User Experience Personalization within software applications: A Data Mining approach

Vlad-Mihai Dănilă
Technical University of Cluj-Napoca, Computer Science Department
Cluj-Napoca, Romania
vlad96mihai@gmail.com

Teodor Ștefănuț
Technical University of Cluj-Napoca, Computer Science Department
Cluj-Napoca, Romania
teodor.stefanut@cs.utcluj.ro

ABSTRACT
Developing software products that provide meaningful, relevant and user-targeted experiences to users has lately become a trend. This paper proposes a centralized stand-alone system that yields personalized user experience to a target application under logs analysis context. We introduce a real-time Sequential Pattern Mining approach to gather knowledge about on-going user's habits. The modules that comprise a personalization system are introduced, emphasizing the usage mining module. Solutions and limitations on integrating personalization aspects in a to-be-personalized environment will be presented as well. The applicability of the personalized gathered knowledge will be exemplified in the context of a Web application used to manage and configure products in a company.

Author Keywords
User Experience Personalization; Top-K Sequential Pattern Mining; Logs analysis; Web application;

ACM Classification Keywords
H.5.2: User Interfaces

INTRODUCTION

The problem of user's experience within large software applications becomes more and more important as there is a worldwide increase in complexity of working environments. Traditional software applications are focused on the final deliverables that they were built for, often surpassing the entire journey aspects needed for its users to end up there. An ideal scenario would comprehend knowledge infused in standard solutions that is capable of collecting data surrounding the user and optimizing the human-made effort. By discerning who the users are and what they are targeting for, actual changes can be made to the system to efficiently manage resources, save time and increase usability.

Given the broad variety of software environments that comprehend a considerable mixture of information passing, a personalization system that could be integrated regardless particularities and context differences between systems requires logs format compliance and real-time communication channel. If the real-time behavior allows some constraint fluctuations (i.e. batch of logs could be delivered at a certain frequency), the logs structure needs to obey a standardized structure, where features would be the same for each target application.

Gathering logs that are of the format (user_id, action, timestamp) heightens pattern discovery tasks on the actions that mark each uniquely identified user. Such tasks are generally covered by Data Mining applications.

Knowledge gathered by means of useful patterns mining stands as a flexible and influential resource that can be delivered and further used inside other environments so to enhance their usefulness, usability and accessibility.

User's journey prevails within a personalized software application, characterized by centering its behavior on user needs. This behavior can be defined as a sequence of actions which mark human interaction with the application. Given historical information on how the application is used, most used sequences of actions can be exploited.

Users' learning process within the application should be improved and backed when the personalization feature is operating. Within a working environment, users are in continuous learning process, which will not be diminished or canceled when bringing changes, as result of personalization. This entails carefully adjustments that should not overwhelm user's existing knowledge nor disturb or abolish habits already established.

The rest of the paper is organized as follows. Next section presents related work. The following section describes the Data Mining approach used to obtain the system's usage tendencies. Next, an experimental application with direct personalization integration is described, followed by an evaluation section. Finally, last section presents the conclusions and potential future extensions and improvements.

RELATED WORK

In this paper, TKS algorithm [1] is proposed as solution to identify the sequence of actions that best characterizes an individual. A stand-alone system will process usage logs and will expose most used workflows to systems to-be-personalized, which, in their return, expose usage logs whenever updates occur.

In literature, there are several works that use Data Mining approaches to analyze user behavior, as well as analyses on

how user's journey within an application can be improved by targeting User Experience personalization aspects. However, there is a lack of binding alternatives for managing the bridge connection and integration within a unitary structure of an environment build upon integration of analyzed laterals.

A unitary system that holds both areas targeted in this work, User Experience and Data Mining, brings context counterparts mainly due to extensive sphere of fields that both are activating in. Nonetheless, assimilating both as parts of a single system involves User Experience feed data as Data Mining information output. Existing solutions will be discussed for both Data Analysis and User Experience sides, beginning with Data Analysis discussion.

