Finding Canonical Behaviors in User Protocols

Walter C. Mankowski, Peter Bogunovich, Ali Shokoufandeh, and Dario D. Salvucci
Department of Computer Science, Drexel University
Philadelphia, PA 19104, USA
{walt, pjb38, ashokouf, salvucci}@cs.drexel.edu

ABSTRACT
While the collection of behavioral protocols has been common practice in human-computer interaction research for many years, the analysis of large protocol data sets is often extremely tedious and time-consuming, and automated analysis methods have been slow to develop. This paper proposes an automated method of protocol analysis to find *canonical behaviors* — a small subset of protocols that is most representative of the full data set, providing a reasonable "big picture" view of the data with as few protocols as possible. The automated method takes advantage of recent algorithmic developments in computational vision, modifying them to allow for distance measures between behavioral protocols. The paper includes an application of the method to web-browsing protocols, showing how the canonical behaviors found by the method match well to sets of behaviors identified by expert human coders.

Author Keywords
Protocol analysis, sequential data analysis, web browsing

ACM Classification Keywords
H.1.2 Models and Principles: User/Machine Systems; H.5.2 Information Interfaces and Presentation: User Interfaces

INTRODUCTION

In the study of user behavior, researchers and practitioners alike often collect data in the form of behavioral *protocols* — sequences of actions recorded during the execution of a task. The analysis of these protocols provides a wealth of information about user behavior, and thus protocol data have been collected for a wide range of data types and utilized in a wide variety of ways. For instance, protocols have been employed for examining manual actions such as mouse clicks and keystrokes [e.g., 2], verbal reports [e.g., 7], and eye movements [e.g., 1], and sometimes combinations of these and other data [e.g., 9]. Based on this rich set of data, researchers have used protocols for such varied purposes as exploratory data analysis [e.g., 15], classification and recognition [14], and cognitive modeling [e.g., 17]. At the same time, the richness and quantity of protocol data have a severe limitation: it is typically difficult, if not impossible, to process and analyze the data by hand, thus requiring some form of data aggregation to make analysis and understanding feasible. While such aggregation provides information with respect to overall behavior, it washes over the interesting patterns that may appear in particular protocol instances from individual users.

In this paper we introduce the idea of *canonical behaviors* in user protocols and propose a computational method to identify them in an automated way. Canonical behaviors are a small subset of the protocol data set that is most representative of the entire data set, providing a reasonable "big picture" view of the data with as few protocols as possible. The identification of canonical behaviors is often performed laboriously by hand in standard protocol analysis; for instance, eye-movement or verbal protocol analysis often includes the identification and dissection of a few individual protocols that exemplify interesting strategies in the task [see, e.g., 7, 14]. While methods for automated protocol analysis have been studied in previous efforts, the methods focus on aligning observed protocols with the predicted behaviors of a user model [e.g., 13, 16]. Our work takes a very different but complementary approach, using a specification of the similarity between behaviors to automatically identify canonical behaviors in a set of user protocols.

To illustrate our approach in a real-world domain, we apply the method to the domain of web browsing. Recent efforts [e.g., 4, 5] have analyzed web-browsing behavior in a number of ways, typically involving some type of aggregation across individual user protocols. There has been some work on the analysis of individual protocols by aligning browsing protocols with the predictions of a cognitive or task model [3, 11, 12]. We aim to complement this work by proposing a method for finding canonical web-browsing behaviors without the need for any type of *a priori* model. The resulting canonical behaviors could be useful, for instance, in determining standard paths to desired information on a given web site, or in identifying circuitous paths of confused users and revising the web site accordingly.

FINDING CANONICAL BEHAVIORS

Our technique for computing canonical behaviors derives from work in the area of computational vision, where techniques have been developed to identify canonical members of a class of visual patterns [6]. In the context of user proto-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CHI 2009, April 4 - 9, 2009, Boston, Massachusetts, USA.
Copyright 2009 ACM 978-1-60558-246-7/09/04...$5.00.

cols, our goal is to reduce a large set of behavioral protocols to a smaller (often much smaller) subset of canonical behaviors that is most representative of the full data set. Specifically, we define a canonical set of behaviors as a subset such that (1) behaviors in the canonical set are minimally similar, and (2) behaviors in the canonical set are maximally similar to behaviors not in the set. The key aspects of the method are the specification of a similarity measure between behaviors and the determination of canonical behaviors given this similarity measure. We now describe each in turn, defining each aspect in general terms as well as providing an instantiation of the framework for our chosen application domain of web browsing.

