Exploring client logs towards characterizing the user behavior on Web applications

Leandro Guarino de Vasconcelos[a], Rafael Duarte Coelho dos Santos[a] and Laércio Augusto Baldochi Jr.[b]

[a]Instituto Nacional de Pesquisas Espaciais, Sao Jose dos Campos, Brazil;
[b]Universidade Federal de Itajuba, Itajuba, Brazil

ABSTRACT

Analysis of user interaction with computer systems can be used for several purposes, the most common being analysis of the effectiveness of the interfaces used for interaction (in order to adapt or enhance its usefulness) and analysis of intention and behavior of the users when interacting with these systems. For web applications, often the analysis of user interaction is done using the web server logs collected for every document sent to the user in response to his/her request. In order to capture more detailed data on the users' interaction with sites, one could collect actions the user performs in the client side. An effective approach to this is the USABILICS system, which also allows the definition and analysis of tasks in web applications. The fine granularity of logs collected by USABILICS allows a much more detailed log of users' interaction with a web application. These logs can be converted into graphs where vertices are users' actions and edges are paths made by the user to accomplish a task. Graph analysis and visualization tools and techniques allow the analysis of actions taken in relation to an expected action path, or characterization of common (and uncommon) paths on the interaction with the application. This paper describes how to estimate users' behavior and characterize their intentions during interaction with a web application, presents the analysis and visualization tools on those graphs and shows some practical results with an educational site, commenting on the results and implications of the possibilities of using these techniques.

Keywords: client log analysis, graph mining, user behavior

1. INTRODUCTION

Every day, a huge amount of information is produced by information systems. Most of this information is logging data, wich is usually stored in order to keep history records about the usage of the systems. In spite of being hardly used, this data remains stored for security reasons.

Exploring logging data is not a trivial task, as it is necessary a large amount of time and resources in order to process this data. However, logs contain hidden information which may be valuable for a given organization, such as patterns of usage of its systems. We advocate that, by understanding these patterns, organizations may improve their systems.

As of today, the majority of information systems are Web-based. In Web applications, it is possible to log data in server-side or in client-side. Server-side logging captures the pages visited by a given user. Client-side logging, on the other hand, may capture much more information, such as mouse moves, mouse clicks, page scrolling, and so on. Therefore, analyzing client-side logs can reveal how users behave. Thus, when logs from several users of the same application is analyzed, it is possible to discover usage patterns.

Towards discovering such patterns, we propose USAGRAPH, an algorithm that analyzes client logs and provides information that helps the characterization of users' behavior in Web applications. USAGRAPH defines a sequence of steps for processing client-side logs that allows identifying the most common actions

Further author information: (Send correspondence to L.G.V.)
L.G.V.: E-mail: le.guarino@gmail.com
R.D.C.S.: E-mail: rafael.santos@inpe.br
L.A.B.J.: E-mail: baldochi@unifei.edu.br

Next-Generation Analyst, edited by Barbara D. Broome, David L. Hall, James Llinas,
Proc. of SPIE Vol. 8758, 87580G · © 2013 SPIE · CCC code: 0277-786X/13/$18
doi: 10.1117/12.2015711

performed by end-users. In order to shorten the processing time, our algorithm explores USABILICS'[1] task definition and analysis tool, which is used to filter the actions that are going to be processed according to the preferences of the web application evaluator. After processing the logs for a given task, USAGRAPH helps finding patterns of actions performed by users during their interactions in the application interface.

This paper is organized as follows: Section 2 compares server logging and client logging, and presents related work that explores client-side logs; Section 3 presents the USAGRAPH algorithm, detailing its six steps, and presents a case study of the usage of this algorithm in a real Web application; Section 4 presents a discussion of our approach for processing logs; finally, Section 5 brings our conclusions and discuss future work.

2. CLIENT LOG ANALYSIS

The log analysis has become an important subject of study in computing due to interdisciplinarity involved in the use of logs (eg, data mining, artificial intelligence, data storage, high-performance processing, data traffic). Additionally, logs have been used to generate knowledge for business, personalize web applications,[2] evaluate the quality of Web applications,[3] build user profiles[4] , detect topics and events in search logs,[5] identify web spam pages[6] and improve web site search.[7]

Many approaches use server logs, which contain information about the pages visited by users, the timestamp of access and the IP address of the machine that requested the page. Other approaches utilize client logs, which tend to store more information about the users' actions on the local machine. Both approaches can be effective for some purposes, but server logs may be insufficient for others due to the reduced amount of information about users' actions.

For the analysis of user behavior on interfaces, server logs allow identifying the acessed pages, but do not offer details about the traveled paths to the execution of these actions. Therefore, for this purpose, the most recent approaches utilize client logs.

Section 2.1 presents the most relevant approaches on usability evaluation and analysis of user behavior in web applications.

2.1 Related works

The most used approach for remote usability evaluation is based on capturing the end-user interaction using logs. WELFIT,[8] UsaProxy,[9] WAUTER,[10] WebRemUSINE[11] and USABILICS[1] are examples of tools that explore this approach.

