Mining the Dataflow in Interactive Systems: Automatic Generation of User Behavior Patterns

Rafael Duque, José L. Montaña, and Cristina Tîrnăucă

Universidad de Cantabria, Santander, Spain
{rafael.duque, joseluis.montana, cristina.tirnauca}@unican.es

Abstract. Today's environments require computational interactive systems to exhibit abilities such as autonomy, adaptive and collaborative behavior, and inferential capability. Such abilities are based on the knowledge about users and their actions. To automatically develop an user behavior pattern an interactive system needs to analyze users' logs and recognize the tasks and the ultimate goals the users are trying to achieve. We propose a data mining approach to the problem of automatically generating user behavior patterns and group profiles in interactive computer systems. Our technical development has a general purpose and the only precondition is that we can observe and represent the actions performed by the user when interacting with the system by means of a finite alphabet of symbols.

1 Introduction

When an internet user visits a website, the IP address from which the connection is carried out, the geolocation, the pages visited, the time spent on each page, the links that were clicked and other information is recorded by some companies to create a *profile* which is associated with the IP address, Wi-Fi, geolocation or some other identifier associated with the visitor. This allows site publishers to use these data to create audience segments based upon users that have similar profiles. This is an illustrative example -coming from the field of behavioral targeting- in which *profiling* is an important task. In Information Science, *profiling* refers to the process of creation, use and application of patterns generated by computerized data analysis (see [11]). This involves the utilization of algorithms and formal techniques that allow to discover prototypes and/or correlations in data aggregated in data sets. When these patterns are used to represent people, they are called *profiles*. Other notions of the word *profiling* are used in Software Engineering and Computer Programming with a very different meaning. Models for generating dataflow models in computer systems have been under study for many years. Very often these models are represented by stochastic processes ([6], [7]) modeling the workload at low level components of the computer like the Ethernet layer or in a public wireless LAN as in [1]. Modern user behavior models try to catch the sequence of user interactions at a higher level. The goal of this kind of studies can be very different: in [13], a better understanding of the sources of video popularity through analysis of a number of internal and external factors is achieved, while in [10], user behavior profiling is used for intrusion detection. Cases of study in knowledge management can be found in [9], where an ontology-based framework for modeling user behavior is developed. Application or use of profiles to groups can be found in the cases of credit scoring, price discrimination, or identification of security risks (see [5], [3]).

In this paper we look for a low level representation of user profiles by means of Probabilistic Finite Automatons (PFA). We identify profiling with the process of construction and application of user behavior patterns generated by data analysis performed with machine learning techniques. This involves the use of data mining algorithms which allow to discover patterns in data sets. Our notion of profiling is not just about the construction of profiles, but also concerns the application of group profiles to individuals.

We divide the user profile task into three phases. First phase is to automatically generate a computational model of the user behavior that, in a broad sense, behaves as he does. This phase must be achieved by unobtrusive observation, recording the user interactions with the system. The result of this phase is what we call the *user model* (one for each user of the system). Second, based on the user models, we apply clustering techniques to partition the set of users into homogenous groups in which users have similar behavior. And lastly, we automatically generate a set of representative *profiles* taking as input the user groups coming from the second phase of the process. The output of this third phase is a list of finite state machines (PFAs), each of them representing one of the above mentioned groups. The whole process is summarized in Figure 1.

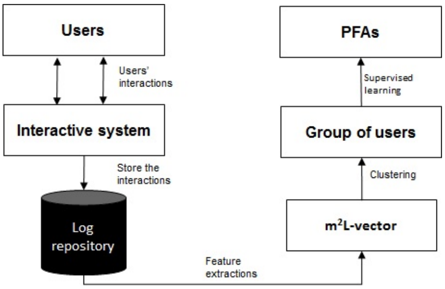


Fig. 1. The steps used to model the user behavior

2 Modeling the Users Behavior

The logs that record the users' interactions with a computer or mobile platform are the natural data source for user behavior modeling and for research experiments. These logs could actually be analyzed as a time series, but standard time series techniques do not perform well with discrete data. Markov models and, more generally, dynamical models representing stochastic process, seem to be more appropriate for our analysis. We propose in this paper a probabilistic approach for modeling the dataflow generated by (one or several) users interacting with a computer or a mobile system. The only assumption is that users interact with the application executing actions chosen from a finite set of possibilities.

2.1 Data Segmentation and Model Selection

The dataflow generated by all users of an interactive system can be seen as a stream of tuples $\mathcal{D}$ of the form (id, t, a), where id is the user's identifier in the system and t represents the instant time when the action a took place.

