An Episode-based Approach to Identify Website User Access Patterns

Madhuka Udantha, Surangika Ranathunga and Gihan Dias
Department of Computer Science and Engineering, University of Moratuwa, Colombo, Sri Lanka

Keywords: Web Usage Mining, Pattern Mining, Regular Expressions, Anomaly Detection.

Abstract: Mining web access log data is a popular technique to identify frequent access patterns of website users. There are many mining techniques such as clustering, sequential pattern mining and association rule mining to identify these frequent access patterns. Each can find interesting access patterns and group the users, but they cannot identify the slight differences between accesses patterns included in individual clusters. But in reality these could refer to important information about attacks. This paper introduces a methodology to identify these access patterns at a much lower level than what is provided by traditional clustering techniques, such as nearest neighbour based techniques and classification techniques. This technique makes use of the concept of episodes to represent web sessions. These episodes are expressed in the form of regular expressions. To the best of our knowledge, this is the first time to apply the concept of regular expressions to identify user access patterns in web server log data. In addition to identifying frequent patterns, we demonstrate that this technique is able to identify access patterns that occur rarely, which would have been simply treated as noise in traditional clustering mechanisms.

1 INTRODUCTION

In a website, web access logs record user navigations and other activities such as searching, site registering and editing web pages. Access logs are semi-structured files. Log records consist of user Internet Protocol (IP) address, HTTP status code, timestamp, web request URI, user agent, HTTP referrer and HTTP request type. The log records are distributed and occupy a large volume. A web session is a collection of web requests from a user with a unique IP address during a specified period of time. Web sessions can be very long. Therefore processing and web log mining can be difficult. Even though web log files contain valuable information, above reasons make it less viable to manually extract this information. A method to extract this valuable information would help detect anomalies and help website administration on Human Computer Interaction (HCI) improvements.

In web usage mining, there are many methodologies to identify user access patterns. These include clustering, Apriori algorithms, web access pattern tree (WAP-tree) and mining frequent patterns (Facca and Lanzi, 2005). But in these methods it is difficult to determine how each session deviates from other sessions in the same group or cluster. It is important to find these slight differences between sessions. These seemingly insignificant changes could be the most important for a domain expert as they may resemble anomalies such as an attack.

One of the limitations in k-means (Hartigan and John, 1979) and other clustering mechanisms is that they remove noise from the data(Fu and Sandhu, 1999). Here important data or patterns might be lost. For example, an attacker's access sequence might be discarded as noise since it may be considered as an outlier. When data contains noise, the completeness of the system lowers due to data being lost as noise. One technique to identify these anomalies is by only considering outliers, but this will not work if the pattern is inside a cluster. If insider a cluster, the slight difference between the attacker's access pattern is not obvious.

A session is a collection of episodes. An episode is a sequence of access pages that has a semantic meaning (Srivastava and Cooley, 2000). For example in a commercial website, an episode would refer to payment. This involves pages {payment page, enter credit card details, confirmation}.Similarly, login episode contains pages {home page, enter credentials, user account} and purchase episode contains pages {shopping cart, add items to cart, checkout, confirm} (page access occurs in the given order).

This paper presents a mechanism to identify these access patterns using regular expressions (regex) (Garofalakis and Rastogi, 1999), which is a technique very commonly used for similar tasks in domains such as genetic algorithms and text searching (Goldberg, 2006). A regex is a special text string for describing a particular pattern (Liang and Tianyi, 2014). When an episode is represented as a regular expression, it reduces the volume of session data. When the episodes are represented by regular expressions, slight changes to major changes can be easily identified. Slight changes can be detected through the repetitions and alterations of individual pages. Major changes can be detected by repetitions and alternation of page groups.

The identified episodes are stored in a suffix array that keeps track of the episode count. This is an improved version of normal suffix array (Manber and Myers, 1991).

System results are evaluated quantitatively and qualitatively and they were conducted on three web sites from different domains: non-profit organization, financial institution and educational institution. Interesting patterns and attacks (Chandola et al., 2009) were identified and some of them were not even known by the domain experts.

Rest of the paper is organized as follows. Section 2 reviews related work regard to web log pre-processing, web session clustering, regular expressions and episodes (sub sequences). In section 3 we describe the implemented system. Section 4 presents the evaluation and section 6 concludes the paper.

2 RELATED WORK

There are many web-based sites to cater for unending user requirements. These websites contain user access logs that contain important information such as access sequence patterns and web traffic. Web mining and web session mining discover and extract the above hidden information (Fu and Sandhu, 1999).

Before web session mining, web log data cleaning and pre-processing is needed. Web log pre-processing contains two main units, cleaning log data and user session identification. The log data cleaning steps include data selection, normalization, cleaning and transformation (Aye, 2011). Web log data pre-processing is a complex process and takes more than half of the time of the total log mining process (Cooley et al., 2013). The purpose of data cleaning is the removal of outliers or irrelevant data.

