On Computing Canonical Subsets of Graph-Based Behavioral Representations

Walter C. Mankowski, Peter Bogunovich, Ali Shokoufandeh, and Dario D. Salvucci

Drexel University
Department of Computer Science
Philadelphia PA 19104, USA
{walt,pjb38,ashokouf,salvucci}@drexel.edu

Abstract. The collection of behavior protocols is a common practice in human factors research, but the analysis of these large data sets has always been a tedious and time-consuming process. We are interested in automatically finding *canonical behaviors*: a small subset of behavioral protocols that is most representative of the full data set, providing a view of the data with as few protocols as possible. Behavior protocols often have a natural graph-based representation, yet there has been little work applying graph theory to their study. In this paper we extend our recent algorithm by taking into account the graph topology induced by the paths taken through the space of possible behaviors. We applied this technique to find canonical web-browsing behaviors for computer users. By comparing identified canonical sets to a ground truth determined by expert human coders, we found that this graph-based metric outperforms our previous metric based on edit distance.

1 Introduction

In many domains involving the analysis of human behavior, data are often collected in the form of time-series known as *behavioral protocols* — sequences of actions performed during the execution of a task. Behavioral protocols offer a rich source of information about human behavior and have been used, for example, to examine how computer users perform basic tasks (e.g., [1]), how math students solve algebra problems (e.g., [2]), and how drivers steer a vehicle down the road (e.g., [3]). However, the many benefits of behavioral protocols come with one significant limitation: The typically sizable amount of data often makes it difficult, if not impossible, to analyze the data manually. At times, researchers have tried to overcome this limitation by using some form of aggregation in order to make sense of the data (e.g., [4,5]). While this aggregation has its merits in seeing overall behavior, it masks potentially interesting patterns in individuals and subsets of individuals. Alternatively, researchers have sometimes laboriously studied individual protocols by hand to identify interesting behaviors (e.g. [6,7]). Although some work has been done on automated protocol analysis, such techniques focus on matching observed behaviors to the predictions of a step-by-step process model (e.g. [8,9]), and often such models are not available and/or their development is infeasible given the complexity of the behaviors.

A. Torsello, F. Escolano, and L. Brun (Eds.): GbRPR 2009, LNCS 5534, pp. 215–222, 2009.

In our previous work we have introduced the notion of *canonical behaviors* as a novel way of providing automated analysis of behavioral protocols [10]. Canonical behaviors are a small subset of behavioral 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. In contrast with previous techniques, our method identifies the canonical behavior patterns without any a priori step-by-step process model; all that is needed is a similarity measure between pairs of behaviors. To illustrate our approach in a real-world domain, we applied the method to the domain of web browsing. We found that the canonical browsing paths found by our algorithm compared well with those identified by two expert human coders with significant experience in cognitive task analysis and modeling. However, our technique was limited by the fact that our similarity measure treated each browsing path as a string, ignoring the underlying graph structure of the web site. In this paper we explore a graph-based similarity measure which takes into account the effects of graph topology when computing the similarity between two patterns.

The remainder of this paper is structured as follows. In Sect. 2 we describe our canonical set algorithm and our new similarity metric. In Sect. 3 we review our web browsing experiment from [10]. In Sect. 4 we compare the results of our new metric with those from our previous experiment. Finally in Sect. 5 we summarize our findings and discuss possible future directions of research.

2 Finding Canonical Behaviors

At a high level, our goal in finding canonical behavior patterns is to reduce a large set of protocols to a smaller subset that is most representative of the full data set. We define a canonical set of behaviors as a subset such that the behaviors within the subset are minimally similar to each other and are maximally similar to those behaviors not in the subset.

Our technique for finding canonical behavior patterns derives from work on the canonical set problem. Given a set of patterns $\mathcal{P} = \{p_1, \ldots, p_n\}$ and a similarity function $\mathcal{S} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$, the *canonical set problem* is to find a subset $\mathcal{P}' \subseteq \mathcal{P}$ that best characterizes the elements of $\mathcal{P}$ with respect to $\mathcal{S}$. The key aspects of our method are an approximation algorithm for the canonical set problem, and the specification of an appropriate similarity metric for the particular problem being modeled. We now describe each in turn.