More precisely $\mathcal{D}$ is a sequence of the form:

$$\mathcal{D} := [(id_i, t_i, a_i)_i]_{1 \leq i \leq N}, \tag{1}$$

where $\mathcal{D}[i] := (id_i, t_i, a_i) \in \mathcal{ID} \times \mathcal{T} \times \mathcal{A}$; $\mathcal{ID}$ is a finite set of users' identifiers, $\mathcal{T}$ is a finite set of instants in which the process is observed and $\mathcal{A}$ is a finite set of actions. For convenience we denote by $\mathcal{D}[i].id = id_i$, that is, the id-component of tuple $\mathcal{D}[i]$. Analogously $\mathcal{D}[i].t$ and $\mathcal{D}[i].a$ represent (respectively) the t-component and the a-component of tuple $\mathcal{D}[i]$. We assume that $\mathcal{D}[i].t \leq \mathcal{D}[i+1].t$, for $1 \leq i < N-1$. The information contained in $\mathcal{D}$ can be segmented according to different criteria. If we are interested in the case in which $\mathcal{D}$ is partitioned according to the user identifiers, $\mathcal{ID} := \{ID_1, ..., ID_l\}$ each segment of information is of the form:

$$\mathcal{D}(k) := [(a_i, t_i) : \ \mathcal{D}[i].id = ID_k] = [(\ a_{i_1}, t_{i_1}), (a_{i_2}, t_{i_2}), \ldots, (a_{i_l}, t_{i_l})] \tag{2}$$

That is, segment $\mathcal{D}(k)$ is the subsequence of $\mathcal{D}$ formed by those tuples (a_i, t_i) in which action a_i is performed by the user whose identifier is ID_k (the user's id information is the same for all tuples in $\mathcal{D}(k)$ and thus it is omitted). We call segment $\mathcal{D}(k)$ a trace, (also *path* or *trajectory*) of user ID_k in the system.

For data analysis and user profiling, one might also be interested in the duration of transitions between actions, but not in the exact instant time in which actions take place. In such cases, we can rewrite the trace of an user as a new path of the form:

$$\pi : [(\ a_{i_1}, d_{i_1}), (a_{i_2}, d_{i_2}), \ldots, a_{i_l}] \tag{3}$$

were $d_{i_j} := t_{i_{j+1}} - t_{i_j}$ is the elapsed time between action a_{i_j} and action $a_{i_{j+1}}$, for $1 \leq j < l$. The elapsed time between actions is a continuous attribute. We can discretize the elapsed times into a fixed number of categories, say $\mathcal{C} := \{c_1, \ldots, c_L\}$ where each value c_i represents a time interval chosen *ad hoc* for the system under study.

2.2 Construction of User Models and Group Profiles using Finite States Machines

Suppose that $\mathcal{A} = \{a_1, a_2, ..., a_m\}$ is the set of actions that can be performed by an user in a given interactive computer system $\mathcal{S}$. For each user we build the path π as defined in Section 2.1 expression (3). That is, $\pi : [(a_{i_1}, d_{i_1}), (a_{i_2}, d_{i_2}), \ldots, a_{i_l}]$. In this path, actions a_{i_j} are in $\mathcal{A}$ and durations d_{i_j} belong to $\mathcal{C}$, where $\mathcal{C}$ is a finite set of time interval categories as explained in Section 2.1. Let L be the number of time categories.

We represent the information contained in a path using an m^2L-vector of natural numbers containing frequencies, $f := (f_1 \ldots, f_L)$, where

$$f_i = (f_i[a_1], \ldots, f_i[a_m]), \tag{4}$$

$$f_i[a_j] = (count_i(a_j, a_1), \ldots, count_i(a_j, a_m)), \tag{5}$$

for all $j \in \{1, \ldots, m\}$, and $count_i(a_j, a_l)$ is the number of times action a_j precedes action a_l in the trajectory of the respective user and the time elapsed between the two actions is in the time category $c_i \in \mathcal{C}$.

Next we are interested in clustering the users into a certain specified quantity of *representative* user profiles such that users in the same group (cluster) behave more similarly (in some precise sense) to each other than to those users in other clusters.

To this end we minimize the *within-cluster sum of squares* (WCSS) obtained by adding together the square of the Euclidean distance between each point f (representing an user's behavior) and its closest centroid:

$$WCSS = \sum_{j=1}^{k} \sum_{f \in S_j} \|f - q_j\|^2 \tag{6}$$

where $q_j = \frac{1}{|S_j|} \sum_{f \in S_j} f$ is the centroid of S_j and $(S_1, \ldots, S_k)$ is a partition of the set of users. For this purpose, we use the k-means algorithm, a heuristic method commonly employed that converges quickly to a local optimum (see [4]).

