Analyzing and Visualizing Log Files:
A Computational Science of Usability[1]

Mark Guzdial, Paulo Santos, Albert Badre, Scott Hudson, Mark Gray
GVU Center
College of Computing
Georgia Institute of Technology

Introduction

Researchers in human-computer interactions know that software can easily be instrumented to create a trace of user events[2] in the interface (which we call a *log file*) for later analysis. Using these data for studying usability or other HCI questions is certainly attractive.

- The data are cheap – data gathering is totally automated.
- The data are gathered discretely, so the possibility of a Hawthorne effect is diminished.
- Data can be collected outside of the laboratory, while the user is engaged in real tasks, so the results have greater ecological validity.

The problem has always been what to do with the volume of data that gets generated. With so much data, it's difficult to determine what's useful and how to display the useful portion in a meaningful way.

HCI is not the first discipline to be faced with analyzing mountains of data. Meteorologists gathering a nation or world of weather data, or chemists simulating interactions at the molecular level must also make sense of large volumes of data. The answer in these fields has been to harness computational power to create a new kind of science: *computational science* [8, 31]. Upson et al. [36] provide a diagram of what activities constitute simulation-based computational science, a modified form of which appears as Figure 1.

- **Computer modeling.** Based on research, a program is created which performs a numeric simulation with a specification of the particular domain being studied.
- **Computer simulation.** The program is run to create data for analysis.
- **Computer visualization.** The data is analyzed by creating graphic representations of the data: *visualizations*. If satisfactory, the visualization is summarized as the results of the analysis. Otherwise, the cycle begins again with new insight into the problems.

Simulation for studying HCI is a current area of research (see, for example, Peck and John's work in SOAR [32] or Kieras' work with GOMS [27, 28]). We can imagine using simulation to create simulated log file analyses for depicting predicted user behavior, which might then be compared to log files gathered from actual users. However, modeling and simulation do not help in addressing the primary problem of too much data. Simulation just adds more to the pile.

Visualization seems to offer a better hope. Visualization is about filtering data, mapping the remaining data to graphics primitives, and rendering the result for repeated playback and study (see Figure 2 from Upson et al. [36]). Graphic representations of log file data are fairly common in the literature, but what's missing are the representations which take advantage of human visual processing in the ways that computer visualizations allow us to. For example, there is little use of color, the actual screen layout as the geometry for the representation, and dynamic representations (e.g., animations) – all of which are common characteristics of modern visualizations [3, 4, 12].

---

[1] Presented at HCI Consortium Workshop, Feb. 2-6, 1994. The research described was funded in part by a grant from Intel.

[2] We can imagine other interfaces where the concept of an *event* is less obvious, e.g., moving in a virtual reality. Our focus is on interfaces where events are the focus, e.g., graphical user interfaces.

The goal of visualizations is to provide a new way of seeing the data. In the oft-quoted words of Hamming, "the purpose of computing is insight, not numbers" [21]. In this paper, we will review the literature on log file analysis; consider how people have visualized log file data and offer a taxonomy for log file visualizations; and finally present some new methods of visualizing log file data to describe some of the potential for using visualization to gain insight into log file data.

Previous Work on Log File Analysis

Log file data comes in two types: resulting from generic data collection software (e.g., [20]) and resulting from software modifications (e.g., [13]).

- Figure 3 is a segment of a log file generated by CHIME [2, 33] which collects log file data on use of any Macintosh application. The advantages of generic data collection software is that it can be used to study use of any software, but the disadvantage is that the log file is typically at a fairly low-level. The data in Figure 3 identify events (ButtonPress), time stamps, locations of the mouseclicks, and current window identification, but do not identify what was being clicked.
- Figure 4 is a segment of a log file generated by a multimedia composition application that was modified to generate log file data. The advantage of this approach is that the semantics of the user actions can be included in the log file data, such as what operation was selected, what item was selected, and so on. The disadvantages of this approach are (1) that it requires access to the source code of an application and (2) that it is difficult to determine what level of event should be output. If both the low-level events (e.g., mouseclick locations) and higher-level events (e.g., button name) are output, the volume of data is even larger. If they are not both output, then there is a danger that the desired data during analysis will not be available (e.g., you may know what button was pressed, but not where it was pressed, in order to measure mouse travel).