The Data Analysis output comes as harvesting long-time developed and improved solutions, which, however, should be pruned given our User Experience personalization context. Mining techniques have no direct binding or field targeting personalization aspects; hence variants need to be identified and adjusted. Also, before any technique could make sense to a personalization context, amount of processing required to deliver mined knowledge should be seen as a top priority since generally, real-time behavior is targeted. [2]

The Data Mining field occurred in 1990 with Agrawal and Srikant paper [3] which introduced the Apriori algorithm for discovering frequent itemsets in a database. Up until then, several pattern mining techniques were developed, such as frequent itemset mining [4] [5] [6] or association rule mining [3]. These techniques, however do not take into consideration the sequential ordering of events, which could be the source of valuable information in the personalization context. [7]

Sequential Pattern Mining overcomes this lack of sequential ordering of time-based itemsets [8] [9] [10] [11] and was initially proposed by Agrawal and Srikant. [12]

Personalization is achieved by following two main approaches: collaborative filtering and content-based filtering.

Collaborative filtering is the most popular technique to provide personalized recommendations [13] [14] in which clusters of users are having their behavior observed. Ringo is a music recommendation prototype based on this approach [14], where each user is requested to provide ratings to sets of music and based on these ratings, groups of users are identified and are receiving relevant recommendations.

There are two problems with this approach. To begin with, a large number of users need to participate in the rating process before the recommendation starts to make sense. User Experience within the system increases due to personalization but on the other side decreases due to necessity of direct input from the user to make it happen. An intuitive solution for this would be not to explicitly require direct user effort to gather knowledge, but develop techniques to build that knowledge in the background. For example, Spotify, a digital music service that makes extensive use of personalized recommendations, injected the user behavior understanding process in actions that already exist and could themselves provide valuable information. If a musical piece was completely listened to, there's very likely that the user prefers it or if it has been listened to more than a couple of times, the probability for the user to listen to it again is very high. On the other hand, if the piece was skipped immediately after it began, it is probably not matching the user's preferences.

The second problem of collaborative filtering is that rating schemes can naturally be applied only to homogeneous types of simple products. Sparse data cannot be considered for such a filtering, since intersection between two different users would be zero and so would be the dot product calculations in the similarity matrix, which would also raise considerably scalability problems, for large number of users. For example, in a store selling both PCs and DVDs, there are differences between cost, purpose and characteristics of these products. It would be problematic to determine the closeness of the persons buying these products, since many aspects of the items can be considered and a single number conveying a person's taste would not be relevant. A low rating might be given by a person because of the color used on the PCs case, and a low rating as well from another person who likes the color, but dislikes the computation power.

Content based filtering goes one step forward and analyzes the content of the items preferred by a person, and identifies items with similar one [15] [16]. This overcomes collaborative filtering sparsity problem, since recommendations rely on the characteristics of objects themselves. It also has several other improvements:

- Cold-start is avoided, since far less initial input than the one needed for collaborative filtering is required and reliable early recommendation can be delivered.
- New objects do not require users to interact with them before the system starts recommending them.

Two major problems can be identified in the content-based filtering approach for personalized recommendations.

First problem represents limitation brought in personalized items, raised by the tendency to narrowly focus and recommend only items that score highly against user's profile or past behavior [17].

Another problem consists in the way the content analysis for each item should be handled. Straight forward potential dimensions, like strategy, adventure, shooter dimensions could be easily assigned to video games [18]. Each game can be rated based on these dimensions, where the resulting rating vector will serve as content descriptor or state vector of the game. However, for more ambiguous items like an autonomous vacuum, significant effort must be invested in rating across to-be-discovered dimensions by means of developed feature extraction techniques.

Hybrid approaches that combine collaborative and content-based filtering exist, where clusters of users are formed based on content filtering. An example of such a hybrid system is Intelligent Recommendation Analyzer, developed at IBM T.J. Watson Research Center [17]. IRA proposes a personalization engine for E-commerce, allowing usage of different methods based on products type, while also being able to take into account revenue objectives like number of exposers vs. number of clicks or a mixture of the two. [19]

Clustering is another data mining approach that proposes user's segmentation in database records. Trend analysis and pattern recognition are targeted as well, generally in database targeted areas and statistics ones too. [20] [21] [22]

Closeness between users is identifiable by mapping user's interests on feature vectors. Large number of items brings scalability concerns and sparsity issues as well, considering traditional feature selection solutions [23], which prune away irrelevant dimensions. However, selecting certain features in advance can lead to loss of information [17], action which itself require knowledge on features to pe pruned.