Quantifying the Similarity between Behaviors

A critical aspect of finding canonical behaviors is to define some measure that quantifies the similarity (or, inversely, the distance) between two given behaviors. Let us assume that we have two behavioral protocols x and y, each of which represents an ordered sequence of actions. The similarity measure can be defined as a function $\mathbf{S}(x, y)$ that returns the similarity between the two behaviors. Clearly this function is dependent on the exact nature of the behavioral protocols: for instance, defining a similarity function for verbal protocols would be extremely challenging if one wanted to take into account the syntactic and semantic complexities of normal speech. Nevertheless, we assume that a similarity measure can be defined at an appropriate level such that it captures the desired aspects of the task domain: for example, for eye-movement protocols, a similarity measure could compare the number of fixated items, the overlap in these items, the exact sequence of items, or even the precise x, y coordinates that embody the fixations and saccades of eye movements [e.g., 14].

For the domain of web browsing, we chose to compute similarity between browsing protocols by using the well-known *edit-distance* metric [10]. Intuitively, the edit-distance $\mathbf{ED}(x, y)$ between two protocols measures the minimum cost of inserting, deleting, or substituting actions to transform one protocol to the other. In our work, we tested two different cost functions. The first cost function simply assigned a uniform cost of 1 to all insertions, deletions, and substitutions. We also tested a second function that computed the shortest path between one web page and another (i.e., smallest number of link transitions between pages), and used the length of the shortest path as the edit-distance cost. In our testing, however, we found that this second cost function did not significantly improve performance; thus in our analysis below, we focus exclusively on results that derive from the first simple cost function. The edit-distance cost was converted to similarity as $\mathbf{S}(x, y) = 1/(1 + \mathbf{ED}(x, y))$.

Determining Canonical Behaviors

Given the above specification of similarity among behaviors, we now focus on finding canonical behaviors within a large protocol data set. First, the user behaviors are encoded into an affinity (similarity) graph where each behavior is represented as a vertex in the graph. Next, we add edges to the graph with weights corresponding to the similarity between the behaviors represented by the connected vertices, as specified in the previous section. Finally, we find the canonical set by computing a cut that bisects the graph into two subsets, as shown in Figure 1.

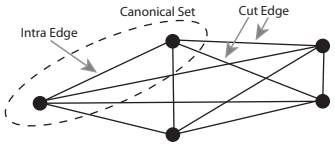

Figure 1. Canonical-set graph with behaviors at vertices and edge weights corresponding to behavioral similarities.

The computation of the canonical set graph cut can be formulated as an optimization problem on the aforementioned graph. The objective is to minimize the sum of the weights of the intra edges — the edges within the canonical set, included in Figure 1 — while at the same time maximizing the sum of the weights of the cut edges — edges between behaviors in the canonical set and those outside the set. Such an optimization is known to be intractable [8], and thus we utilize an approximation algorithm: we formulate the canonical-set problem as an integer-programming problem; relax it to a semidefinite program; and use an off-the-shelf solver [18] to find the approximate solution. Please refer to [6] for details of the full algorithm. In summary, this optimization translates into finding canonical elements that are most representative of the other elements, while keeping the canonical elements as dissimilar as possible — fulfilling our goal of identifying canonical behaviors.

The algorithm includes one free parameter, $\lambda \in [0, 1]$, which scales the weighting given to cut edges versus intra edges. Intuitively, this parameter trades off the priority of minimizing the similarity among canonical elements with the priority of maximizing the similarity between canonical and non-canonical elements. Thus, as we will discuss in our application to web browsing, λ can be thought of as an individual coder's tolerance for accepting variations of canonical elements versus allowing more significant variations to become canonical elements themselves.