WELFIT employs a graph-based algorithm for identifying patterns on the events performed by the end user in a single page. As a result, it presents a chart that summarizes the most important actions that have been performed. Comparing to our system, this tool do not consider paths connecting different pages and do not provide specific information concerning usability problems.

UsaProxy is a proxy-based system in which the capture of events depends on the configuration of the user's browser and the usage of a proxy-server. This system performs the task analysis.

WAUTER also performs task analysis in its evaluation process and collects data by a proxy-based system installed on the client machine, it is not browser-dependent. The proxy-based software intercepts the HTML code (regardless of how it was generated on the server) enroute to the browser, and inserts within it, appropriate logic (in standard JavaScript) to capture all the user events of interest.

WebRemUSINE relies on a task model to perform the analysis of tasks. The data are collected through a tool implemented by a combination of JavaScript and Java applet.

USABILICS allows an efficient approach for remote and automatic usability evaluation[12] because collects many details of user interactions in client-side and provides an efficient way to analyze the tasks of a web application, comparing the performed actions by the user with the optimal path defined by the evaluator.

In this paper, the proposed algorithm uses the task analysis of USABILICS to filter the relevant actions of the client logs and uses its data collection module to analyze user behavior. Thus, Section 2.1.1 briefly describes the operation of the task analysis and data collection of USABILICS.

2.1.1 USABILICS

Users interact interfaces to comply goals such as making a purchase, to send a comment or download a file. These goals can be separated into one or more tasks that represent their parts. According to this idea, USABILICS offers a module where the evaluator of the web application can define the tasks that must be studied. From this, the data collection module of USABILICS captures all users' actions on interface with high level of detail of manipulated objects in each action. All HTML and CSS attributes of the manipulated objects are collected in addition to information about the display of pages in the browser.

Combining the data collected and the defined tasks by the evaluator allows automatic task analysis, resulting in a usability index, which represents the effectiveness and efficiency of the interface, and in wrong actions occurring in each interaction.

A task analysis of USABILICS processes performed actions by users from the first action until the final action of the defined optimal path by the evaluator. Thus, it is possible to limit the data that will be processed in accordance with the evaluator's interest.

3. EXPLORING CLIENT LOGS

The user interaction in web applications usually produce a big amount of server requests, proportionally to the number of users that access the application and also the time that users remain browsing, resulting in large files of server logs. When client logs are used, the amount of data is even greater, because all users' actions are captured and stored during browsing, such as clicks, mouse moves, scrolling, and the page requests stored in server logs.

Thus, to process client logs in order to extract relevant information is not an easy job, due to the amount of data that must be processed and the expected response time.

In this section, we present an algorithm for analyzing client logs, which aims to extract relevant information about the users' behavior in web applications. Also in this section, we present a case study of this algorithm in an e-learning, exemplifying as it does the extraction of information from the client logs in a real web application. In the case study, we present the performed tests and the achieved results in the web application studied.

3.1 USAGRAPH Algorithm

As presented in Section 2, client logs of web applications are composed of all executed actions by the user and by the application in client-side. So in the same way that is possible to identify in server logs the sequence of pages visited, in client logs, for instance, it is possible to know the sequence of clicks, mouse moves and loaded pages.

Therefore, the executed actions by a user during the interaction on the web application can be transformed into a linked list, where each element represents an action. However, it is likely that some actions are executed repeatedly, so it is possible to transform this linked list in a directed graph, where the vertices represent the users' actions and the edges represent the precedence relation between actions. Consequently, each user interaction can be represented by a directed graph of captured actions on the web application.

According to this idea, the USAGRAPH algorithm was developed to represent the users' interactions and to extract the most relevant performed actions during browsing in order to analyze the users' behavior.

Considering the huge amount of generated data by a web application in client logs, there is a high cost (time and resources) for the processing of all data in the logs, and not all data are relevant to the analysis of user behavior. Therefore, we need an effective way to filter the relevant data from the client logs.

For this, the proposed algorithm uses the result of task analysis from USABILICS. Based on the defined tasks by the evaluator of the web application, the aspects of interest are defined. According to propose of the evaluation, it can be defined a single task or all tasks that users perform on interface. Then, performing tasks analysis of USABILICS, the performed actions by users are filtered from the client logs since the beginning of a task by the end of its execution (if the user has completed it). So the relevant actions are filtered, because they were executed in an attempt to accomplish a task during browsing. It's important to emphasize that the graphs are generated for both interactions where the user accomplished the task as those in which he did not accomplish it, because one objective of the analysis of user's behavior is to discover why a started task was not completed.

The steps of the algorithm USAGRAPH are:

1. **To execute the task analysis of USABILICS:** for each defined task by the evaluator of the web application, it is executed the algorithm of the task analysis of USABILICS, in order to filter the relevant actions from the client logs.

2. **To generate graphs for each user's interaction:** In this step, for each user's interaction where a defined task of the evaluator was initiated, it is generated a directed graph, where the vertices represent the actions and the edges indicate the precedence relation between the actions.