2.1 Modeling Canonical Sets as Graphs

Exact solutions to the canonical set problem require integer programming, which is known to be NP-Hard [11]. Denton et al. [12] have developed an approximation algorithm using semidefinite programming which has been shown to work very well on a wide variety of applications. First, a complete graph G is constructed such that each pattern (in this case, a behavior protocol) is a vertex, and each edge is given a weight such that $w(u, v)$ is the similarity of the patterns corresponding to the vertices u and v. Finally, we find the canonical set by computing a cut that bisects the graph into two subsets, as shown in Fig. 1.

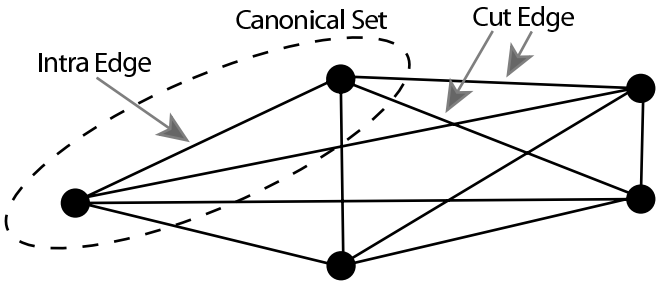

Fig. 1. Canonical-set graph with behaviors at vertices and edge weights corresponding to behavioral similarities (from [10]). Finding the canonical set can be expressed as a optimization problem, where the goal is to minimize the weights of the intra edges while simultaneously maximizing the weights of the cut edges.

Algorithm 1. Approximation of Canonical Sets [12]

1. Construct an edge-weighted graph $G(\mathcal{P})$ from the set of patterns $\mathcal{P}$ and the similarity function $\mathcal{S} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$
2. Form a semidefinite program with the combined objective of minimizing the weights of the intra edges and maximizing the weights of the cut edges (see Fig. 1).
3. Solve the semidefinite program from step 2 using the algorithm in [13], obtaining positive semidefinite matrix $\mathcal{X}^*$.
4. Compute the Cholesky decomposition $\mathcal{X}^* = \mathcal{V}^t\mathcal{V}$.
5. Construct indicator variables $y_1, \ldots, y_n, y_{n+1}$ from the matrix $\mathcal{V}$.
6. Form the canonical set $\mathcal{V}^*$.

The task of determining the proper graph cut to find the canonical set can be expressed as an optimization problem, where the objective is to minimize the sum of the weights of the intra edges — those edges between vertices within the canonical set, as shown in Fig. 1 — while simultaneously maximizing the sum of the weights of the cut edges — those edges between vertices in the canonical set and those outside the set. This optimization is known to be intractable [11] and thus Denton et al. employ an approximation algorithm (please see Algorithm 1): they formulate the canonical set problem as an integer programming problem, relax it to a semidefinite program, and then use an off-the-shelf solver [13] to find the approximate solution. Please refer to [12] for a full derivation and description of the algorithm.

The algorithm includes one free parameter, $\lambda \in [0, 1]$, which scales the weighting given to cut edges verses intra edges. Higher values of λ favor maximizing the cut edge weights, resulting in fewer but larger subsets of patterns; lower values favor minimizing the intra edge weights, resulting smaller, more numerous subsets.

There are two main advantages of the canonical set algorithm compared to many similar methods of extracting key items from sets. First, it is an unsupervised algorithm; no training dataset is necessary. Second, no a priori knowledge of the number of representative elements (in this case, behaviors) is needed. Both the sets themselves and

the most representative elements of the sets arise naturally from the algorithm. As a result, the canonical set algorithm has applications in a wide variety of machine learning areas, for example image matching [14] and software engineering [15].

2.2 Graph-Based Similarity Measures

A critical aspect of finding canonical sets is the definition of some measure that quantifies the similarity (or, inversely, the distance) between two given patterns. Let $\mathbf{S}(x, y)$ be the similarity between two patterns x and y. Clearly $\mathbf{S}$ is highly dependent upon the nature of the domain being studied. For example, in image matching, the earth-mover's distance [16] might be an appropriate similarity measure, while in an eye-movement study the similarity measure might take into account the sequence of items fixated upon (e.g., [7]).

In our original work on web browsing [10], we used a simple *edit-distance* metric [17] to compute the similarity between browsing protocols. Intuitively, the edit-distance $\mathbf{ED}(x, y)$ between two protocols measures the minimum cost of inserting, deleting, or substituting actions to transform one sequence of web pages to the other. We assigned a uniform cost of 1 to all insertions, deletions, and substitutions. The edit-distance cost was converted to similarity as $\mathbf{S}(x, y) = 1/(1 + \mathbf{ED}(x, y))$.

