Extracting Realistic User Behavior Models

Reiner Jung
rju@informatik.uni-kiel.de
Kiel University, Germany

Marc Adolf
mad@informatik.uni-kiel.de
Kiel University, Germany

Abstract

Workloads play a central role in assessing software qualities, like performance and privacy. They are characterized by intensity and user behavior patterns. Combining multiple intensities and behaviors are used to create workload profiles which, among others, sevaluate software design, predict of system utilization. The central challenge for workload profiles is their fit to real workloads and in particular the match to specific behaviors. This is especially relevant for understanding and identifying specific user groups and support workload composition by operators.

In this paper, we address the identification of such realistic user behaviors by utilizing domain specific attributes, report on our evaluation of the fitness of behavior clustering approaches, and discuss our setup to evaluate further clustering approaches.

1 Introduction

Service quality of software systems is influenced by workload intensity and the user behavior. Both factors play a vital role characterizing the system workload [7], which is relevant to understand past workloads and construct workload profiles to estimate future system utilization and performance. For example, the resource consumption of browsing a catalog, searching the inventory, and purchasing items can be quite different. Therefore, it is necessary to be able to distinguish specific kinds of user behavior to characterize the workload sufficiently.

State of the art workload characterization approaches, such as WESSBAS [8], use a behavior mix, where different workload intensities are combined with specific user behavior models to construct a workload model. These approaches collect user sessions and aggregate them to behavior models. WESSBAS estimates behavior models utilizing X-means clustering [3]. Such behavior models have three key shortcomings: (I) They reflect the observed behavior of the past, but might not represent specific user groups correctly harming predictability. For example, a *detergent shopper* might reappear frequently while a *sunscreen shopper* has a different seasonal profile. Unfortunately, current approaches cannot distinguish between them. (II) The behavior models use cyclic graphs with edge counts or probabilities to create compact representations. However, in this process we may lose behavioral information, as two visits of a page might differ in purpose and a looping behavior might actually be dissimilar than another loop along the same pages. (III) X-means only yields acceptable results for small parameter vectors of at least ordinal values, but current behavior models are mapped to vectors. More model transitions imply more parameters, which harm clustering [3].

To mitigate these issues, we want (a) to advance behavior models to better capture the specific properties of different kinds of users based on domain knowledge of the observed software system and other parameters, e.g., time, and (b) to improve clustering and classification of observed user behaviors. Therefore, we extend classic behavior models with domain knowledge and evaluate different aggregation approaches capable of handling large parameter sets or find ways to reduce the parameter sets to be able to use aggregation approaches well suited for limited number of parameters.

In this paper, we report on our preliminary findings regarding two clustering approaches X-means and Expectation-Maximization (EM) [2] in context of realistic user behaviors, present additional approaches which we are currently investigating, and formulate key questions regarding the identification behavior models and their quality.

The paper is structured as follows: Section 2 discusses user behavior models. Section 3 introduces clustering and aggregation approaches. Section 4 presents the concept of realistic user behaviors. Section 5 describes the evaluation and Section 6 discusses preliminary results for X-means and EM clustering. Finally, Section 7 summarizes our findings, discusses further research, and describes our key questions.

2 Behavior Models

Behavior models describe kinds of users and are aggregations of single user behaviors with similar behavior patterns. A single user behavior comprises all system invocations (entry level events) of a user during a session. It can be modeled as a path over visited pages or transformed into a behavior graph or a Markov-chain, which may contain loops for repetitive behavior. These paths or graphs are then grouped for similarity and merged into a behavior model.

Figure 1 depicts an annotated behavior graph of a user interacting with the JPetStore [12], an exam-

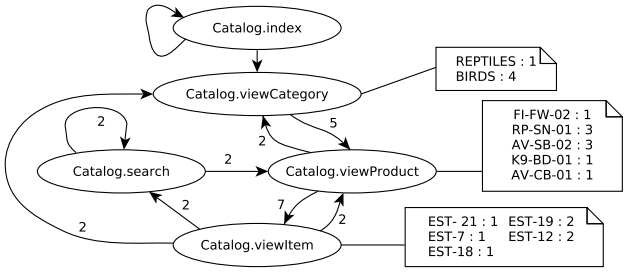

Figure 1: Behavior graph representing a shopper from our JPetStore [12] use case.

ple application resembling a shop system for pets. In this graph, nodes represent page visits and edges express the transitions between pages. The numbers at the edges indicate the amount of transitions between pages. In addition, we added domain specific information, like the viewed category and product, which can be used to support the behavior clustering.