We denote by k the number of representative user profiles in an interactive system. The correct choice of k depends most of the times on the application. The optimal k will strike a balance between maximum compression of user profiles using a single cluster, and maximum accuracy by assigning each profile to its own cluster (having one cluster per user). If an appropriate value of k is not apparent from prior knowledge on the properties of the profile set, it must be somehow determined. In the literature, there are several proposals for making this decision effective. We use here an extended strategy known as the *elbow* *method* ([12]), which consists in plotting k against the WCSS and choosing the k at which the WCSS decreases abruptly.

Once k is chosen and the groups of similar users are identified, next step is to find a model that describes the profile of each group of users.

We propose to train a Probabilistic Finite Automaton (PFA) for each group. Formally, a PFA is a 5-tuple $\mathcal{M} = (\Sigma, Q, \phi, \iota, \gamma)$ where Σ is a finite alphabet (that is, a discrete set of symbols), Q is a finite collection of states, $\phi : Q \times \Sigma \times Q \longrightarrow [0, 1]$ is a function defining the transition probability (i.e., $\phi(q, a, q')$ is the probability of emission of symbol a while transitioning to state q' from state q), $\iota : Q \longrightarrow [0, 1]$ is the initial state probability function and $\gamma : Q \longrightarrow [0, 1]$ is the final state probability function (see [8]). In our case, the alphabet Σ of the automaton $\mathcal{M}$ is the set of duration categories $\mathcal{C} = \{c_1, \ldots, c_L\}$, while the state space $Q := \mathcal{A} = \{a_1, \ldots, a_m\}$ is the set of actions. Training the automaton $\mathcal{M}$ from a path π consists on determining the parameters $\lambda = (\iota, \phi, \gamma)$.

Following [2], we define ϕ by

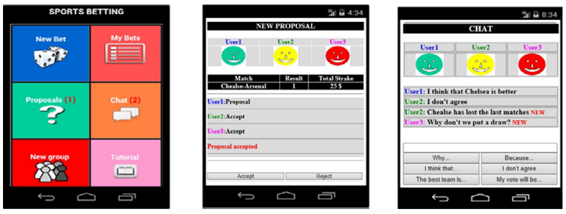
$$\phi(a_p, c_i, a_r) = \frac{count_i(a_p, a_r)}{\sum\limits_{1\leq j\leq L}\sum\limits_{1\leq q\leq m} count_j(a_p, a_q)} \tag{7}$$

for all p, r in $\{1, \ldots, m\}$ and for all $i \in \{1, \ldots, L\}$. Initial and final probabilities can be similarly defined, but they are not relevant in this case so we omit giving explicit formulas for ι and γ.

3 Case Study

A case study was carried out to put in action the proposal for modeling the users and groups behavior. This case study was designed to model the behavior of users of a mobile groupware system that supports collaborative sport betting. Figure 2 shows the main user interface of the system. The users build their own groups of users to collaborate in generating sport bets. This collaboration starts when a user accesses the *new bet* space, selects a sport event from a given list and proposes a certain bet as a prediction for the result of the game. Then, the members of the group analyze this proposal and use a voting panel (see Figure 2-center) to accept or reject the bet. If all the members accept the proposal, the bet becomes effective and is included in the *my bets* space of each user. Debates on a given proposal between users of the same group is allowed in the form of a structured chat (see Figure 2-right). Finally, the system includes other tools (see Figure 2-left) that enable users to manage the groups and to learn the system's features using a tutorial. This paper includes an appendix with the actions supported by this user interface of the groupware application (note that not all actions were used by users in our case study). Each action has an unique identifier.

Thirty-one users participated in this case study. They were randomly organized in a group of four users and nine groups of three users. Each group collaborated in five previously generated sport bets.



Match	Result	Total Strake
Chealse-Arsenal	1	25 $

User1:Proposal

User2:Accept

User3:Accept

Proposal accepted

User1: I think that Chelsea is better

User2: I don't agree

User2: Chealse has lost the last matches NEW

User3: Why don't we put a draw? NEW

Fig. 2. User interface of the groupware system

As described in Section 2.2, each user is represented by a vector of Lm^2 components, with $m = 12$ (only twelve actions were used by users) and $L = 4$ (we discretized the set of all time intervals into four quartiles). The precise four intervals are (time is expressed in seconds): $c_1 = [0, 46], c_2 = [47, 104], c_3 = [107, 295], c_4 = [297, 5251]$. Roughly speaking, c_1 corresponds to actions that take less than three quarters of a minute, c_2 corresponds to longer actions that take less than one minute and three quarters, c_3 corresponds to actions that take less than 5 minutes, and c_4 corresponds to actions that are longer than 5 minutes.