3. **Overlay graph:** different users can perform the same actions during browsing, therefore, in this step, the graphs of each interaction are overlapped in order to merge the vertices and create a single directed graph with the actions of all users that initiated the task. Even in this step, weights are assigned to the edges, where the weight of an edge AB is the number of times that the action B was performed immediately after the action A.

4. **Graph analysis - visualization:** for each defined task by the evaluator, the graph resulting from the previous step is designed in order to verify visually the existence of patterns of the most relevant actions. Due to the large number of vertices and edges, it was used a force-directed layout algorithm, the Fruchterman-Reingold,[13] to enhance the graph visualization.

5. **Graph analysis - vertices and edges:** in order to complement the analysis of the previous step, for each defined task, the heaviest edges and vertices with higher degree (in-degree + out-degree) are identified. This information reflects the sequences of most performed actions by the users.

6. **Graph analysis - maximum and minimum paths:** From the resulting graphs of the third step, two algorithms are executed: one to find the shortest path and other to find the maximum path (heavier) between the initial action and the final action of each defined task. To find the shortest path, we used Dijkstra's algorithm[14] and, for the maximum path, we adapted this algorithm to find the path of heavier edges, which we called *Reverse Dijkstra*. For each task, these algorithms find the maximum and minimum paths passing through all the actions of the optimal path of the task. The results of these algorithms are presented in text format, containing details about the actions that belong to the paths found, in order to allow analyzing user's behavior.

To exemplify the use of the algorithm USAGRAPH, it was realized a case study with one real web application. This case study is shown in Section 3.2.

3.2 Case study

For this case study, it was selected one real web application to exemplify the use and efficiency of the algorithm USAGRAPH. Section 3.2.1 describes the web application that was studied, briefly explaining its purpose and functionality; and Section 3.2.2 shows how the steps of USAGRAPH were executed and what results were achieved regarding the analysis of user's behavior.

3.2.1 Web application: e-learning

The web application studied is an e-learning site (available on www.4learn.pro.br), which allows the communication between professors and students in various academic degrees. Basically, in this system, the professor can publish his material using hyperlinks or typing it to build an electronic book, he may create activities to be answered by students and later be corrected by him; and he can send an e-mail to students of a given class.

For the analysis of user's behavior in e-learning, two tasks were defined: (a) Correct a student activity and (b) Send an e-mail to students of a class. To define the optimal paths of these tasks it was used UsaTasker module[15] of USABILICS, resulting in the following actions for each task:

(a) Correct a student activity

1. Click the link *Correct activities*
2. Open the page with the list of classes and courses that the professor teaches
3. Click the image concerning the correction of activities, beside a class's identification
4. Open the page with the list of questions to be corrected
5. Click the image beside each question in screen to open the answer form
6. Click the button *Answer*

(b) Send an e-mail to students of a class

1. Click the link *Send e-mail to the class*
2. Open the page with the list of classes and courses that the professor teaches
3. Click on a given class/course
4. Open the page with list of students of the class
5. Type the message subject
6. Type the message text
7. Click the button *Send*

3.2.2 Tests

For this case study, data was collected during 197 days, resulting in 4346 different interactions and 1,234,375 performed actions during browsing on web application. Using these data we performed the steps of USA-GRAPH, which are detailed below.

Step 1: task analysis of USABILICS was performed to filter the relevant actions of the defined tasks by the evaluator. This step resulted in 70,107 actions of the task (a) e 28,164 actions of the task (b), reducing significantly (92%) the amount of actions to be processed, which allows speed up the processing. It's important to emphasize the filtered actions are those between start and end of the attempt to accomplish a task. In the task (a), for example, all filtered actions were performed since the time that the user clicked the link *Correct activities* until click the *Answer* (if the user has completed the task) .

Step 2: Using the filtered actions in step 1, graphs of each interaction in which the user initiated a task were created. For the task (a), 38 graphs were created, since there were 38 attempts to perform the correction of a student activity. For task (b), 52 graphs were created, because there were 52 attempts by professors send an e-mail to students.

Step 3: For each task, the generated graphs in the previous step were overlapped, eliminating the duplicity of the vertices and assigning weights to the edges, as explained in Section 3.1. The result of the overlap were two graphs, one for each task. The overlay for the task (a) resulted in a graph with 1504 vertices and 7106 edges, the overlay for the task (b) resulted in 1356 vertices and 6305 edges. It is possible to note that, although there is a greater number of interactions (52) for the task (b), the task (a) resulted in more vertices and edges because it is more complex than the task (b) in terms of web interface.

Step 4: The generated graphs in the previous step were designed by applying the algorithm 13, resulting in the following graphs, Figures 1 and 2 for the tasks (a) e (b), respectively.