3 Clustering Approaches

Clustering can be used to identify groups of data points which share similarities. In our context, we use clustering to identify user behavior models, like WESSBAS, which uses X-means. Clustering is affected by density, distances, and distribution of data points. Depending on the clustering approach, the dimension of the data points can have a significant impact on the quality of the clusters. We employ clustering methods provided by Weka [4]:

X-Means X-means builds on the K-means clustering algorithm [3], which consists of three steps [1]: (1) For every expected cluster (K), a center point, called centroid, is randomly chosen from the data points. (2) According to a chosen distance metric, each point x of the data set is assigned to the closest centroid. (3) The centroids are recomputed according to the center of mass of the points belonging to it. (2) and (3) are repeated until a convergence criterion is met.

In contrast to K-means, X-means searches over a range (e.g., 2 to 10) for a set of clusters, which provide the best fit. Therefore, X-means starts with computing K-means for the lower bound (e.g., K=2). Subsequently, each cluster is split into two using 2-means to try to improve the fit. Both steps are iterated while incrementing K until the upper bound is reached or an iteration is worse than the one before [3].

Expectation-Maximization EM is an iterative method consisting of two phases (E- and M-step) that are repeated until the convergence criteria is met and a final set of clusters is identified. Initially, a random set of cluster identifying data points are defined which are the initial parameters for EM. The *E-step* uses these parameters to compute the expected values of each data point. The *M-step* uses the E-step results to compute a new maximum likelihood for the data points regarding the parameters. This way, the new parameters for the next iteration are computed. These two steps are repeated until the convergence criterion is reached [2].

Hierarchical Clustering Hierarchical clustering is a approach which builds a hierarchy of clusters. Where the root cluster contains all individuals which are then further divided into smaller clusters. The hierarchy can either be build up from the root cluster to the leaves (divisive) or vise versa (agglomerative). With the agglomerate approach, we start with clusters containing each only one individual. Then we determine the distance between all pairs of clusters. The pair which has the shortest distance is then merged into one cluster. This process is repeated until all individuals are merged into one single cluster [4, p. 95]. The shortest distance between two elements is determined by a distance function, like Euclidian distance or Manhattan distance.

Similarity Matching In contrast to the other approaches, similarity matching uses two metrics to compare graphs based on structural similarity and on the distance of parameter values based on their semantic similarity. The algorithm computes initially the distance between each graph of a set of behavior graphs, creating a vector for each graph containing the distances to all others graphs. The distance between two vectors is the sum of differences $\sum_{i=1}^{n} |d_{a,i} - d_{b,i}|$ where n is the number of graphs, and $d_{j,i}$ refers to the values in a vector j. Graphs where the distance is lower than a defined threshold, are then considered similar and grouped together.

These groups are further divided based on their semantic difference. For example, in JPetStore categories, products, and items form a tree. The distance between two values in the tree determines their semantic difference, e.g., two cats Fritz and Felix belong both to the product male cartoon cat, having a distance of one, while the cat Amber belong to the product female cat, and therefore the distance is two.

4 Realistic User Behaviors

We define realistic user behavior models as behavior models which reflect real groups of users in contrast to approximated groups, i.e., groups solely defined by their transitions, neglecting domain-specific data. For example, our detergent shoppers should form a separate group from those buying sunscreen. This is helpful to better understand seasonal behavioral changes and allow to create and modify workloads more realistically. This is relevant in scenarios, where workload characterizations can be modified to provide the system with knowledge of upcoming events, like a sunscreen shopper just before the holiday season.

A key ingredient for realistic user behaviors is domain-specific data, like the products or categories. With this additional data, user behavior can be classified in different groups. To be able to use such values

in a clustering approach, a suitable metric must be defined, e.g., products of similar type should be closer together than products which are in another category.

5 Evaluation

Our evaluation uses the iObserve analysis service with different aggregation filters to examine clustering and aggregation approaches. Figure 2 depicts an excerpt of the pipe and filter setup for our analysis. The *Session Collector* collects *EntryEvent*s caused by a users and creates sessions (collection of *EntryEvent*s). The filter sends out a session event either when it receives a *SessionEndEvent* or a timeout is reached and the filter is triggered by the time trigger filter. The *AnnotatedGraphBuilder* creates from the *SessionEvent* an annotated graph and sends it to an *AggregationFilter*. Depending on the aggregation and clustering approach, a specific filter is inserted there. Finally, the aggregated graphs are serialized with the *GraphOutputFilter* or alternatively transformed into a behavior model suitable for the Palladio Component Model.