Clustering approach is feasible in a general personalization system until a point highly achievable in early stages and difficult to improve based on clustering approaches only. In an ideal situation, where clustering was successfully achieved with high accuracy, individual users predicted personalized features would rely on cluster-belonging users' contribution, which even achieved with desired accuracy, might not deliver the same accuracy on individual targeted results.

Association rule mining is another approach that can be used to identify items which are often purchased together and to analyze users' profiles in order to directly target actions towards them, like promotions. Initially introduced in the context of supermarket data where studies on relationships between bought items and buying patterns were analyzed [22], the concept itself can be adapted to different scenarios as well.

Also called Market Basket Analysis, Relationship Mining or Affinity Analysis, Association Rules come with a huge impediment for our needs. If we come up with a rule, like milk and bread are usually bought together, we define a kind of generic strategy which is applicable to all customers [18], which is very different from a personalized recommendation system, where the items that are recommended to one person are not necessarily the same with the ones recommended to another person. One-rule-person concept cannot be achieved using association rules, given the generic, applicable to all user's characteristic.

Classification problems are targeting personalization aspects in databases and Artificial Intelligence communities [17]. Classification solutions, given the context of data mining, assume labeled data that might need significant human expertise, aspect often not reliable in a log analysis context. Given a supervised context, with available data that suits building-a-model required needs, searching and processing in high scale data by means of classification enhancements brings improvements regarding performance and real-time capabilities additions. Yet, in an unsupervised context, machine learning techniques like Self Organizing Maps are to be considered, but this is out of our scope.

Data mining comprehend various types of fields and approaches that are not applicable as direct personalization variants need to be analyzed, since their usefulness could map on personalization aspects.

User's interaction with a system can be seen as a sequence of actions that builds an interaction model. This interaction model has predisposed tendency towards high volume of data and extensive required computation power. Given the real-time requirement regarding personalization system's applicability effects, efficient solution need to be considered such that computations provide an outcome in viable time.

User Experience tasks tend to refer their solutions to the phrase "User is always right" [24]. The reasons for investigating UX are, as stated in [25] related to well-being increase [26], users' lives improvement [27], failure prevention and reduction [28], brand stimulation [29], or customer preference understanding by taking into account affective reactions. [26] [30]

Current literature on UX is widening with more and more theories and frameworks [31] [32]. The authors of [25] describe experience as being dynamic, always evolving and cumulative [26] [32], underlying its nature to permanently enrich and predisposition to change.

This personalization enrichment is an evolvement in knowledge, actions and results that adapt to human change.

The most popular example of personalized recommendation system is Amazon.com [33]. The system holds past purchases and analyzes them in order to bring to front similar items that users might be interested in.

Commercial websites have affiliations to marketing companies such as DoubleClick Inc. [33], which monitors users' activities by collecting cookies based on which users' profiles are built. As for standalone applications, collecting and managing data from other sources can become problematic.

We propose a system that is capable of delivering personalized features taking into account only a log component which is part of most software solutions.

SYSTEM OVERVIEW

The system that we propose is a client-server application. The server is the core of our system, containing Data Mining approaches used to analyze logs. We will use Sequential Pattern Mining Framework (SPMF) [34], which is an open-source data mining library, written in Java.

Client applications mark environments considered for personalization. For our purpose, an Angular based Web application will be developed, but this is not a constraint, just a demonstration. Any application that has a logging component and is able to process JSON responses received from the system's core would suit in the system. *Figure 1* bellow depicts the system's architecture.

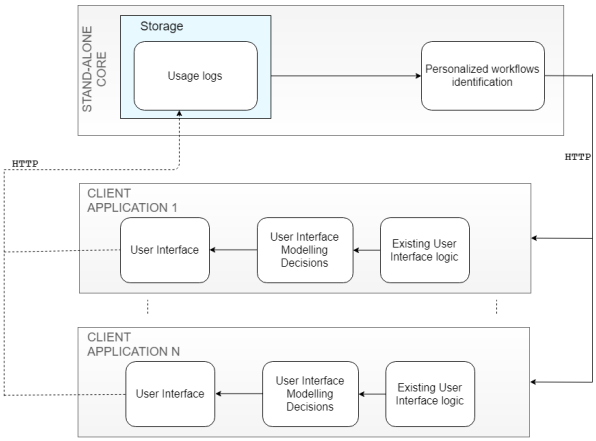

Figure 1: System's high-level architecture

Under normal circumstances (i.e. without personalization impact), client applications are perfectly functional systems. Personalization shall not alter the targeted environment existing logic and functionality by any means; it should come as an enhancement factor, and not a limitation nor a logic-improvement aspect. This implies a necessity of client systems to follow best coding-practices (i.e. high cohesion, low coupling) so that the impact and effort needed to add, modify or remove elements (i.e. User Interface buttons) is minimum.
The entire personalization process should behave, from end user's point of view, as an invisible assistant that follows performed steps, understands the situation that user resides in and is capable of making changes without his explicit intervention.