A good compromise solution is to build the log file generation into the UIMS, as Olsen and Halverson did in the MIKE UIMS [30]. This approach guarantees that data is automatically generated, guarantees the kind and level of data available for analysis, and can present some semantic information as well as low-level data. The disadvantage is that the application must be written in the given UIMS, so is ineffectyg.

Log files have been analyzed to answer a fairly wide range of questions.

- **Usability Measures.** Log file data have often been used to compare interfaces or evaluate the effectiveness of a given interface. Perhaps the most extensive work of this kind is Olsen and Halverson's work on the MIKE UIMS [30]. The MIKE UIMS gathered data on a range of metrics, including the time required to perform operations, the extent of screen updates, and the commands executed by a user. Good used log file analysis to compare the usability of text editors (in terms of the kinds of commands used) [15]. Gray has used digitized screen images (a kind of log file in an alternative medium) for determining the goal tree of students in using an interface [16].
- **Usage Patterns.** Log files are also often used to characterize usage patterns as part of a formative evaluation – not to analyze the current interface as much as to inform the design of the next interface. For example, both Instone et al. [26] and Egan et al. [13] have used log file analysis to identify when users frequently pair two commands in hypertext systems, which suggests either that the commands be combined or that the screen be laid out in such way to make easier the combination of the commands. McDonald et al. [29] have used log file analysis to construct Pathfinder networks of how UNIX commands are conceptually related, in order to define conceptually useful hypertext links in a UNIX help system. Guzdial et al. [18, 19], Dershimer et al. [9, 10], Winne and Gupta [38], and Horney and Anderson-Inman [24] have all sought to identify usage patterns of groups of students in order to determine how interfaces might be tuned to different kinds of users.

For example, Horney and Anderson-Inman are identifying kinds of readers of hypertext documents in order to define features that facilitate each kind of reader.

- **Inferring knowledge or expertise.** Vaubel and Gettys [37], Desmarais and Pavel [11], Santos and Badre [33], and to a lesser extent, Hammer and Rouse [20] have sought to use log files to infer knowledge or expertise of users. Vaubel and Gettys infer the text editing expertise of users by noting key metrics such as the number of requests for help and the use of "power key" equivalents for menu operations. Desmarais and Pavel study UNIX command sequences, and given the typical order of UNIX learning, infer what commands a user already knows. Santos and Badre have been comparing novice and expert users in terms of the length and composition of their interface chunks. Hammer and Rouse also compared novice and expert users in terms of the sequence of events characterized as a Markov chain.
- **Ethnographic analysis.** As the recent Exploratory Sequential Data Analysis (ESDA) workshop notes [14], there is a growing interest in studying how the interface relates to other events around the user. Tools for this kind of analysis are still in development. Trigg has developed a tool which allows for multiple data streams on a single timeline, which would permit analysis of log file data juxtaposed with other analyses of the setting for the interface use [35].

In general, the ESDA workshop noted that tools and methods for supporting log file and similar analyses are just starting to appear. Most are homegrown for particular purposes. In the following section, we describe the kinds of visualizations that various researchers have developed for their analyses.

Current Log File Visualizations

Figure 2 is a depiction of the process of developing and using visualizations in analysis (borrowed from [36]). For the purposes of this paper, rendering primitives into visualizations and playing these visualizations back are not the key points. The critical questions for reviewing and developing new kinds of log file visualizations are filtering and mapping – What data gets studied (filtering) and how does it appear (mapping)?

We will use filtering and mapping as the dimensions for considering the kinds of visualizations which have been used in log file analysis. Table 1 summarizes these analyses. Up the left side of the table are mappings in increasing dimensionality, from scalars through three-dimensions. Across the bottom are different filters (focii in analyses), both time and space-based.

The filtering categories we define here are based mostly on what exists in the literature, as opposed to what one might imagine as filtering categories.

- We distinguish between time-based analyses (e.g., considering just the ordering of the events in the log file[3] versus the absolute time of when the events occurred) and space-based analyses (e.g., considering screen layout). Certainly one can imagine analyses that consider, for example, both screen layout and absolute time, but we don't see those kinds of analyses currently reported in the literature. A more complete taxonomy would probably define three dimensions to input space: time-based (event relative, absolute time, and chunk relative), space-based (screen relative), and external to the interface (on-line and off-line events). All three dimensions could play a role in an analysis and a visualization.