DATA COLLECTION

To validate our automated method for finding canonical behaviors, we collected data from users performing standard web-browsing tasks. Specifically, we devised a list of 32 questions whose answers could be found on a university web site, namely that of Carnegie Mellon University (http://www.cmu.edu). The questions comprised 8 sets of 4 grouped questions, where the questions in each group were syntactically and categorically similar — for example, 4 questions of the form "What is the phone number of ⟨*department*⟩ professor ⟨*name*⟩?". The 8 groups covered a range of realistic topics related to finding information about athletic programs, academic departments, and so on.

Each user was asked to answer all 32 questions, randomly presented such that one question from each of the 8 question groups was answered before repeating another question of any group. For each question, the user began at the web

site home page and followed links until the desired information was found. For simplicity in data analysis, users were asked not to utilize either external search engines or the site's own search engine (which itself uses Google). All browsing was done in the Firefox browser (v. 3.0.1) with previously viewed link highlighting turned off. The collected data consisted of the sequences of web pages that were visited by users and the corresponding answers to each question. (Firefox was modified so that navigation when using the "Back" and "Forward" buttons was also logged correctly.) Thus, each behavioral protocol corresponded to a sequence of web pages visited while answering a particular question.

A total of 12 users, 2 female and 10 male, participated in the data collection. The users' ages ranged from 19 to 29 (M=23.5, SD=3.2). Prior to participating, users were asked to rate their familiarity with the CMU web site on a scale of 1–4 (1=never seen, 2=very unfamiliar, 3=somewhat familiar, 4=very familiar); the mean rating was 1.75 (SD=.86), indicating that users were generally starting with little to no familiarity with the web site.

ANALYSIS AND RESULTS

To test our automated method on the web-browsing protocols, we removed protocols corresponding to incorrect answers (9% of all protocols) and ran the method to find canonical behaviors for each question. However, we also required a "ground truth" against which to compare the results of the method. For this purpose, we recruited two expert human coders (a professor and advanced graduate student) with significant experience in analyzing behaviors as part of cognitive task analysis and modeling. We asked the coders, given the set of protocols (as sequences of URLs), to identify subsets of protocols for each question such that each subset represented a different behavior. The coders clearly could have their own notion of what would constitute "similar" and "different" behaviors; rather than defining this specifically for them, we allowed them to use their own expert judgments to decide on the best division of protocols for each question.

Illustrative Example

Let us first examine the automated and coder results for an individual question to illustrate the analysis in detail. Figure 2 shows the various behaviors for a single question ("What is the phone number of . . . ?") for (a) one of the human expert coders, and (b) the canonical set algorithm (λ=.36). In both graphs, each node represents a web page (each labeled with a unique integer), and each edge represents a traversal (i.e., clicked link) from one page to another taken by some user. The expert coder found 6 sets of behaviors: A and B represent different ways of clicking through the academic departments to the faculty member's home page; C and D represent ways of clicking through to a directory search page ("14"); and E and F represent slight variations of C and D. The canonical set algorithm identified 4 canonical behaviors in this same behavior graph; canonical behaviors are shown in bold in the figure, and the other behaviors are labeled in terms of their nearest canonical behavior. The behavior sets found by the algorithm correspond directly to the expert coder's sets A–D. Instead of separating out the expert coder's sets E and F as distinct canonical behaviors, the algorithm (in part because of the choice of λ) decided to group these variations into the nearest canonical set D. Overall, we see from this example what the canonical set algorithm provides: (1) the identification of canonical behaviors that best represent the sum total of all behaviors in the set, and (2) the mapping, if desired, of each behavior to its closest canonical behavior to allow for classification of the entire set in terms of the canonical behaviors.