Users should be informed if changes were made or can be made, while giving them the ability to continue as personalization process intends to, or to interrupt the already-prepared adjustments.

Direct interaction between users and personalization system will not disturb the state that user is currently in, which will always be brought to front. Standard personalization elements (i.e. notification messages from the core system) will be as minimalist possible, while personalized features shall integrate in the application overall touch and feeling.

Personalized adjustments set a delicate curve that, given the context of an intelligent system, can be followed in an oscillating manner. Faulty situations (i.e. wrong understanding of user habits) can occur and should be considered as must-deal-with situations. Alternatives can generally be handled by taking advantage of the real-time characteristic of the system, where immediate actions can be taken, given faulty or unexpected scenarios and not only.

User's interaction with the system will have priority in both usability-making and personalization-making processes. This means that, given the situation where personalized built behavior becomes questionable due to changes followed in user behavior, those changes should prevail and trigger adjustments to settle personalization output on the expected path.

The execution flow within the system shall be triggered when users begin using the application (i.e. on login). This does not imply immediate visible personalized effects, but immediate awareness of changes that occur during usage. When personalization is triggered, the core-system is responsible to gather, organize and preprocess logs belonging to end user. In order to maintain consistency, a relational database is proposed, where a table is populated with received logs.

To surpass type dependencies and logs format mismatches, we propose a Data Definition Language trigger attached on the database and a Data Manipulation Language trigger attached on the logs table. DDL trigger will be used to create an additional table where logs data is converted to format understood by Data Mining algorithms (i.e. assure action names identifiable by integer ids). DML trigger is used to populate the new created table whenever logs table is updated (i.e. new log is added). *Figure 2* illustrates this flow of actions.

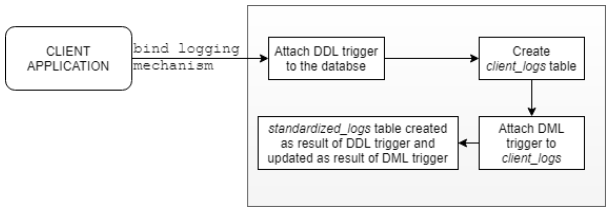

Figure 2: DDL and DML operations on logs data

Once data is properly stored and updated, it will be used by application logic consisting in pattern sequences identification. For this purpose, current fastest sequential pattern mining algorithm [1] TKS is considered. The most used sequences delivered by TKS will be sent over to client application, which will made contextual changes that are not bound to the server, mining algorithm or other client applications, but to current application's context only. This gives flexibility and openness in changing decisions taken on personalization level and stand-alone characteristic of the core server responsible of data mining tasks. *Figure 3* details mining algorithm workflow:

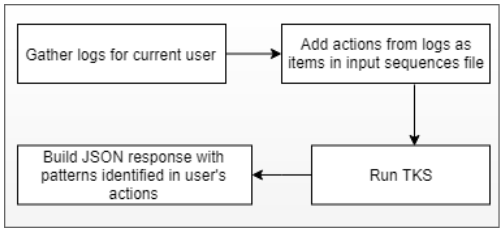

Figure 3: Mining workflow

SEQUENTIAL PATTERN APPROACH: TKS Algorithm

To address the problem of identifying the most relevant sequences that define user's behavior, we propose the usage of Sequential Patterns Algorithms [7]. Sequential Pattern Mining approach addresses the problem of identifying all frequent sequential patterns such that their occurrence is greater than a set minimum threshold.