---

[3] Event-relative ordering is not just a time-based metric, since we can imagine using events to measure events in the user's input space (e.g., use of the mouse versus use of the keyboard). For our purposes, we will combine time and space in the term *event-relative* when considering the user's input event stream.

- We also note the importance and feasibility of chunk-based analyses. Olsen and Halverson's MIKE UIMS [30] explicitly associated events with tasks, allowing for analysis in terms of tasks (or as their paper put it, "blame" could be assigned to time/effort-expensive tasks by studying the component events). More recently, Santos and Badre [33] have developed techniques for identifying chunks from low-level events such as mouseclicks and key presses. As Olsen and Halverson point out, chunk-based analyses may be even more meaningful than time-based analyses since they allow for studying usability in terms of conceptually meaningful units.

The mapping categories are particularly interesting for identifying the focus of recent work in log file visualizations:

- **Scalar.** Much of the work in log file analysis has been aimed at producing a number to be used in quantitative analysis: an expertise level, command frequencies, or the output space and extent, as examples. While certainly useful for activities such as hypothesis testing, quantitative analysis is too gross a tool for capturing the kinds of data needed for determining usage patterns, for example.
- **1-D.** The formative evaluation uses of log files are most often concerned with what events follow what other events – looking at events or screen updates along a timeline. Badre, Hudson, and Santos have been using one-dimensional timelines based on chunks to represent and provide access to a control stream [1] (Figure 5).
- **1-D Color and Dynamic.** One might imagine that, even in one dimension, using color or displaying events on a timeline in a dynamic fashion may aid human perception in noting patterns which might otherwise be missed. We discuss such a visualization technique in the following section.
- **2-D Abstract**. There is a good deal of work studying how we might display log file data in various abstract two-dimensional representations. These range from a simple table of the frequency of command pairings (how often command X follows command Y, for all X and Y) to Markov network charts (Figure 6) showing the observed probability of one command following another, action code charts (Figure 7) showing the ordering of events indexed by position in a hypertext database, and process pattern charts (Figure 8) showing the interleave of events on an absolute time scale.
- **2-D Color and Dynamic.** Color might be used to add additional dimensionality to an abstract two-dimensional visualization. One might also imagine using the geometry of the screen as part of a dynamic visualization of user events.
- **3-D.** Modern visualization techniques allow for complex, photorealistic three-dimensional images. While these techniques have not yet been used in log file visualizations, we suggest an application in the following section.

In general, while there have been interesting representations of log file data in the literature, there has been little use of the power of computational visualizations as of yet. Log file analysis has not yet developed into a computational science, in the sense of computational chemistry or meteorology. The following section presents some suggestions and prototypes for how more advanced visualization techniques might be applied to log file analysis.

New visualization approaches for log file analysis

Our technique for developing the visualizations in this section has been to consider analyses from the literature and develop alternative visualizations which take advantage of color and dynamics. As Table 2 describes, we see this activity as adding a higher dimensionality to existing visualizations, which we hope presents more information while still in a meaningful manner. We present these visualizations in increasing dimensionality of mapping (output space).

Our vision for the use of this kind of visualization is cyclical, as depicted in the Upson et al. figure (Figure 2). When first faced with log file data, we imagine that analysts and researchers would use higher-dimensionality visualizations to gain insight into the dataset. During further cycles, the researchers may drop to lower dimensionality visualizations, to fine-tune the analysis to areas of interest. Finally, the scalar metrics might be used to compute measures across interfaces or users for comparison.

1-D Color: Color ESDA Patterns

Exploratory Sequential Data Analysis is concerned with noting patterns in sequential data, like log files and conversational data [14]. We are interested in the use of color for identifying these patterns. Figure 9 is a screenshot from a prototype that uses color to highlight patterns in a log file. The screenshot depicts using color to identify menu use in a multimedia composition system. The window on the right of Figure 9 displays patterns found in the log file data as colors. The window on the left defines what data is to be read, the patterns to be noted (currently in a text programming language format), the colors to be assigned to the patterns, and the time scale for depicting the patterns. The two time scales highlight different characteristics of the data:

- Time relative compacts the entire log file into the space of the Sequences window (100 segments), using sampling to determine which event colors each line. Time relative would be important, for example, in noting the dispersement of user events among devices (e.g., how much time spent with the mouse, how much time spent with the keyboard, and how this time was interleaved).
- Entry relative selects only the first 100 entries in the log file and displays those one per line in the Sequences window. Entry relative highlights the existence of other events which might be affecting the patterns seen in the Sequences window. For example, there are large gaps between command uses in Figure 9, which would suggest that perhaps other significant events are occuring besides those identified which could be influencing menu use.