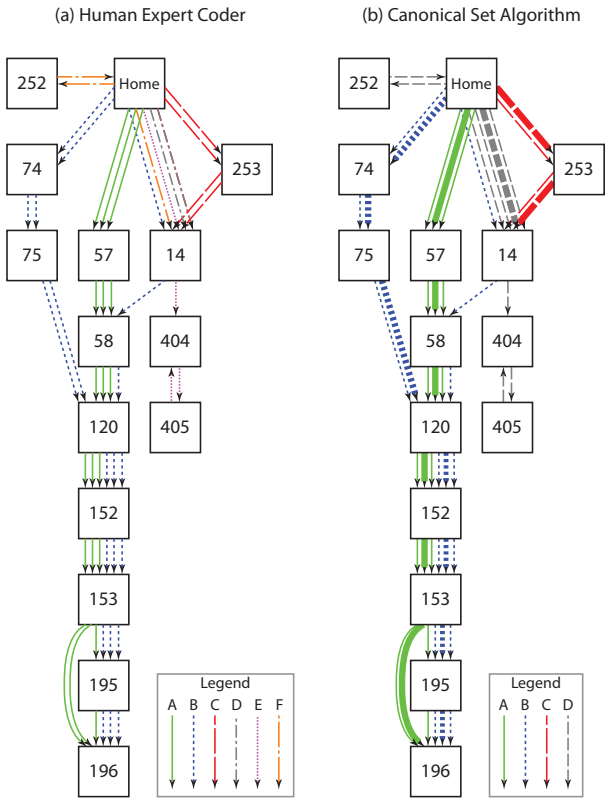

Figure 2. Sample analysis graphs.

Aggregate Analysis

We can also validate the canonical set method by comparing its results to those of the expert coders at an aggregate level, which requires us to define a straightforward measure of error (or inversely goodness-of-fit) of how well the algorithm's and coders' analyses match. Let A be the collection of sets found by an expert coder and B be the set of canonical protocols found for a given λ. The *size error* is defined as $E^{size} = |(\mathbf{Card}(A) - \mathbf{Card}(B))/\mathbf{Card}(A)|$, where $\mathbf{Card}(\cdot)$ is the cardinality of the collection or set. This measure captures the difference in the number of elements in A and B. Let us also define the function $\tilde{\chi}(\cdot,\cdot)$ on two sets X and Y as $\tilde{\chi}(X,Y) = 1$ if $X \cap Y = \emptyset$, and $\tilde{\chi}(X,Y) = 0$ otherwise. The *cover error* is then defined as $E^{cover} = \left(\sum_{A_i \in A} \tilde{\chi}(A_i, B)\right) / \mathbf{Card}(A)$. This measure counts the number of sets A_i in the collection A which do not have any elements in the canonical set B; thus, the measure captures how well the algorithm's canonical behaviors cover different sets found by the expert coder. Finally, we

define our overall error measure *Error* as the average of E^{size} and E^{cover}. For example, for the coder and algorithm results in Figure 2, both E^{size} and E^{cover} are $2/6$, and thus $Error = 2/6$.

Using this overall error measure, we can analyze the behavior protocols for all questions in comparison with the coding of the two human experts. Because the human experts may have different tolerances for the size versus coverage of canonical sets, we varied the λ parameter across a reasonable range and plotted the error measure for this range for both expert coders. These results are shown in Figure 3. Both experts exhibit a roughly U-shaped curve across the range of λ values: higher values favor the canonical behaviors being different from one another, resulting in fewer but larger subsets of protocols; lower values instead favor the similarity of the canonical behaviors to non-canonical protocols, resulting in smaller, more numerous subsets.

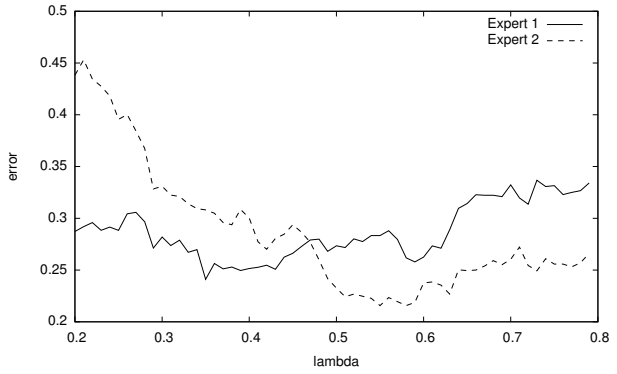

Figure 3. Graph of the overall error between the canonical set algorithm and the two expert coders for a range of λ values.

Each expert coder can be associated with a λ value that best captures their tolerance for the size-coverage tradeoff. The algorithm performed best for Expert 1 at $\lambda = 0.35$, and for Expert 2 at $\lambda = 0.55$. At these values, the experts' average size error was .31 and their average cover error was .15. This means that the algorithm found roughly 70% of the canonical behavior sets identified by the experts, and that roughly 85% of the experts' behavior sets could be associated with one of the algorithm's canonical behaviors. While we have no previous work with which to compare these results, we were pleased with this fairly high match given the lack of any *a priori* cognitive or task model with which to guide the method.