An itemset X is a set of items such that $X \subseteq I$, where $I = \{i_1, i_2, \ldots, i_m\}$ is a set of items. A sequence is an ordered list of itemsets $s = (I_1, I_2, \ldots, I_m)$, such that $I_k \subseteq I$ $(1 \leq k \leq n)$. For example, the sequence $(\{a, b\}, \{c\}, \{f, g\}, \{g\}, \{e\})$ can represent five transactions made by a customer in a store, where each letter represents an item and items between curly brackets represent an itemset [7]. Multiple such sequences having assigned a sequence identifier SID, could build a sequence database.

ID	Sequences
S1	(1), (1 2 3), (1 3), (4), (3 6)
S2	(1 4), (3), (2 3), (1 5)
S3	(5 6), (1 2), (4 6), (3), (2)
S4	(5), (7), (1 6), (3), (2), (3)

Figure 4: Sequence database example (adapted from [34])

Such sequence databases are targeted for analyzing and processing so to find interesting patterns. These actions define an enumeration problem. The aim of the enumeration problem is to list all subsequences that are classified as being frequent, based on threshold selection.

A naive approach would be to compute the occurrences of all possible subsequences in a sequence database and then output only those meeting the requirements. This would be very inefficient, because the number of the subsequences can be very large (a sequence database having *n* items can have up to 2^n-1 distinct subsequences). Hence, efficient algorithms designed to avoid entire search space of all possible subsequence's exploration must be considered.

The algorithms considered for sequential pattern mining problems generally differ by four aspects:

- Whether they use *Breadth-first Search* or *Depth-first Search* approach to scan the database. *Breadth-first Search* approach is treated, for example, by GSP algorithm [12] , and implies a first scan of the database to find all sequential patterns containing only one item, out of which are generated frequent-2 sequences, then frequent-3 sequences and so on (level-wise approach, since ascending order length sequences are considered) [7].
  Depth-first Search approach, is treated, for example, by Spade algorithm [35] or Spam [9], and implies recursively extensions made on sequences consisting of only one item in order to generate larger sequences.
- Type of database representation: *vertical* representation or *horizontal* representation, which differ only in the way the information is stored.
- How next patterns are generated: maintaining candidate patterns in memory or generating possible non-existing candidate patterns.
- How the occurrence (support) of a subsequence is counted.

Advantages and disadvantages of each available alternative can be underlined, but the differences would be noticeable only on the efficiency level. A very important aspect of sequential pattern mining is that no matter the chosen algorithm, there is always only one correct answer to a sequential mining task, given a sequence database and a minimum threshold. [7]

This can be seen as a great advantage in the context of personalization system, where prediction probabilities would hardly match 100% accuracy. One could argue that the outcome of sequential mining might be always correct, but the context given the personalization might still often fail, meaning that the sequences that are representative for user's behavior even if accurate, could not reflect the behavior in its entirety. We will discuss this later in this work.

In a personalization system, the human expertise needed for the personalization logic to work should be minimum. Given the context of Sequential Pattern Mining, the output provided by algorithms is very dependent of their data and very likely to require extensive tuning, since the number of patterns generated is highly dependent on how the threshold is chosen. This problem is important because, in practice, limited resources are available and fine-tuning parameters is time-consuming [1]. Since very high number of patterns are likely to be generated, and situations like setting to low threshold and omitting valuable information are to be avoided, option to input a desired number of patterns to be identified is a must. In this sense, TKS Algorithm [1] was developed, allowing users to skip threshold tuning in order to get the desired amount of sequences.

In the same context of mapping Sequential Pattern Mining on a personalized environment, the assumptions made on the sequence database made by most of the available algorithms must be considered. All above mentioned algorithms are batch algorithms and assume database to be static, meaning that they are designed to be applied once on a given data and any data updates would require entire data to be rescanned. It is of great importance to develop extensions (if batch

algorithms are to be used) or consider alternatives (like Incremental Algorithms proposed in the fields of Stream Mining), if real-time personalization is to be achieved.