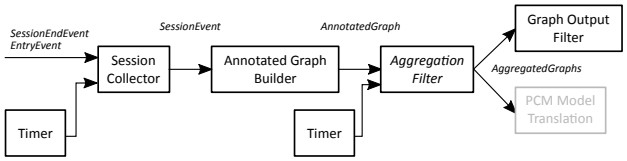

Figure 2: Analysis Pipe-and-Filter setup for user behavior aggregation

The input for this analysis is provided by two software systems. The first observed systems is an JPetStore [12] instance instrumented with Kieker [5]. The second one is an instrumented instance of our research group's JIRA [11] which is used by students during a four week practical course. We use the JPetStore to evaluate whether a specific previously defined setup of realistic behaviors can be detected. While the JIRA experiment is used to apply the approaches to another domain where we want to explore whether they can produce reasonable results in a realistic scenario.

5.1 JPetStore Experiment

For the JPetStore experiments, we modeled seven realistic (ideal) user behaviors [9], which utilize all functions of the JPetStore. We created workloads with Selenium [13] that represent these behavior models. We execute JPetStore together with our workload and collected monitoring data. This data is then processed by the iObserve analysis [6] using different clustering algorithms provided by their respective filters. Then, we compare the detected behaviors with the mentioned set of seven ideal behavior models (IBM).

Workload The behaviors are tailored to share common behavior, but also include significant differences regarding pages, transitions, and request parameters, e.g., whether the person shops cats or fishes:

Account manager (AM) Changes contact information after login. Inspects one of the prior orders.
Browsing user (BU) Searches products and only browses categories, products, and items.
Product lover (*L) Visits the CATS (CL) or FISH (FL) category and selects one product. Repeats 8 times and concludes shopping.
Single product buyer (S*) Goes to a category (REPTILES (SR) or CATS (SC)) and buys one item.
New customer (NC) Registers as a new customer, logs in, and buys a reptile.

Experiment Execution At the end of each analysis run, we compare the detected clustered behavior models with the prepared IBMs. First, we identified which detected behavior model matches best to an IBM. Second, we identified the distance of the matching model. In case a match can be identified, this counts as a hit (score=0). The match between two models is computed in three steps: (1) We remove nodes which are not connected to the behavior graph, as they are created by mapping graphs to matrices and back. (2) We identify missing and additional nodes and edges, and compute ratios between these differences and the IBM. The lower the ratios, the better the fit of the detected behavior model. (3) We compare the request parameters in the behavior models. For example, in the IBM the parameter CATS appears once, but the detected behavior model includes REPTILES, then the behaviors do not match.

5.2 JIRA Experiment

The JIRA experiment utilizes real word monitoring data which we collect every semester during a four week practical course where multiple groups of students plan and develop a small software system. Therefore, we do not have predefined IBMs for this experiment. However, we want to detect and isolate specific behaviors. To examine which computed behavior models match the reality, we will discuss the aggregated behavior models with the students. Beside this qualitative evaluation, we also will gain insight in how students use JIRA and whether we have to introduce the functionality differently.

6 Preliminary Results

We already evaluated X-means and EM clustering. The X-means setup is based on preliminary work, where we tested different configuration parameters for the algorithm [9]. We choose a configuration for X-means which provided the best fit to the JPetStore scenario. We set the range for the number of expected clusters to [6..12] and use the Manhattan Metric.

For this clustering, we decided to go with the standard setting in Weka and are not setting any parameters, including the pre-estimated number of clusters. Since both algorithms start with randomly chosen values, the results may differ between each execution.

Table 1: Comparison between the generated clusters and the IBMs (scores and content).