2-D Color and Dynamic: Overlapping and Blending Screens

Olsen and Halverson developed some interesting measures of output space and extent to characterize where the screen is getting updated to determine if the user's attention is getting stretched across a wide area during interaction with the interface [30]. Rather than identify a number for the extent, we are using overlapping screens which sum color levels to show areas of screen updates. Figures 10 and 11 are examples of this approach. We digitized a ten second clip of a user manipulating an appointment application at four frames per second, then overlaid these frames where later frames sit on top of earlier frames. We are exploring different ink effects for highlighting output space and extent.

- In Figure 10, we use an ink effect such that the later frames' colors are blended (*addmax*-ed) with the earlier frames such that the resultant color of any pixel is the maximum values in the source and destination for each of red, green, and blue. Figure 9 is displayed on a gray scale where white is the maximum color. Thus, white areas in Figure 9 show areas where either white has been displayed or that the overlaid layers of screens have added up to white.
- In Figure 11, we use an ink effect such that the later frames' colors are blended (*addmin*-ed) with the earlier frames such that the resultant color of any pixel is the minimum of source and destination on each of the red, green, and blue values. The effect is that we can "see through" all the layers and see where updates took place: menus dropping, highlight different days, and dialog boxes.

Neither of these methods are entirely satisfactory, though they indicate promise. We are exploring alternative blending operations where pixels that do not change are slowly grayed out with successive layers while changes are highlighted or darkened to make them more pronounced.

We are also exploring dynamic representations of overlaid screens, as part of our research into combining video and log file data [1]. Given a digitized representation of a user's interaction (say, 400 frames from a 100 second session at four frames per second), we are experimenting with blending ten or more of these frames, by averaging the colors in each set of ten in order to reduce the number of frames by a magnitude. We are interested in whether the interface expert can get the same understanding of the user's interactions in less time by viewing the reduced data set. Our experiments suggest that, again, the kind of blending or averaging function used is critical. It is an empirical question how far we can compress the video stream with a good blending function that highlights change.

2-D Dynamic: Mouseclick Locations

Another dynamic representation we are exploring is tracking mouse movement (measured as distance between mouse clicks). Rather than compute a mouse movement distance (as did Olsen and Halverson), we are dynamically displaying the mouse movements (at an increased rate) and representing mouse clicks as a darkening of a partitioned representation of the screen. Figure 12 is a screenshot of the running visualization, reading from a CHIME log file of a user working with a Macintosh drawing package. As each mouseclick is encountered in the log file, the partition in which the mouseclick occurs is darkened, but overall, all partitions are lightened slightly. The effect is that clicks increase the highlighting (darkening) of any partition, but highlights fade over time. At any moment, the darkened partitions indicate a working set of recently accessed areas of the screen, where darker indicates more recent or more frequent use. In the particular visualization of which Figure 11 is a snapshot, partitions in the upper left are the most often dark because of the tool bar appearing vertically on the left of the screen.

We are exploring variations for this visualization.

- In cases where there are buttons or toolbars to be clicked, a representation of the typical screen underlying the partitions would help to make sense of the user's mouseclicks. In tools such as text editors, the underlying representation may not be as useful.
- Rather than partition the screen statically, we are considering darkening a circle around a click. In this variation, frequently accessed areas would be blurred, and the shapes and sizes of the blurs may be more informative than the static partitions.

2-D Dynamic: Visual Command Frequency and Recency

Olsen and Halverson were also concerned with menu operations: command frequency, command pairs, and position on the menu bar of frequently used commands. Figure 13 uses the same technique as Figure 12 (partitioning the screen where use darkens but all highlights fade), but on menus instead of the overall screen. Here, the set of darkened positions indicates commands which are often used together. This visualization can provide information on how often commands are used, which commands are used together, and on whether the placement of commands in the menu is optimal given their frequency of use.