While the edit-distance similarity measure worked well overall, it had one drawback. Web sites are by their nature graph-based (with pages as nodes and links as edges), but the edit-distance measure ignores this and treats each path taken by the subjects as a simple sequence of pages. We hypothesized that our performance would be improved by using a measure which takes into account the underlying topology of the graphs.

As our new similarity metric, we chose to use a modified version of Pelillo's subgraph isomorphism algorithm [18] due to its flexibility in encoding node similarity constraints, especially if the graphs are induced from a fixed topology. In brief, the algorithm works as follows (please see Algorithm 2). Given two graphs U and V, an association graph G is built from the product graph of U and V. A vertex $\{u, v\}$ exists in G for every pair of vertices u in U and v in V. An edge is added between vertices $\{u_1, v_1\}$ and $\{u_2, v_2\}$ in G only if the shortest path distance between vertices u_1 and u_2 in U is equal to the shortest path distance between vertices v_1 and v_2 in V. Cliques found in the association graph using the Motzkin-Strauss formulation [19] correspond to subgraph isomorphisms between the original graphs U and V.

Since in our experiment we were searching for isomorphisms between induced subgraphs of the same graph (namely, the web site), we modified the construction of the association graph slightly to enforce level consistency — a web site can be thought of as a tree, and we checked that the two paths ended at the same level or depth of the tree. To accomplish this, we only add an edge to the association graph if the distances are the same *and* the two pairs of vertices are identical.

3 Data Collection

To test if the canonical set algorithm could be applied to find canonical behavior protocols, we collected data from users performing typical web-browsing tasks on a university web site [10]. The users were given a set of 32 questions covering a range of topics

Algorithm 2. Modified version of Pelillo's subgraph isomorphism algorithm

Given two graphs U and V, representing two paths taken by users through a web site:

1. Compute all-pairs shortest path distances for U and V [20].
2. Build an association graph G from the product graph of U and V:
   (a) Add vertex $\{u, v\}$ to G for every pair of vertices u in U and v in V.
   (b) Add edge between vertices $\{u_1, v_1\}$ and $\{u_2, v_2\}$ if:
   - the shortest path distance between vertices u_1 and u_2 in U is equal to the shortest path distance between vertices v_1 and v_2 in V, and
   - u_1 and u_2 refer to the same URL, and
   - v_1 and v_2 refer to the same URL.
3. Find a clique in G using the Motzkin-Strauss formulation [19].
4. The clique corresponds to a subgraph isomorphism between U and V.

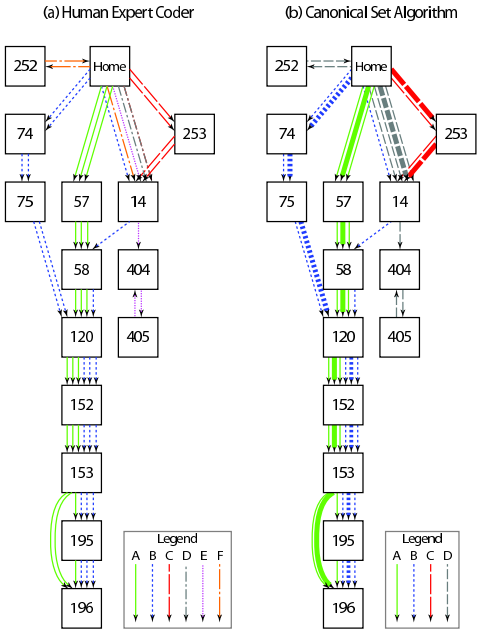

Fig. 2. Sample analysis graphs (from [10]). The canonical behaviors found by our algorithm are shown in bold in (b), and the other behaviors are labeled according to their nearest neighbor.

related to finding information about athletic programs, academic departments, and so on. We also required a "ground truth" against which to compare the canonical sets found by our method. For this purpose, we recruited two experts (a professor and an advanced graduate student) with significant experience in cognitive task analysis and modeling. We asked the experts, given the sequence of URLs visited by the users for each question, to identify subsets that they felt represented distinct behaviors. Clearly the experts could each have their own notion of what would constitute "similar" and "different" behaviors, analogous to the λ parameter in the canonical set algorithm. We left

this undefined and allowed them to use their own judgments to decide on the best partition for each question.