TKS proposes a vertical database representation, basic candidate generation and several efficient strategies to discover top-k sequential patterns efficiently. [1]
The vertical database representation [9] is defined as a set of bit vectors, where each bit is set to 1 if and only if item x appears in the itemset represented by this bit, otherwise, is set to 0.
The procedure for candidate generation [9] consists in one database scan, during which the occurrence (support) of each single item is computed. Then, each frequent candidate pattern starting with single items identified is recursively computed. Note that infrequent patterns are not extended and, unlike previous approach, TSP, no additional database scan is required and no projection-based approach is considered.
Having SPAM algorithm [9] as starting point, four strategies are used to improve efficiency [1]. First consists in raising the support threshold to the support of pattern identified valid and with the lowest support. This prunes the search space so that unnecessary computations are avoided. Second strategy consists in generating most promising patterns first, so that minimum support can be raised faster. Discarding infrequent items in the candidate generation stage consists the third improvement strategy. This way, unnecessary bit vector intersections are avoided when frequent sequences are to be discovered. The last strategy consists in introducing a new data structure, called Precedence Map, which holds for each item *i*, a list a triple <*j, m, x*>, where m is the number of sequences where *j* appears after *i* by *x*-extension.
We included TKS algorithm and used the created *standardized_logs* table so to fit on default input format required. Since logs by themselves are a pure list of items and sequences input is needed, we took advantage of the timestamp of each log and considered that for less than 1-minute gap, all the logs registered are belonging to one sequence. If a new log is added, it is checked if the previous existing log happened longer than 1 minute ago. If no, the new log is added to the current sequence, and if yes, a new sequence is created This allows us to add an incremental approach to the TKS batch algorithm. The algorithm is currently stated as the fastest Top-K Sequential Pattern algorithm [1], and reliable results can be obtained if running happens at predefined time intervals (i.e. when a new sequence is completed).

WEB APPLICATION INTEGRATION

We propose an Angular framework-based application in the context of configuring products in a company. Users interaction with the system is marked by 20 different actions, each adding a new entry in the logs table when executed.
The actions are focused on changing products and system properties and on system's navigation possibilities (i.e. *searchProduct*, *changeLanguage*, *generateDocuments*, *createCart*, *addProductToCart,* etc.)
Changes that we propose do not impact application's business logic and are mainly focused on template changes. User Interface elements position or presence in the current window are the basic noticeable changes that could mark an improvement generated as result of TKS output. For example, if TKS has identified that *editProduct* action always follows *createNewProduct* action, and each is identified with a HTML button in the UI, a usability improvement would be either to automatically trigger the *editProduct* action after *createNewProduct* is executed, or to bring those buttons closer to each other. An example of change that could be made is depicted in Figure 5.

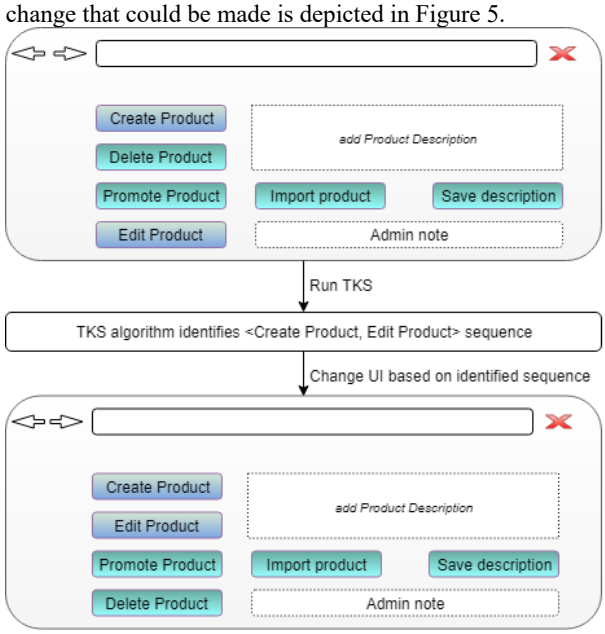

Figure 5: Action taken to improve usability based on TKS output

CORRECTNESS AND EVALUATION

Since TKS is correct and complete, and no changes in the functionality of the algorithm were brought, it can be concluded that the Data Analysis stage is correct and complete. Most used actions that mark usage sequences in user interaction with the system are correct and so the habits that users manifest within the system are valid.
Changes in User Experience that involve for example User Interface elements adjustments, are the key aspect to the evaluation of the entire system.
Based on the correctness certainty of the TKS, we will evaluate the core system's efficiency by performance measurements on running TKS on real log data within a Software company.
The evaluation of the end to-be-personalized application is completely dependent of the application type and changes that are planned are made based on what the focus should be during user's journey. Those changes are assumed to happen

exclusively inside the application structure and no influence from the core personalization system can be brought, except the performance of the TKS algorithm. Few experimental studies have been made, by taking 3 users with different amount of log history data.

