arbitrary key combinations. This design decision was forced in order to avoid possible undesirable interactions with the original hypertext behaviour. For example, providing new buttons to facilitate navigation could have had side-effects on existing HyperTalk code. The design choices were made, then, subject to certain unavoidable restrictions,* in order to make the approach applicable to the widest possible range of test hypertexts.

Notwithstanding these superficial restrictions and bearing in mind that the decision to make a general tool necessarily lost opportunities for task-specific optimisations, the study can be considered a success. What it demonstrated was that if the user is treated 'as' a computer, that is, treating the user interface as requiring the design considerations *normal* for programming, then the user interface becomes far more powerful. In other words, this approach brings hypertext into the realm of mindtools.

Treating the user as a computer in itself is clearly a creative mindtool, for it recruits the entire body of computational solutions and methods, both algorithmic and heuristic, to user interface design. To do less is to exploit the user. Furthermore, the Church-Turing Thesis suggests that this approach is not optional but necessary, for hypertext or whatever purpose (Thimbleby, 1990): if, in principle, a computer cannot operate a user interface, then a human certainly cannot or could only do so by trial-and-error.

Whether hypertext and other computer-based technologies will have a social effect as dramatic as the printing press can only be judged from the vantage of the far future. Printing itself was a combination of technologies situated in the fortuitous and fertile circumstances of the Renaissance. At first just a technology to avoid transcription errors, printing was soon to fight in the intellectual wars of the Reformation, and was in turns suppressed and encouraged. That fighting is reminiscent of the current wars on intellectual property rights, software piracy, the status of remote working, copyright (including font copyright) and desk-top publishing. And in these wars of conflicting interests there is the increasing freedom of the individual to use his mind—and body—as a computer-enhanced tool. The computer will be a mindtool only when the user is held in at least as high regard as the technology he uses.

References

Apple Computer Inc. (1987) *HyperCard User's Guide*.

A. F. Monk (1989) "The personal browser: a tool for directed navigation in hypertext systems," *Interacting with Computers*, 1(2), pp190–196.

J. Nielsen (1990a) "The art of navigating through hypertext," *Communications of the ACM*, **33**(3), pp296–310.

J. Nielsen (1990b) *Hypertext and Hypermedia*, Academic Press.

W. J. Ong (1974) *Ramus: Method, and the Decay of Dialogue*, Octagon Books.

J. Pearl (1984) *Heuristics*, Addison-Wesley.

C. Runciman & N. V. Hammond (1986) "User programs: a way to match computer systems and human cognition," *Proceedings of British Computer Society Conference on People and Computers: Designing for Usability*, M. D. Harrison & A. F. Monk, editors, pp464–481, Cambridge University Press.

H. W. Thimbleby (1989) "You're right about the cure: don't do that," *Interacting with Computers*, **2**(1), pp8–25.

H. W. Thimbleby (1990) *User Interface Design*, Addison-Wesley.

H. W. Thimbleby, S. O. Anderson & I. H. Witten (in press) "Reflexive CSCW: Supporting long-term personal work," *Interacting with Computers*.

P. H. Winston (1984; second printing) *Artificial Intelligence*, Addison-Wesley.

* The syntax of HyperTalk itself is a serious hindrance to metaprogramming. Among the many forms of the `go` command, for instance, HyperTalk supports forms such as '`go card x`', where the symbol `x` may be either a *literal* card name or a *variable* name evaluating to a card description (i.e., a dynamically computed transition), and worse '`do x`' where the variable `x` might have the value '`go y`': such cases are impractical to analyse statically!

I. H. Witten, H. W. Thimbleby, S. Greenberg & G. F. Coulouris (in press) "Liveware: A new approach to sharing data in social networks," *International Journal of Man-Machine Studies.*