Scores	AM	BU	CL	FL	SC	SR	NC
EM	0.25	0	0	0	0.8	0.8	0.9
X-means	0.25	0	0	0	0.1	0.1	0.4
Content	AM	BU	CL	FL	SC	SR	NC
EM		+	+	+	(a)	(c)	(e)
X-means		+	+	+	(b)	(d)	(f)

Therefore, we execute each clustering five times to avoid results solely based on arbitrary starting values. In X-means, the resulting user behaviors are the computed centroids of each cluster. They do not necessarily correspond to a real behavior. In contrast, the EM clustering only groups measured behaviors. In this evaluation, we simply took one representative behavior of each group. This can be improved, e.g., by creating a mean vector of every cluster.

Both approaches could not detect all 7 IBMs (EM=4 and X-means=5 clusters), but some of the detected models match an IBM. Table 1 depicts scores and parameter matches of behaviors to IBMs.

EM and X-means both detected the *account manager* behavior, but there where minor discrepancies between the aggregations and IBMs. As we did not record parameters for these pages, we could not compare the behaviors content wise. The *browsing user*, *cat lover*, and *fish lover* where detected correctly by both algorithms. The *single cat buyer*, however, could not be detected by EM, the closest match was the *single reptile buyer* behavior (a). X-means created a merged cluster of cat and reptile buyer, and the new customer, identifiably by 1/3 possibility for a cat and 2/3 for reptiles (b). Similarly, the *single reptile buyer* was identified by EM (c) and X-means closest match was the same as for the *single cat buyer* (d). Finally, the *new customer* detection failed, as the returned cluster deviated significantly from the IBM. Also the found graph better matches a single buyer or product lover than the new customer.

7 Conclusion

We presented our efforts towards realistic user behavior clustering to improve the understanding of workloads and their composition, which can improve software quality assessment and support more precise workload alterations. All experiment data, notes, and artifacts can be found in our replication package [10].

We reported on the detection quality of the two clustering approaches EM and X-means for behavior models. Our current findings are that both clustering approaches are able to differentiate some behaviors based on parameter information, which is an improvement in comparison to the clustering without this information. However, they are unable to detect all behaviors correctly. Key issues are for X-means larger vectors resulting in less precise clustering [3], and for EM the behavior model merge might be amendable.

In future, we will evaluate further aggregation algorithms, including hierarchical clustering and similarity matching. We want to include other factors, like, seasonal factors, into the clustering to improve detection. In context of the EMLS'18, our key questions are: (a) Are these specific approaches useful to improve the quality of aggregated behaviors? (b) Are cyclic graphs the best possible way to describe realistic user behaviors? (c) How can we improve our evaluation, especially in scenarios where we cannot predefine ideal behavior models?

References

[1] J. B. MacQueen. "Some Methods for Classification and Analysis of MultiVariate Observations". In: *5th Berkeley Symp. on Math. Statistics and Probability*. Vol. 1. UC Press, 1967.

[2] T. K. Moon. "The expectation-maximization algorithm". In: *IEEE Signal Process. Mag.* 13.6 (Nov. 1996).

[3] D. Pelleg et al. "X-means: Extending K-means with Efficient Estimation of the Number of Clusters." In: *ICML*. Vol. 1. 2000.

[4] M. Hall et al. "The WEKA Data Mining Software: An Update". In: *SIGKDD Explor. Newsl.* 11.1 (Nov. 2009).

[5] A. v. Hoorn et al. "Kieker: A Framework for Application Performance Monitoring and Dynamic Software Analysis". In: *Proceedings of ICPE 2012*. ACM, Apr. 2012.

[6] R. Heinrich et al. "Architectural Run-Time Models for Operator-in-the-Loop Adaptation of Cloud Applications". In: *Proceedings of MESOCA*. IEEE Computer Society, Sept. 2015.

[7] M. C. Calzarossa et al. "Workload Characterization: A Survey Revisited". In: *CSUR* 48.3 (Feb. 2016).

[8] C. Vögele et al. "WESSBAS: extraction of probabilistic workload specifications for load testing and performance prediction—a model-driven approach for session-based application systems". In: *SoSyM* (Oct. 2016).

[9] C. Dornieden. "Knowledge-Driven User Behavior Model Extraction for iObserve". Master thesis. Kiel University, June 2017.

[10] *Replication package*. `doi . org / 10 . 5281 / zenodo.883095`. 2017.

[11] *JIRA – Issue Tracking System*. `https://de.atlassian.com/software/jira`. 2017.

[12] *MyBatis JPetStore application*. `http://www.mybatis.org/spring/sample.html`. 2017.

[13] *Selenium browser automation*. `http://www.seleniumhq.org`. 2017.