In Figures 1 and 2, thicker lines represent edges with greater weight, the green dots represent the actions of the optimal paths of tasks and highlighted areas by red ellipses represent vertices that have high in-degree

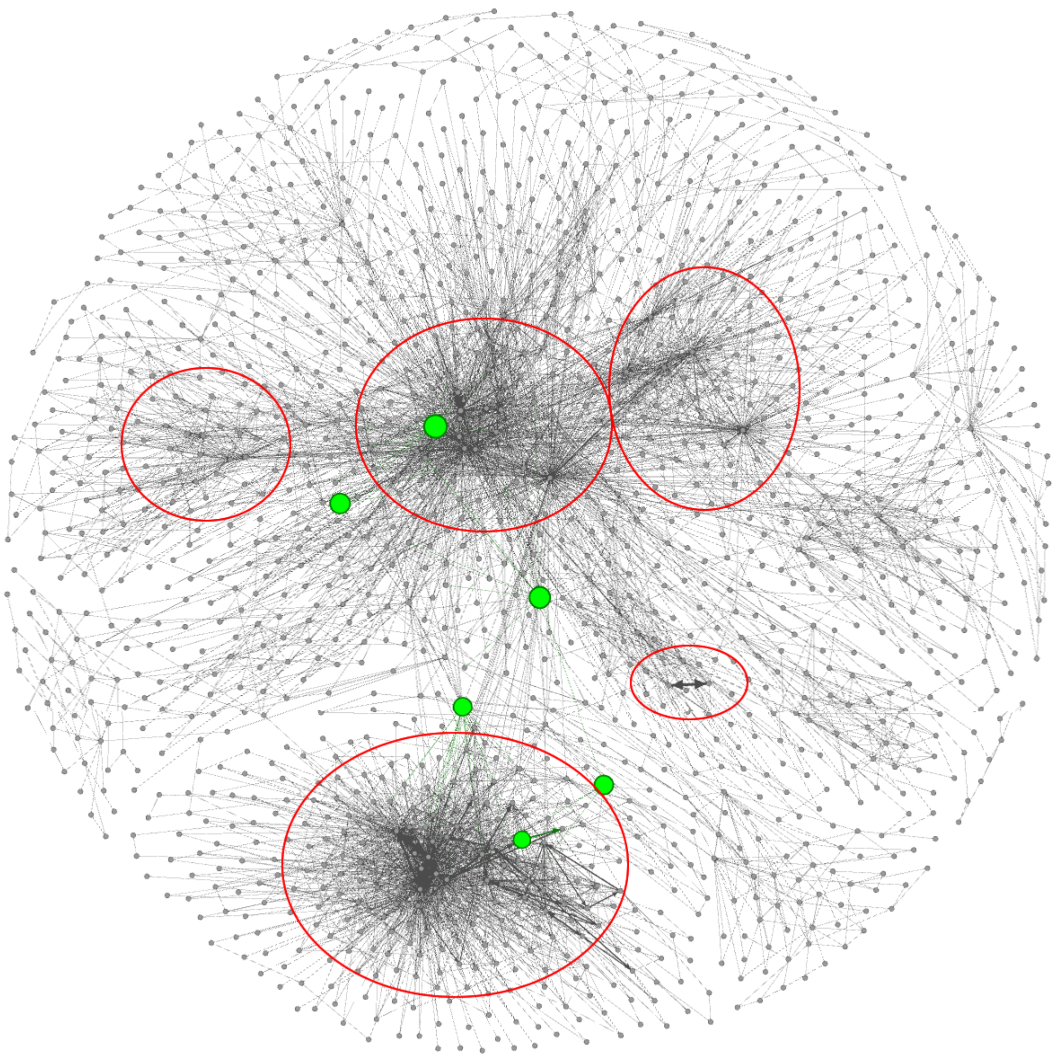


Figure 1. The generated graph by applying the algorithm USAGRAPH on data of the task (a).

or out-degree, that indicate patterns of the most relevant actions. However, visually analyze the graphs of Figures 1 and 2 is difficult due to the number of vertices and edges. To minimize this problem, new graphs were generated by removing the actions that can represent the interest but not a user's decision, such as mousemove, mouseover, mouseout, blur, focus. Applying this filter, the graphs shown in Figures 3a and 3b were generated for the tasks (a) and (b), respectively.

The goal of visualizing the graphs of Figure 3 is to identify connections between vertices that may indicate patterns of relevant actions.

Step 5: In order to emphasize the most performed actions, an algorithm was performed to identify the heaviest edges and vertices with greater degree (in-degree + out-degree), which represent more performed actions on each task, allowing to verify obvious actions and mainly to detect unusual actions.