UserID	Logs number	K	Patterns found	Total time(ms)	Max. Memory Used (Mb)
Usr256	158	3	3	1	100.01
		5	6	5	84.55
		10	10	4	82.41
Usr1	2660	3	3	2	96.527
		5	7	3	105.05
		10	10	4	103.68
Usr27	10753	3	5	13	171.73
		5	6	14	75.83
		10	10	12	69.19

Table 1: Performance results

Setting a value for parameter K (i.e. changing the desired amount of top sequences) does not mean that exactly K sequences will be generated. If 2 sequences having the same occurrence number are encountered, both patterns will be returned.

Algorithm is influenced by intersections count between logs data, but even when talking about tens of thousands of logs fed into the algorithm, no more than few tens of milliseconds are required to gather needed insights. Hence, the real time behavior can be obtained using TKS batch algorithm approach.

It can be observed that increasing the number of logs does not necessarily mean performance degradation. Given the complexity of the sequential problem, where no threshold tuning is required, the resources and time needed are minimum.

CONCLUSION

We proposed a centralized stand-alone system that yields personalized user experience to a target application under logs analysis context. The modules that comprise a personalization system were underlined, emphasizing the usage mining module. We introduce a real-time Sequential Pattern Mining approach to gather knowledge about on-going user's habits. Solutions and limitations on integrating personalization aspects in a to-be-personalized environment were mentioned and the applicability of the personalized gathered knowledge was exemplified in the context of a Web application used to manage and configure products in a company.

The system is able to deliver knowledge about user habits in real-time constraints with no error regarding most used sequences.

Further improvements involve a pure incremental approach such that the entire database scan can be avoided when new logs are added. Also, the direct impact on the personalized application should not involve implicit implementation changes. Changes visible to user should be extended as attachments that can be integrated when needed.

REFERENCES

[1] P. Fournier-Viger, A. Gomariz, T. Gueniche, E. Mwamikazi and R. Thomas, "TKS: Efficient Mining of Top-K Sequential Patterns," *The International Conference on Advanced Data Mining and Applications,* pp. 109-120, 2013.

[2] P. S. Yu, "Data mining and personalization technologies," in *Sixth International Conference on Database Systems for Advanced Applications, DASFAA, pp. 6–13*, 1999.

[3] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc 20th int conf very large data bases, VLDB p. 487–99*, 1994.

[4] J. Han, J. Pei, Y. Yin and R. Mao, "Mining frequent patterns without candidate generation: a frequent-pattern tree approach," *Data Mining and Knowledge Discovery,* vol. 8, no. 1, pp. 53-87, 2004.

[5] J. M. Zaki, "Scalable algorithms for assciation mining," *Transactions on Knowledge and Data Engineering,* vol. 12, no. 3, pp. 372-390, 2000.

[6] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang and D. Yang, "H-mine: Hyper-structure mining of frequent patterns in large databases," *International Conference on Data Mining,* pp. 441-448, 2001.

[7] P. F. Viger, Chun-Wei, J. Lin, U. K. Rage, Y. S. Koh and R. Thomas, "A Survey of Sequential Pattern Mining," *Data Science and Pattern Recognition,* vol. 1, no. 1, pp. 54-77, 2017.

[8] J. Han, J. Pei, M. Mortazavi-Asl, Q. Chen, D. U. and C. H. M., "FreeSpan: frequent pattern-projected sequential pattern mining, "*International Conference on Knowledge Discovery and Data Mining",* pp. 355-359, 2000.

[9] J. Ayres, J. Flannick, J. Gehrke and T. Yiu, "Sequential pattern mining using bitmap representation," *International Conference on Knowledge Discovery and Data Mining,* pp. 429-435, 2002.

[10] P. F. Viger, A. Gomariz, M. Šebek and M. Hlosta, "VGEN: fast vertical mining of sequential generator patterns," *The International Conference on Data Warehousing and Knowledge Discover,* pp. 476-488, 2014.

[11] P. F. Viger, C.-W. Wu and V. S. Tseng, "Mining Maximal Sequential Patterns without Candidate Maintenance," *The International Conference on*

Advanced Data Mining and Applications, pp. 169-180, 2013.

[12] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," *The International Conference on Extending Database Technology,* pp. 1-17, 1996.