DISCUSSION

We have presented an automated method for finding *canonical behaviors* in user protocols. The method can be used in two important ways. First and foremost, it can identify a small set of canonical behaviors that is best representative of the entire protocol data set. These canonical behaviors can be used, for instance, to extract an interesting subset of behaviors for detailed protocol analysis; to find exemplar protocols for purposes of visualizing interesting behavior; or to extract critical behaviors for purposes of developing cognitive models of user behavior. Second, as a straightforward extension, the method can label every behavior in a data set by associating each behavior in terms of its nearest canonical behavior. Thus, the method can be used to analyze the entire corpus of protocols, for instance, to classify sets of behaviors, to analyze frequencies of related behaviors, or even to interpret behaviors in real time as part of an intelligent interactive system. We believe that this work is an important first step in the development of new automated techniques that facilitate protocol analysis both for individual protocols and across large data sets.

ACKNOWLEDGMENTS

This work was supported by ONR grants #N00014-03-1-0036 and #N00014-08-1-0925 and NSF grant #IIS-0426674.

REFERENCES

1. Byrne, M. D., Anderson, J. A., Douglass, S., & Matessa, M. (1999). Eye tracking the visual search of click-down menus. *Proc. CHI 99*, 402–409.
2. Card, S., Moran, T., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum.
3. Card, S. K., Pirolli, P., Van Der Wege, M., Morrison, J. B., Reeder, R. W., Schraedley, P. K., & Boshart, J. (2001). Information scent as a driver of web behavior graphs: Results of a protocol analysis method for web usability. *Proc. CHI 2001*, 498–505.
4. Chi, E. H., et al. (2003). The bloodhound project: automating discovery of web usability issues using the InfoScent simulator. *Proc. CHI 2003*, 505–512.
5. Cutrell, E., & Guan, Z. (2007). What are you looking for? An eye-tracking study of information usage in web search. *Proc. CHI 2007*, 407–416.
6. Denton, T., Shokoufandeh, A., Novatnack, J., & Nishino, K. (2008). Canonical subsets of image features. *Computer Vision and Image Understanding, 112*, 55–66.
7. Ericsson, K.A., & Simon, H.A. (1984). *Protocol analysis: Verbal reports as data*. Cambridge, MA: MIT Press.
8. Gary, M.R., & Johnson, D.S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. San Francisco, CA: Freeman.
9. Hornof, A. J., & Halverson, T. (2003). Cognitive strategies and eye movements for searching hierarchical computer displays. *Proc. CHI 2003*, 249–256.
10. Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Daklady, 10*, 707–710.
11. Paganelli, L., & Paternò, F. (2002). Intelligent analysis of user interactions with web applications. *Proc. CHI 2002*, 111–118.
12. Pirolli, P., Fu, W.-T., Reeder, R., & Card, S.K. (2002). A user-tracing architecture for modeling interaction with the World Wide Web. In *Proc. AVI 2002*.
13. Ritter, F.E., & Larkin, J.H. (1994). Developing process models as summaries of HCI action sequences. *Human-Computer Interaction, 9*, 345–383.
14. Salvucci, D.D., & Anderson, J.R. (2001). Automated eye-movement protocol analysis. *Human-Computer Interaction*, 16, 39–86.
15. Sanderson, P., Scott, J., Johnston, T., Mainzer, J., Watanabe, L., & James, J. (1994). MacSHAPA and the enterprise of exploratory sequential data analysis (ESDA). *International Journal of Human-Computer Studies, 41*, 633–681.
16. Smith, J.B., Smith, D.K., & Kuptsas, E. (1993). Automated protocol analysis. *Human-Computer Interaction, 8*, 101–145.
17. Taatgen, N. A., & Lee, F. J. (2003). Production compilation: A simple mechanism to model complex skill acquisition. *Human Factors, 45*, 61–76.
18. Toh, K.C., Todd, M.J., & Tutuncu, R. (1999). SDPT3: a Matlab package for semidefinite programming. *Optimization Methods and Software, 11*, 545–581.