Step 6: Although the visualization of graphs from step 4 promote the perception of patterns of actions, travel through the vertices can not be fast and practical. Therefore, using Dijkstra's algorithm and an adaptation of what we called *Reverse Dijkstra*, the minimum and maximum paths were found for each task between the initial action and the final action on the optimal path defined by the evaluator. The actions of found paths were presented in textual format containing the following information: the distance (cumulative weight

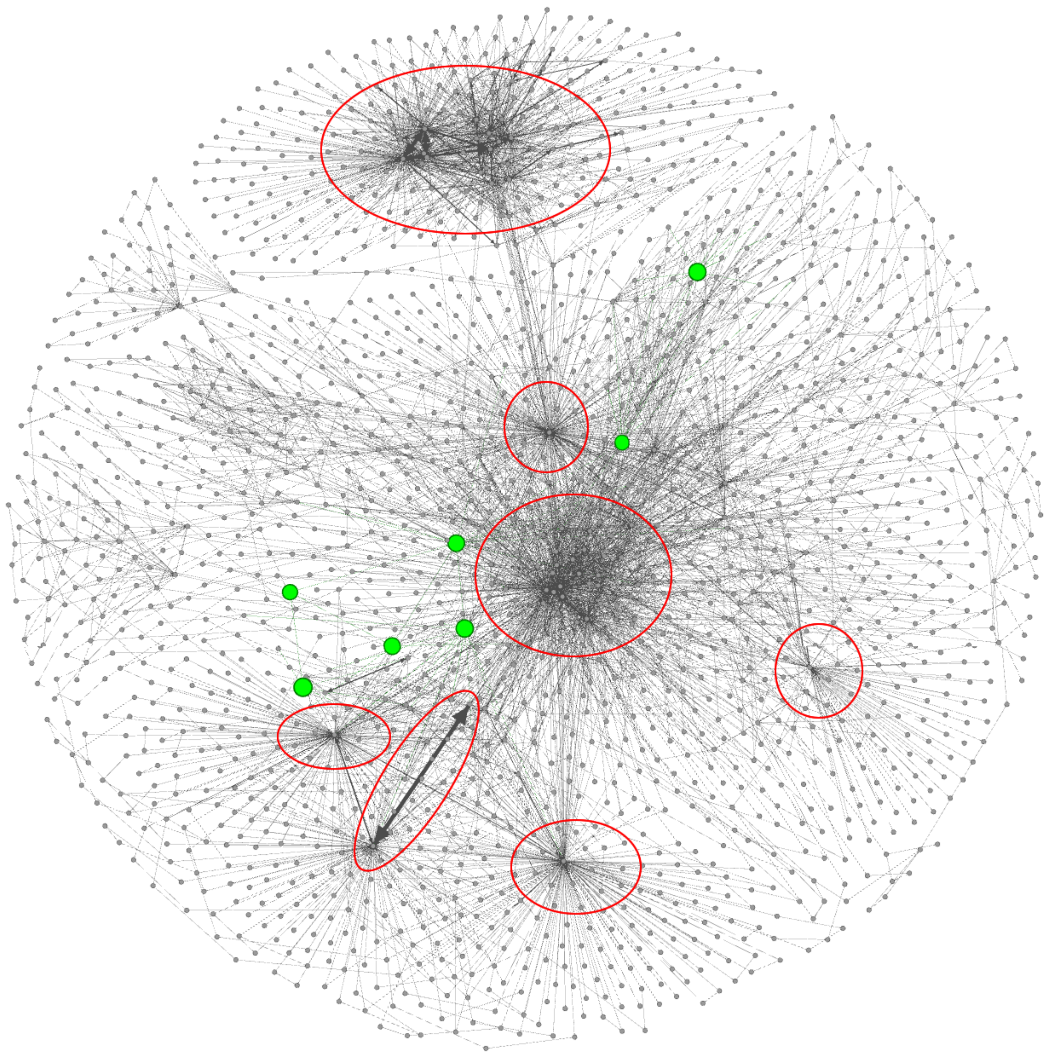


Figure 2. The generated graph by applying the algorithm USAGRAPH on data of the task (b).

of edges) in the graph, the type of action, the manipulated object on the page, and the page on which the action occurred.

The results of analysis of the found paths and information from step 5 are described in Section 3.3.

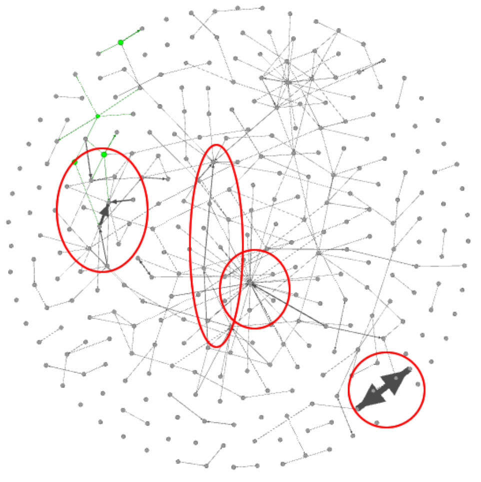
3.3 Results

In this section, to demonstrate how the analysis was done in the case study, tables are presented containing a subset of the generated data by USAGRAPH.

Figures 1, 2 and 3 shown in Section 3.2.2 suggest that there are patterns of performed actions by users during attempting to accomplish the tasks, especially in the highlighted areas by red ellipses. Thus, executing the step 5 of USAGRAPH the heavier edges were returned, i.e., pairs of more executed actions, and also the vertices with greater degree (in-degree + out-degree) were returned.