3-D: Mouseclick Depth

Another way of looking at the mouseclicks is in terms of density, as if the mouseclicks pile up on one another. Figure 13 depicts this kind of visualization, where mouseclicks were recorded for use of a HyperCard appointment book and then summed along 30x30 pixel partitions. The height of the cones in the visualization indicates the number of clicks in that partition. Figure 14 is the same visualization viewed from directly above – down the cones onto the underlying screenshot[4].

Conclusion

The goal of the visualizations described in this paper are to provide some evidence for the benefits of developing a computational science of usability based on log file analysis. We feel that there's untapped potential for making log files work for HCI research and usability in particular through use of visualization. While we recognize the importance of exploring pattern recognition and learning routines for making observations on log file data, our current focus is on creating visualizations which allow humans to use their skills at identifying patterns.

The most critical question to be addressed is probably: What visualizations for what questions? There is a developing science of visualization which seeks to define what kinds of visualizations help in drawing out various characteristics of the data (see, [6, 25]). There needs to be a domain-specific follow-up to this effort which looks at what the visualized characteristics of the data tell us about usability and usage patterns, and what questions are better addressed using other methods.

References

1. Badre, A.N., S.E. Hudson, and P.J. Santos. *An environment to support user interface evaluation using synchronized video and event trace recording*. Georgia Institute of Technology, GVU Center. Report #GIT-GVU-93-16. 1993.

2. Badre, A.N. and P.J. Santos. *A knowledge-based system for capturing human-computer interaction events: CHIME*. Georgia Institute of Technology. Graphics, Visualization, and Usability Center Technical Report. Report #GIT-GVU-91-21. 1991.

3. Bailey, M., C. Hansen, T.T. Elvins, and M. Krogh. *Introduction to Scientific Visualization: Tools and Techniques*. SIGGRAPH 93 Course Notes. Report 1993.

4. Brodlie, K.W., L.A. Carpenter, *et al.*, *Scientific Visualization: Techniques and Applications*. 1992, Berlin: Springer-Verlag.

5. Card, S.K., T.P. Moran, and A. Newell, *The Pyschology of Human-Computer Interaction*. 1983, Hillsdale, NJ: Lawrence Erlbaum and Associates.

6. Chappel, H. and M. Wilson, *Knowledge-based design of graphical responses,* in *Proceedings of the 1993 International Workshop on Intelligent User Interfaces,* W.D. Gray, W.E. Hefley, andD. Murray, Editors. 1993, ACM: New York. p. 29-36.

7. Cypher, A., *The structure of users' activities,* in *User Centered System Design,* D.A. Norman and S.W. Draper, Editors. 1986, Lawrence Erlbaum Associates: Hillsdale, NJ.

8. Denning, P., *Computing, applications, and computational science.* Communications of the ACM, 1991. **34** (10): p. 129-131.

---

[4] Currently, these visualizations are created by hand using an assortment of tools, which accounts for the distortion of the images and the impreciseness of the match between screen and graph coordinates. We are exploring better ways of creating these images.

9. Dershimer, C. and C. Berger. *Characterizing student interactions with a hypermedia learning environment* Paper presented at the American Educational Research Association annual meeting, San Francisco, CA. 1992.

10. Dershimer, C., C. Berger, and D. Jackson. *Designing hyper-media for concept development: Formative evaluation through analysis of log files* Paper presented at the National Association for Research in Science Teaching annual meeting, Fontana, WI. 1991.

11. Desmarais, M.C. and M. Pavel, *User knowledge evaluation: An experiment with UNIX,* in *Human-Computer Interaction – INTERACT'87,* H.J. Bullinger and B. Shackel, Editors. 1987, Elsevier Science Publishers, B.V. (North-Holland): p. 151-156.

12. Earnshaw, R.A. and N. Wiseman, *An Introductory Guide to Scientific Visualization*. 1992, Berlin: Springer-Verlag.

13. Egan, D., J.R. Remde, L. Gomez, T. Landauer, J. Eberhardt, and C. Lochbaum, *Formative design-evaluation of SuperBook.* ACM Transactions on Office Information Systems, 1990. **7**: p. 30-57.

14. Fisher, C. and P. Sanderson, *Exploratory sequential data analysis: Traditions, techniques, and tools.* SIGCHI Bulletin, 1993. **25** (1): p. 34-40.

15. Good, M. *The use of logging data in the design of a new text editor*. in *Proceedings of CHI'85*. 1985.

16. Gray, W.D., S.A. Byrnes, and N.C. Goldberg, *Struggling through with HyperCard: A study of end-user programming.* 1993. Draft.