After running k-means (with 10000 different random initializations) and plotting the WCSS against the number of clusters, we obtained the following graphic (see Figure 3).

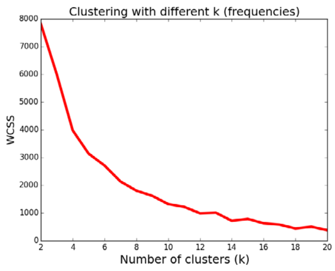


Fig. 3. WCSS for different values of k

The elbow method indicates $k = 4$ as an appropriate number of profiles. The four groups obtained contain one, two, ten and eighteen users, respectively. The four PFAs are represented in Figure 4. A description for each action is provided in the appendix. A transition from a state a_p to a state a_r labeled i/x has to be interpreted as $\phi(a_p, c_i, a_r) = x$. Multiple labels on the same transition are comma separated. Note that only transitions with at least 0.10 of probability are reflected in these graphical representations.

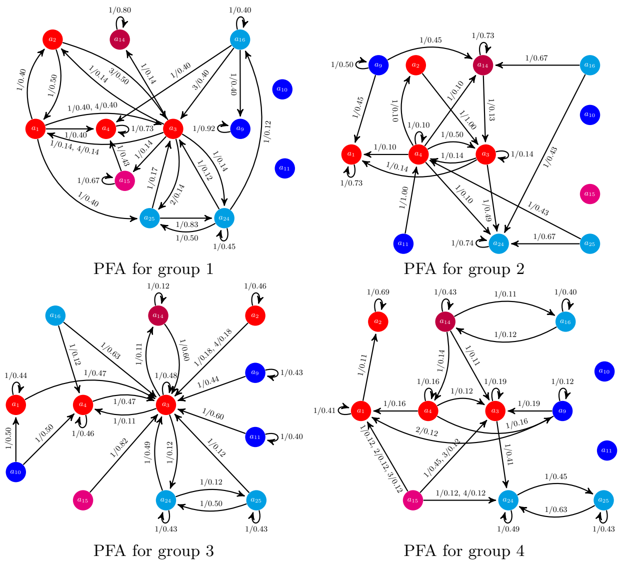


Fig. 4. The PFAs of the four profiles where the states are colored in red (action supported by the chat space), light blue (action supported by the new bet space), navy blue (action supported by the voting panel), magenta (action supported by the tutorial) and purple (action supported by the my bets space).

Table 1 summarizes some conclusions that can be drawn from these PFAs. Thus, one can better understand the differences between distinct user profiles and the degree to which the groupware system is adapted to the users' requirements. For instance, the level of usability provided by this groupware system is rather low since most of the users are unable to use it before accessing the tutorial (see action a_{14} in Figure 4): only the second group (which is made up of two users) did not take the tutorial before voting. Moreover, the voting panel is not an effective tool to decide the acceptance or rejection of the bets. This conclusion can be drawn from the fact that only the ten users of the third group perform the actions of this tool (see actions a_{10} and a_{11} in Figure 4). The users of the other groups usually only use the chat to communicate their preferences or disagreement with a bet proposal but they do not express their position in the voting panel. Also, it can be observed that only a reduced number of predefined messages of the chat are used (see actions a_1, a_2, a_3 and a_4 in Figure 4) and the other messages are not useful for them (for example, actions a_5, a_6, a_7 and a_8 do not appear at all in Figure 4).

Profile	Number of users	Description of the users
1	1	This profile includes only one user, which can be considered an outlier. This user does not use the voting panel. He has a reflexive attitude and spends several minutes in order to perform some actions.
2	2	These users can be also considered outliers (6% of the users). They are the only users that do not use the tutorial. These users do not use the voting panel and only take advantage of the chat to interact with other members of the group. They do not have a reflexive attitude and perform the actions quickly.
3	10	They are the only users who use actively the voting panel to accept and reject proposals. They do not have a reflexive attitude and perform the actions quickly.
4	18	These users do not use the voting panel and only take advantage of the chat to interact with other members of the group. They have a reflexive attitude and spend several minutes to perform some actions.

Table 1. Description of the profiles.

If we focus our attention on the time factor, it can be observed that users of the first and fourth profile have a more reflexive attitude and spend several minutes on performing some actions. For example, the users of the fourth profile can spend a considerable amount of time to start a new bet after seeing the tutorial (see the transition from action a_{15} to action a_{24} in Figure 4).