For example, Table 1 contains a list of the 16 heaviest edges of the task (a) *Correct a student activity* and Table 2 list a subset of vertices with greater degree in the same task.

In the Table 1, it is possible identify some pairs of obvious actions, such as: the edge 6, the action *mousemove* then the action *click* on a link; the edge 12, action *mouseover* then action *mousemove* on the same button,


(a)

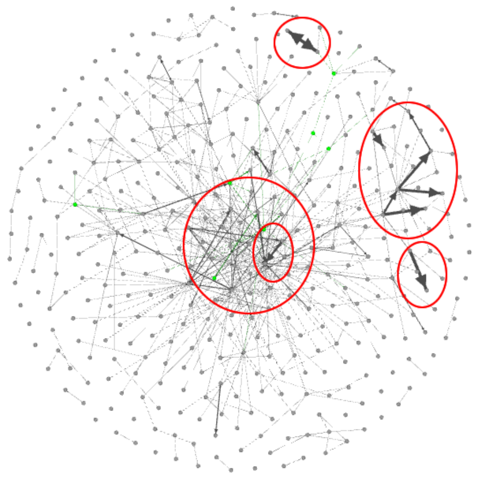

(b)

Figure 3. The generated graph by applying the algorithm USAGRAPH on data of the e-learning site excluding actions that do not represent a user's decision.

among others. But at the same table, a few pairs of actions draw attention when compared to the optimal path of the task. For example, in the edges 1 and 2, which are the heaviest edges in the graph, the actions were executed on a page (PEListaLinks.asp) that is not on the optimal path of the task. This also occurs in the edges 7, 11 and 16. Analyzing the structure of pages of e-learning, it was observed that these actions are not related to the correction of activities, these actions are so unusual. Furthermore, it is also possible to observe that, in the edges 4 and 5, there is the action *scroll*, which refers to the use of the scroll bar on the page. Observing the e-learning interface, it was revealed that *ASAlunosQuestao.asp* page shows the activities of all students, requiring the scrolling continuously.

In the Table 2, which has vertices with greater degree (in-degree + out-degree) of the task (a), there is an agreement with the presented data in Table 1, but the format of Table 2 emphasizes the most performed actions during the trying task execution, instead of pairs of actions presented in Table 1.

Through step 5 of USAGRAPH, to detect more common edges and vertices may help to identify more performed actions. However, como the Table 1 apresenta apenas os pares de aes mais executados, difcil and 2 difficult the visualization of sequences of actions likely traveled by users. To solve this problem, Table 3 presents a subset of the results of step 6 of USAGRAPH, listing the maximum found path between two intermediate actions of the optimal path of the task (a).

To allow a specific analysis of user behavior in interface, maximum and minimum found paths were observed between each pair of actions of the optimal path of tasks, i.e., the paths were observed between actions 1 and 2, 2 and 3, 3 and 4, and so on. Thus, it was possible to analyze more performed actions (obvious or not) in each step of the task, facilitating the detection of possible deviations from the optimal path, for example.

Table 3 presents the maximum found path by USAGRAPH between actions 4 and 5 of the optimal path of the task (a). Analyzing the presented data in this table, there is a predominance of actions *scroll*, *mousemove* and *mouseover* between opening the page with the list of student activities (initial action) and clicking the image to correct an activity (final action).

The analysis illustrated above was applied for the entire set of returned information by USAGRAPH. Com o apoio de um especialista do e-learning estudado, os padres de aes detectados pelo USAGRAPH foram comparados com os passos dos caminhos timos definidos pelo avaliador. Assim, as seguintes concluses foram alcanadas about user behavior on each task of e-learning site.

Table 1. Subset of heavier edges of task (a)

#	Weight of edge	Type of action	Object TAG	Object	Page
1	420	keydown	INPUT	txtURL	PEListaLinks.asp
		keypress	INPUT	txtURL	PEListaLinks.asp
2	407	keypress	INPUT	txtURL	PEListaLinks.asp
		keydown	INPUT	txtURL	PEListaLinks.asp
3	195	mousemove	HTML		ASAlunosQuestao.asp
		scroll			ASAlunosQuestao.asp
4	150	mouseover	HTML		ASAlunosQuestao.asp
		scroll			ASAlunosQuestao.asp
5	117	mousemove	A		ASAlunosQuestao.asp
		click	A		ASAlunosQuestao.asp
6	86	mousemove	IMG		PEListaSessaoDisc.asp
		click	IMG		PEListaSessaoDisc.asp
7	76	click	A		ASAlunosQuestao.asp
		blur			ASAlunosQuestao.asp
8	75	mousemove	A		ASAlunosQuestao.asp
		mousemove	P		ASAlunosQuestao.asp
9	70	mouseover	IFRAME		ASAlunosQuestao.asp
		mouseover	DIV		ASAlunosQuestao.asp
10	68	mousemove	TD		PEListaSessaoDisc.asp
		mousemove	IMG		PEListaSessaoDisc.asp
11	61	mousemove	A		ASAlunosQuestao.asp
		blur			ASAlunosQuestao.asp
12	61	click	A		ASAlunosQuestao.asp
		mousemove	A		ASAlunosQuestao.asp
13	55	click	IMG		PEListaSessaoDisc.asp
		unload			PEListaSessaoDisc.asp
14	54	mousemove	A		ASAlunosQuestao.asp
		mousemove	DIV		ASAlunosQuestao.asp
15	52	keydown	BODY		ASAlunosQuestao.asp
		scroll			ASAlunosQuestao.asp
16	46	click	A		index.asp
		mousemove	A		index.asp