17. Guzdial, M. and J. Merz, *MediaText*. Multimedia Composition Software. 1992, Wings for Learning and Apple Computer.

18. Guzdial, M., C. Walton, M. Konemann, and E. Soloway. *Characterizing process change using log file data*. Georgia Institute of Technology. GVU Center Technical Report. Report #93-44. 1993.

19. Guzdial, M.J. *Deriving software usage patterns from log files*. Georgia Institute of Technology. GVU Center Technical Report. Report #93-41. 1993.

20. Hammer, J.M. and W.B. Rouse, *Analysis and modeling of freeform text editing behavior,* in *Proceedings of the 1979 International Conference on Cybernetics and Society*. 1979, Denver. p. 659-664.

21. Hamming, R.W., *Numerical Methods of Scientists and Engineers*. 1962, New York: McGraw-Hill.

22. Hanson, S.J., R.E. Kraut, and J.M. Farber, *Interface design and multivariate analysis of UNIX command use.* ACM Transcations on Office Information Systems, 1984. **2**(1): p. 42-57.

23. Hay, K.E., P. Weingrad, S. Jackson, R.A. Boyle, M. Guzdial, and E. Soloway, *Student composition of multimedia documents: A preliminary study.* Journal of Educational Computing Research, 1993. .

24. Horney, M.A. and L. Anderson-Inman. *The ElectroText Project: Hypertext reading patterns of middle school students.* Paper presented at the annual meeting of the American Educational Research Association, April. San Francisco, CA. Report 1992.

25. Ignatius, E. and H. Senay, *Visualization assistant.* Workshop on Intelligent Visualization Systems, IEEE Visualization '93, 1993. San Jose, CA. Oct. 25.

26. Instone, K., B.M. Teasley, and L.M. Leventhal. *Empirically-based re-design of a hypertext encyclopedia*. in *INTERCHI'93*. 1993.

27. Kieras, D. and P.G. Polson, *An approach to the formal analysis of user complexity.* International Journal of Man-Machine Studies, 1985. **22** : p. 365-394.

28. Kieras, D.E., *Towards a practical GOMS model methodology for user interface design,* in *Handbook of Human-Computer Interaction,* M. Helander, Editors. 1988, Elsevier Science Publishers B.V.: Amsterdam; New York. p. 135-157.

29. McDonald, J.E., K.R. Paap, and D.R. McDonald, *Hypertext perspectives: Using Pathfinder to build hypertext systems,* in *Pathfinder Associative Networks: Studies in Knowledge Organization,* R.V. Schvaneveldt, Editors. 1990, Ablex: Norwood, NJ. p. 197-212.

30. Olsen, D.R. and B.W. Halverson, *Interface usage measurements in a user interface management system.* UIST'88, 1988. .

31. Pagels, H.R., *The Dreams of Reason: The Computer and the Rise of the Sciences of Complexity*. 1988, New York: Simon and Schuster.

32. Peck, V.A. and B.E. John. *Browser-Soar: A computational model of a highly interactive task.* in *CHI'92: ACM Conference on Human Factors in Computing Systems*. 1992. Monterey, CA: ACM.

33. Santos, P.J. and A.N. Badre. *Automatic chunk detection in human-computer interaction*. Georgia Institute of Technology. Graphics, Visualization, and Usability Center Technical Report. Report #GIT-GVU-94-4. 1994.

34. Shute, V.J. and R. Glaser, *A large-scale evaluation of an intelligent discovery world: Smithtown.* Interactive Learning Environments, 1990. **1**(1): p. 51-77.

35. Trigg, R.H., *Computer support for transcribing recorded activity.* SIGCHI Bulletin, 1989. **21** (2): p. 72-74.

36. Upson, C., T. Faulhaber, *et al.*, *The application visualization system: A computational environment for scientific visualization.* IEEE Computer Graphics and Applications, 1989. **9**(4): p. 30-42.

37. Vaubel, K.P. and C.F. Gettys, *Inferring user expertise for adaptive interfaces.* Human Computer Interaction, 1990. **5**: p. 95-117.

38. Winne, P.H. and L. Gupta. *Data and graph theory measures for modeling cognitive strategies in tutorials with STUDY* Paper presented at the annual meeting of the American Educational Research Association, Atlanta, GA. 1993.