4 Conclusions

We have introduced a methodology for the automatic generation of patterns of user behavior in interactive systems that combines unsupervised and supervised learning. We record the logs of the users in the system in form of traces and, after a first preprocessing phase that consists in describing actions and durations, we compress these traces into feature vectors. Feature vectors are then clustered using the k-means algorithm (unsupervised phase) to group users by similarity of the empirical distributions of actions, durations and transitions. After this process is completed, we model each group of users by means of a probabilistic finite automaton (supervised phase). This finite state machine is what we call an user profile. User profiles, as intended in this paper, constitute a low level representation of user patterns in interactive systems. As future work we plan two ulterior developments: a) *From low level Profiling to Text Profiling*: develop a tool based on associative rules that automatically generates a text taking as input the low level user profile coming from our profiling methodology; b) *Inverse Software Engineering*: develop the methodology that can help software engineers to improve their system's design by using a finite state machine that is able to infer a task model from the interactive system under study.

Acknowledgments. This work was partially supported by project PAC::LFO (MTM2014-55262-P) of Ministerio de Economia y Competividad, Spain.

References

1. Balachandran, A., Voelker, G.M., Bahl, P., Rangan, P.V.: Characterizing user behavior and network performance in a public wireless lan. SIGMETRICS Perform. Eval. Rev. 30(1), 195–205 (Jun 2002)
2. Dupont, P., Denis, F., Esposito, Y.: Links between probabilistic automata and hidden markov models: probability distributions, learning models and induction algorithms. Pattern Recognition 38(9), 1349–1371 (2005)
3. Elmer, G.: Profiling Machines. Mapping the Personal Information Economy. MIT Press (2004)
4. Everitt, B.S., Landau, S., Leese, M., Stahl, D.: Cluster Analysis. John Wiley & Sons, 5th edition edn. (2011)
5. Hildebrandt, M., Gutwirth, S.: Profiling the European Citizen. Cross Disciplinary Perspectives, Springer (2008)
6. Hlavacs H., K.G., C., S.: Traffic source modeling. Technical Report TR-99101, Institute for Applied Computer Science and Information Systems (1999)
7. Jagerman, D.L., Melamed, B., Willinger, W.: Frontiers in queueing. chap. Stochastic Modeling of Traffic Processes, pp. 271–320. CRC Press, Inc., Boca Raton, FL, USA (1997)
8. Rabin, M.O.: Probabilistic automata. Information and Control 6(3), 230–245 (1963)
9. Razmerita, L.: An ontology-based framework for modeling user behavior?a case study in knowledge management. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 41(4), 772–783 (2011)

10. Salem, M.B., Stolfo, S.J.: Modeling user search behavior for masquerade detection. In: Proceedings of the 14th International Conference on Recent Advances in Intrusion Detection. pp. 181–200. RAID'11, Springer-Verlag, Berlin, Heidelberg (2011)
11. Stock Wolfgang, G, S.M.: Handbook of Information Science. De Gruyter Saur (2015)
12. Tibshari, R., Walter, G., Hastie, T.: Estimating the numbre of clusters ina data set via the gap statistic. J. R. Statist. Soc. B 63, 411–423 (2001)
13. Yu, H., Zheng, D., Zhao, B.Y., Zheng, W.: Understanding user behavior in large-scale video-on-demand systems. SIGOPS Oper. Syst. Rev. 40(4), 333–344 (Apr 2006)

Appendix

The following table describes the semantics of the twenty-five actions of the interactive groupware system used for the experimentation.

Identifier	Space or tool that supports the action	Description of the action
a_1	Structured chat	Access to this space
a_2	Structured chat	Send a "free" message
a_3	Structured chat	Send a "Why..." message
a_4	Structured chat	Send a "Because" message
a_5	Structured chat	Send a "I think that" message
a_6	Structured chat	Send a "I don't agree" message
a_7	Structured chat	Send a "The beast team is" message
a_8	Structured chat	Send a "My vote will be" message
a_9	Voting panel	Access to this space
a_{10}	Voting panel	Reject a proposal
a_{11}	Voting panel	Accept a proposal
a_{12}	Voting panel	A proposal was accepted by the group
a_{13}	Voting panel	A proposal was rejected by the group
a_{14}	My Bets	Access to this space
a_{15}	Tutorial	Access to this space
a_{16}	New Bet	See sport events
a_{17}	New Group	New Name
a_{18}	New Group	Add a person
a_{19}	New Group	Select a group
a_{20}	New Group	Rename a group
a_{21}	New Group	Send an invitation
a_{22}	New Group	Accept an invitation
a_{23}	New Group	Reject an invitation
a_{24}	New Bet	Access to this space
a_{25}	New Bet	Send a proposal

Table 2. Actions of the groupware system.