Task (a) Correct a student activity:

- Access to unrelated pages to the correction of activities: USAGRAPH detected the occurrence of action *openpage* in pages outside the optimal path of the task;
- Continuous use the scroll bar before completing the correction of an activity: USAGRAPH detected the occurrence of action *scroll* in 12 of the 38 interactions in which professors started this task;
- The mouse pointer positioned over links related to correction of activities that were not clicked: USAGRAPH detected the occurrence of actions *mousemove* and *mouseover* over links that do not belong the optimal path of the task (for instance, *File sent by the student* and *See previous answers*). However, the action *click* on the links was not returned by USAGRAPH, which suggests that there were clicked in the same proportion that users have positioned the mouse pointer over them.

Table 2. Subset of vertices of the task with a greater degree (a)

#	Count	Type of action	Object TAG	Object	Page
1	920	mousemove	A		ASAlunosQuestao.asp
2	854	keydown	INPUT	txtURL	PEListaLinks.asp
3	840	keypress	INPUT	txtURL	PEListaLinks.asp
4	736	mousemove	TD		PEListaSessaoDisc.asp
5	518	mousemove	HTML		ASAlunosQuestao.asp
6	510	click	INPUT	cmdSalvar	ASAlunosQuestao.asp
7	484	load			ASRespondeQuestaoDis.asp
8	484	openpage			ASRespondeQuestaoDis.asp
9	482	unload			ASRespondeQuestaoDis.asp
10	414	mousemove	DIV		ASAlunosQuestao.asp
11	388	mouseover	HTML		ASAlunosQuestao.asp
12	380	mouseover	INPUT	cmdSalvar	ASAlunosQuestao.asp
13	336	mousemove	IMG		PEListaSessaoDisc.asp
14	324	mouseover	IFRAME		ASAlunosQuestao.asp
15	284	mouseover	TD		PESessaoTurmaDisc.asp
16	276	click	A		ASAlunosQuestao.asp

Table 3. Example of maximum found path between the intermediate actions of the task (a)

Distance	Type of action	Object	Page
0	openpage		ASAlunosQuestao.asp
24	load		ASAlunosQuestao.asp
43	mousemove	P	ASAlunosQuestao.asp
50	scroll		ASAlunosQuestao.asp
420	mouseover	P	ASAlunosQuestao.asp
999	mouseover	DIV	ASAlunosQuestao.asp
1593	mousemove	DIV	ASAlunosQuestao.asp
1747	mousemove	SPAN	ASAlunosQuestao.asp
1842	mousemove	IMG	ASAlunosQuestao.asp
1865	click	IMG	ASAlunosQuestao.asp

Task (b) Send an e-mail to students of a class:

- Continuous use the scroll bar before they start filling the message subject: the maximum path identified by USAGRAPH before step 5 of the task, it was identified the occurrence of action *scroll*.
- access unrelated pages to the task: as well as the task (a), the maximum path identified by USAGRAPH emphasized the action *openpage* on pages that belong to the optimal path of the task.

Section 4 discusses the achieved results by USAGRAPH algorithm and its potential uses.

4. DISCUSSION

The literature reports many motivations and approaches using logs, emphasizing the importance of this subject in computing and business. One of the existing approaches is the analysis of user behavior in interfaces, driven by need to improve the interfaces of information systems. In web applications accessed by many users, the analysis of user behavior and improvement of the interface are essentials to the system acceptance and to facilitate the execution of tasks intended by users. For this, client logs provide fine-grained information about users' actions in client-side.

Collected information at client side generates a large amount of data that needs to be processed in an efficient way, according to the expected response time. Towards to solve this problem, we proposed an algorithm, USAGRAPH, that explore client logs and uses visualization techniques and graph analysis, providing information to evaluate user behavior in web applications. The proposed algorithm uses the result of the task analysis and collected data by the system USABILICS,[1] in order to filter the relevant actions of the client logs according to the interest of the evaluator of the web application. The execution of the steps of the algorithm emphasizes the most performed actions by users during interaction, in addition to inform about the more performed actions between the beginning and the end of the task.

A case study was conducted in a real web application in order to validate the steps of the algorithm USAGRAPH. For the case study, we selected an e-learning and two tasks were defined as aspects of interest to evaluation of the professors' behavior: correction of a student activity and sending an e-mail to students of a class. After analyzing the generated information by the algorithm, it was possible to identify obvious actions and especially unrelated actions to the tasks.