[13] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl, "GroupLens: An Open Architecure for Collaborative Filtering of NetNews," in *Proc of the ACM Conference on Computer-Supported Cooperative Work*, 1994.

[14] U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating "World of Mouth," in *Proc of the Conference on Human Factors in Computing Systems*, 1995.

[15] B. Krulwich and C. Burkey, "Learning User Information Interests through Extraction of Semantically Significant Phrases," in *Proc of the AAAI Spring Symphosium on Machine Learning in Information Access*, 1996.

[16] K. Lang, "Newsweeder: Learning to Filter Net-News," in *Proc of the 12th Intl. Conference on Machine Learning*, 1995.

[17] P. S. Yu, "Data mining and personalization technologies," in *Proc of the Sixth International Conference on Database Systems for Advanced Applications*, 1999.

[18] E. Solutions, "Regression, Data Mining, Text Mining, Forecasting using R," ExcelR Solutions, [Online]. Available: https://www.udemy.com/data-science-using-r/. [Accessed 1 December 2018].

[19] C. C. Aggarwal and P. S. Yu, "A new Framework for Itemset Generation," in *Proc. of the ACM Symposium on Principles of Database Systems*, 1998.

[20] M. J. Berger and I. Rigoutsos, "An Algorithm for Point Clustering and Grid Generation," *Transactions on Systems, Man and Cybernetics,* vol. 21, no. 5, pp. 1278 – 1286, 1991.

[21] T. Zhang, R. Ramakrishnan and M. Livny, "BIRCH: An efficient Data Clustering Method for Very Large Databases," *ACM SIGMOD Conference,* 1996.

[22] T. Imielinski, A. Swami and R. Agrawal, "Mining Association Rules between Sets of Items in Very Large Databases," *ACM SIGMOD Conference,* 1993.

[23] R. Kohavi and D. Sommerfield, "Feature Subset Selection Using the Wrapper Method: Overfitting and Dynamic Search Space Topology," *International Conference Knowledge Discovery and Data Mining,* 1995.

[24] S. Samson, K. Granath and A. Alger, "Journey Mapping the User Experience," 2017.

[25] J. C. O. Nicolás and M. Aurisicchio, "A scenario of user experience," *ICED 11 - 18th International Conference on Engineering Design - Impacting Society Through Engineering Design,* 2011.

[26] M. Hassenzahl, "Experience Design: Technology for All the Right Reasons," *(Morgan & Claypool Publishers,* 2010.

[27] J. Forlizzi and K. Battarbee, "Understanding experience in interactive systems", *Proceedings of the 5th conference on Designing interactive systems: processes, practices, methods, and techniques,* pp. 261-268, 2004.

[28] E. Karapanos, M. Hassenzah and J.-b. Martens, "User experience over time", in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems Pages*, pp. 729-738, 2008.

[29] M. E. H. Creusen and J. P. Schoormans, "The different roles of product appearance in consumer choice", *Journal of product innovation management,* vol. 22, no. 1, p. 63–81, 2005.

[30] N. Tractinsky and D. Zmiri, "Exploring attributes of skins as potential antecedents of emotion in HCI," *Aesthetic computing,* pp. 405–422, 2006.

[31] L.-C. E. Law, V. Roto, M. Hassenzahl, A. P. O. S. Vermeeren and J. Kort, "Understanding, scoping and defining user experience: A survey approach," in *Proc. of the 27th international, pp. 719–728*, 2009.

[32] J. M. Hektner, J. Schmidt and M. Csikszentmihalyi, "Experience sampling method: Measuring the quality of everyday life", SAGE Publications, Inc; 1 edition (August 18, 2006), 2009.

[33] M. Eirinaki and M. Vazirgiannis, "Web mining for Web personalization," *ACM Trans. Internet Technol.,* vol. 3, no. 1, p. 1–27, 2003.

[34] P. Fournier-Viger, C. Lin., A. Gomariz., T. Gueniche., A. Soltani, Z. Deng. and H. T. Lam, "The SPMF Open-Source Data Mining Library Version 2.," in *Proc. 19th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD) Part III, pp. 36-40*, 2016.

[35] J. M. Zaki, "SPADE: An Efficient Algorithm for Mining," *Machine Learning,* vol. 42, pp. 31-60, 2001.