Figure 2 shows an example of the automated and expert results (from [10]) for an individual question ("What is the phone number of Professor ...?") to illustrate our analysis in detail. Each vertex represents a single web page (labeled with a unique integer) and each directed edge represents a clicked link from one page to another taken by one of the users. The expert (graph a) found 6 sets of behaviors: sets A and B represent different ways of clicking through the department web page to get to the professor's home page; sets C and D represent different ways of clicking through to the site's directory search page (vertex 14); and sets E and F represent slight variations on sets C and D. The canonical set algorithm (graph b, with λ=.36)) identified 4 canonical behaviors for this same question; these are shown in bold in the figure, and the other behaviors are labeled according to their nearest neighbor. The behaviors found by the algorithm correspond directly to the expert's sets A–D, but instead of splitting out sets E and F, the algorithm (in part due to the value of λ) grouped these variations with the nearest canonical set D.

4 Analysis and Results

To compare the performance of our graph-based similarity measure with our previous metric based on the edit distance, we used the well-known Rand index [21] to compare the clusterings found by the canonical set algorithm using each measure. Given a set of n elements $S = \{O_1, \ldots, O_n\}$, let $X = \{x_1, \ldots, x_r\}$ and $Y = \{y_1, \ldots, y_s\}$ represent two ways of partitioning S into r and s subsets, respectively. Then let a be the number of pairs of elements that are in the same partition in X and also in the same partition in Y; let b be the number of pairs in the same partition in X but in different partitions in Y; let c be the number of pairs in different partitions in X but in the same partition in Y; and let d be the number of pairs in different partitions in both X and Y. Then the Rand index is simply

$$R = \frac{a+d}{a+b+c+d} = \frac{a+d}{\binom{n}{2}} \quad . \tag{1}$$

We compared the performance of the edit distance and association graph measures across a wide range of possible values of λ. For each λ we computed the canonical sets for each question using both measures. We compared the resulting partitions with those found by our experts using the Rand index, and then computed the average value of R across all questions. The results are shown in Fig. 3. As the graphs illustrate, the association graph measure produced partitions that more closely matched both experts than our previous edit distance measure across nearly the entire range of λ values we tested.

The shapes of the graphs in Fig. 3 are somewhat surprising, as the curves might be expected to be concave with a peak at the actual λ used by each expert. There are several possible explanations for this. First, our canonical set algorithm is not symmetric with respect to values of λ. When λ is very high (above roughly 0.9 in this experiment) only one canonical pattern is found. However, the inverse is not the case: when λ is very low, the algorithm does not consider every element in the set to be canonical. Second, it is possible that our experts did not use a single λ in their evaluations, but rather varied

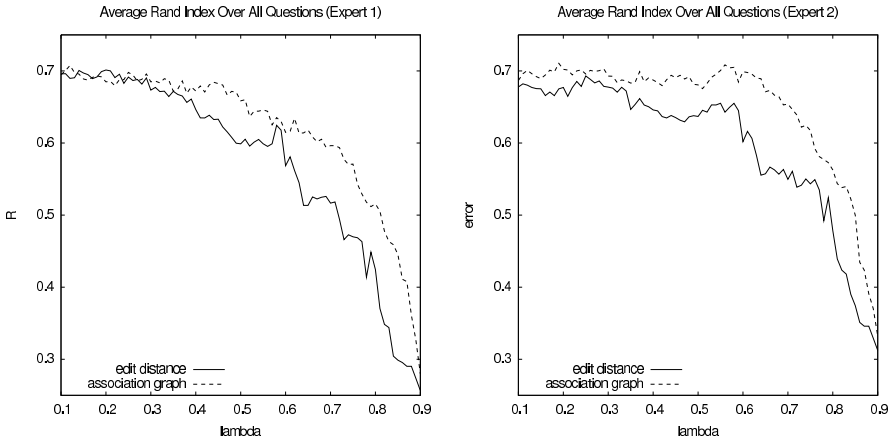

Fig. 3. Rand index comparison of edit distance and association graph similarity measures for the two experts across a range of λ values. The association graph measure outperforms edit distance across the nearly entire range for both experts.

their sense of "similar" and "different" behaviors depending on the particular behaviors they observed for each question. While the selection of the correct λ is beyond the scope of this paper, it is something we plan to study further in our future research.