In summary, the algorithm USAGRAPH identifies: (I) the more relevant actions performed by users for each aspect of interest defined by the evaluator; (II) the existence of patterns of actions through graph visualization; (III) the edges and vertices of patterns visually identified in graphs; e (IV) the paths often traveled by users to accomplish a task in interface.

With these results, it is still possible to analyze user behavior on web applications for some purposes, such as:

- To evaluate the usability of the interface: based on the returned information by USAGRAPH, it is possible that the evaluator of the web application notice the existence of unusual actions and he may find the reasons of these actions through observing the application interface.
- To provide information for web page personalization: detecting actions that do not belong to optimal path of the task suggests potential actions that users will run. So, personalize web pages from the user behavior can help him during navigation.

5. CONCLUSION AND FUTURE WORK

The presented algorithm provides relevant information for the analysis of user's behavior in web applications, highlighting the predominant actions while performing a task. However, we recognize it is still needed some human effort to interpret the information provided by the algorithm. Therefore, in future work we intend to define the relevance of the identified actions using semantic analysis. We expect that this improvement helps the system to provide an automatic interpretation of the graphs. Furthermore, we intend to classify users' interactions regarding expertise in a web application, that is, inferring the level of knowledge the user has about the web application. We also believe that is necessary to refine the visualization of the information generated by USAGRAPH, reducing the amount of information presented to the evaluator.

ACKNOWLEDGMENTS

The authors would like to thank the Brazilian agencies PROAP and CNPq for the financial support.

REFERENCES

[1] Vasconcelos, L. G. and Baldochi, L. A., "USABILICS: remote usability evaluation and metrics based on task analysis (in portuguese)," in [*Proceedings of the 10th Brazilian Symposyum on Human Factors in Computer Systems & 5th Latin American Conference on Human-Computer Interaction*], 303–312 (2011).

[2] Eirinaki, M. and Vazirgiannis, M., "Web site personalization based on link analysis and navigational patterns," *ACM Trans. Internet Technol.* **7** (Oct. 2007).

[3] Fraternali, P., Lanzi, P. L., Matera, M., and Maurino, A., "Model-driven web usage analysis for the evaluation of web application quality," *J. Web Eng.* **3**, 124–152 (Oct. 2004).

[4] Gauch, S., Speretta, M., Chandramouli, A., and Micarelli, A., "The adaptive web," ch. User profiles for personalized information access, 54–89, Springer-Verlag, Berlin, Heidelberg (2007).

[5] Liu, H., He, J., Gu, Y., Xiong, H., and Du, X., "Detecting and tracking topics and events from web search logs," *ACM Trans. Inf. Syst.* **30**, 21:1–21:29 (Nov. 2012).

[6] Yu, H., Liu, Y., Zhang, M., Ru, L., and Ma, S., "Web spam identification with user browsing graph," in [*Proceedings of the 5th Asia Information Retrieval Symposium on Information Retrieval Technology*], *AIRS '09*, 38–49, Springer-Verlag, Berlin, Heidelberg (2009).

[7] Zhou, J., Ding, C., and Androutsos, D., "Improving web site search using web server logs," in [*Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research*], *CASCON '06*, IBM Corp., Riverton, NJ, USA (2006).

[8] Santana, V. F. and Baranauskas, M. C. C., "Summarizing observational client-side data to reveal web usage patterns," in [*Proceedings of the 2010 ACM Symposium on Applied Computing*], *SAC '10*, 1219–1223, ACM (2010).

[9] Atterer, R., "Logging usage of ajax applications with the usaproxy HTTP proxy," in [*Proceedings of the WWW 2006 Workshop on Logging Traces of Web Activity: The Mechanics of Data Collection*], (2006).

[10] Balbo, S., Goschnick, S., Tong, D., and Paris, C., "Leading web usability evaluations to WAUTER," in [*Proceedings of the Eleventh Australasian World Wide Web Conference*], (2005).

[11] Paganelli, L. and Paternò, F., "Intelligent analysis of user interactions with web applications," in [*Proceedings of the 7th international conference on Intelligent user interfaces*], *IUI '02*, 111–118, ACM (2002).

[12] Vasconcelos, L. G. and Baldochi, Jr., L. A., "Towards an automatic evaluation of web applications," in [*SAC '12: Proceedings of the 27th Annual ACM Symposium on Applied Computing*], 709–716, ACM, New York, NY, USA (2012).

[13] Fruchterman, T. M. J. and Reingold, E. M., "Graph drawing by force-directed placement," *Softw. Pract. Exper.* **21**, 1129–1164 (Nov. 1991).

[14] Dijkstra, E. W., "A note on two problems in connection with graphs," *Numerische Mathematik* **1**(1), 269271 (1959).

[15] Vasconcelos, L. G. and Baldochi, Jr., L. A., "Usatasker: a task definition tool for supporting the usability evaluation of web applications," in [*Proceedings of the IADIS International Conference on WWW/Internet 2012*], 307–314, IADIS, Madri, Spain (2012).