5 Discussion

We have presented an automated method of finding canonical subsets of behavior protocols which uses a graph-based representation of the data. The collection of these types of time series is common in psychology and human factors research. While these data can often be naturally represented as graphs, there has been relatively little work in applying graph theory to their study. As users move through the space of possible behaviors in a system, their paths naturally induce a graph topology. We have shown that by taking into account this topology, improved results may be obtained over methods which ignore the underlying graph structure. We believe that this work is an important first step in the application of graph-based representations and algorithms to the analysis of human behavior protocols.

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

References

1. Card, S.K., Newell, A., Moran, T.P.: The Psychology of Human-Computer Interaction. Lawrence Erlbaum Associates, Hillsdale (1983)
2. Milson, R., Lewis, M.W., Anderson, J.R.: The Teacher's Apprentice Project: Building an Algebra Tutor. In: Artificial Intelligence and the Future of Testing, pp. 53–71. Lawrence Erlbaum Associates, Hillsdale (1990)

3. Salvucci, D.D.: Modeling driver behavior in a cognitive architecture. Human Factors 48(2), 362–380 (2006)
4. Chi, E.H., Rosien, A., Supattanasiri, G., Williams, A., Royer, C., Chow, C., Robles, E., Dalal, B., Chen, J., Cousins, S.: The Bloodhound project: automating discovery of web usability issues using the InfoScent™ simulator. In: CHI 2003: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 505–512. ACM, New York (2003)
5. Cutrell, E., Guan, Z.: What are you looking for? An eye-tracking study of information usage in web search. In: CHI 2007: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 407–416. ACM, New York (2007)
6. Ericsson, K.A., Simon, H.A.: Protocol analysis: verbal reports as data, Revised edn. MIT Press, Cambridge (1993)
7. Salvucci, D.D., Anderson, J.R.: Automated eye-movement protocol analysis. Human-Computer Interaction 16(1), 39–86 (2001)
8. Ritter, F.E., Larkin, J.H.: Developing process models as summaries of HCI action sequences. Human-Computer Interaction 9(3), 345–383 (1994)
9. Smith, J.B., Smith, D.K., Kupstas, E.: Automated protocol analysis. Human-Computer Interaction 8(2), 101–145 (1993)
10. Mankowski, W.C., Bogunovich, P., Shokoufandeh, A., Salvucci, D.D.: Finding canonical behaviors in user protocols. In: CHI 2009: Proceedings of the SIGCHI conference on Human factors in computing systems. ACM, New York (2009)
11. Garey, M.R., Johnson, D.S.: Computers and Intractibility: A Guide to the Theory of NP-Completeness. W.H. Freeman and Co., San Francisco (1979)
12. Denton, T., Shokoufandeh, A., Novatnack, J., Nishino, K.: Canonical subsets of image features. Computer Vision and Image Understanding 112(1), 55–66 (2008)
13. Toh, K., Todd, M., Tütüncü, R.: SDPT3 — a MATLAB software package for semidefinite programming. Optimization Methods and Software 11, 545–581 (1999)
14. Novatnack, J., Denton, T., Shokoufandeh, A., Bretzner, L.: Stable bounded canonical sets and image matching. In: Rangarajan, A., Vemuri, B.C., Yuille, A.L. (eds.) EMMCVPR 2005. LNCS, vol. 3757, pp. 316–331. Springer, Heidelberg (2005)
15. Kothari, J., Denton, T., Mancoridis, S., Shokoufandeh, A.: On computing the canonical features of software systems. In: Proceedings of the 13th Working Conference on Reverse Engineering (WCRE), pp. 93–102 (2006)
16. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. International Journal of Computer Vision 40(2), 99–121 (2000)
17. Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics Doklady 10, 707–710 (1966)
18. Pelillo, M., Siddiqi, K., Zucker, S.W.: Matching hierarchical structures using association graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(11), 1105–1120 (1999)
19. Motzkin, T., Strauss, E.: Maxima for graphs and a new proof of a theorem of Turan. Canadian Journal of Mathematics 17(4), 533–540 (1964)
20. Floyd, R.W.: Algorithm 97: Shortest path. Communications of the ACM 5(6), 345 (1962)
21. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66(336), 846–850 (1971)