User session identification unit consists of user identification, session identification and user access path completion (Srivastava and Cooley, 2000). This is the most common log pre-processing order in web usage mining.

2.1 Web Session Mining Techniques

Most of the related research efforts focus on three main paradigms: association rules (Hipp et al., 2000), sequential patterns (Iváncsy and Vajk, 2006), and clustering (Langhnoja et al., 2013). Clustering has been widely used in web usage mining to group similar sessions. For instance divisive hierarchical clustering techniques are used to group web site users according to their interests (Kim and Chan, 2003). Genetic Algorithms are used to improve the results of clustering through user feedback (Holland, 1992).

Clustering based techniques rely on the following assumption: normal data instances lay close to their closest cluster centroid, while anomalies are far away from their closest cluster centroid (Nazeer and Sebastian, 2009). This technique declares any data instance that does not belong to any cluster as anomalous. A disadvantage of such techniques is that they are not optimized to find anomalies, since the main aim of the underlying clustering algorithm is to find clusters. If any attacker or malicious user performs as a normal user, they will be in the same cluster with the normal user, and will not be identified.

Association rule mining and clustering are combined into a method called association rule hypergraph partitioning(Han and Karypis, 1998).This algorithm can handle higher dimensional data sets, but it can only find frequent item sets that have support greater than a specified threshold and it could lose some interesting patterns.

Mining web log datasets has been extensively studied using Frequent Pattern Mining (FPM) (Han et al., 2007). Since item sequences are not considered in these studies, interesting patterns are missed.

The studies of sequential pattern mining have been extended in several different ways. (Yun and Leggett, 2006) proposed an approach with weighted sequential pattern mining in large sequence databases (WSpan) to push the weight constraints into the sequential pattern growth. WSpan introduces a weight range and weight is taken from a defined weight range, but still it has a limitation as the range is pre-defined. W-support (weight support) measurement(Sun and Bai, 2008) is not recommended for dense data sets as well. In sequential pattern mining, researchers face pattern growth problem (Yu and Korkmaz, 2015). In sequential mining, huge numbers of patterns are detected and the length of the patterns is also large.

Anomaly score of a data instance is computed as the sum of its distances from its k nearest neighbours (Wang and Tsong-Li, 1994). Defining distance measures between instances can be challenging when the data is complex and malicious users also act like normal users. In nearest neighbour techniques, if the data has normal instances that do not have enough close neighbours or if the data has anomalies that have enough close neighbours, the technique fails to label them correctly, resulting in missed anomalies.

Density based anomaly detection techniques such as local outlier factor (Pokrajac and Hartford, 2007), and Distance-based outliers (Knorr et al., 2000) estimate the density of the neighbourhood of each data instance. An instance that lies in a neighbourhood with low density is declared to be anomalous while an instance that lies in a dense neighbourhood is declared to be normal. Density based techniques perform poorly if the data has regions of varying densities. In web logs there are many areas of varying densities as there are multiple levels and groups of users for web sites.

2.2 Episodes in Sequences and Regular Expressions

Episode is not a new concept in the domain of web usage mining. Frequent episodes have been identified in sequences, where episodes specify the temporal before-and-after relationship, but without timing-interval restrictions (Mannila et. al., 1995). WebSIFT theoretically presents episode identification for web session mining, which is used in sequential pattern mining (Srivastava and Cooley, 2000). However, this technique has not been implemented. (Yu and Korkmaz, 2015) were able to identify general structures spanning over multiple sequences by detecting the most common sub-sequences (e.g., the pages visited). They are not able to handle repeating (looping) sequences.

Regular expression is a sequence of symbols and characters expressing a pattern, mainly for use in pattern matching with strings (Liang and Tianyi, 2014). The simplest regular expression is a single literal character except for the special meta characters *+?()|. The use of regular expressions has been presented as a sequential pattern mining process (Han et al., 2007). However, regular expressions are not used in web session pattern detection. The suffix array (SA) (Manber and Myers, 1991) of any string T is an array of length |T | such that SA[i] = j, where T [j : |T |] is the i-th lexicographically smallest suffix of T. The suffix array for any string of length |T | can be constructed in O(|T |) time (Kasai et al.,1997). Suffix array is a popular data structure in genetic algorithms to find the longest pattern (Huo et al., 2014). But in web sessions we need to identify all the patterns in sessions and they are not as long as genetic datasets.

3 EPISODE BASED SYSTEM

Our system contains three modules. First is the web log data cleaning module, which is responsible for purifying the web access log data by removing unwanted parameters. Web sessions are built from this module.

The second module generates regular expressions by processing web sessions. These regular expressions are indexed with the count of results.

The final module groups the regular expressions by looking at their similarity and the count of their occurrences in a web session. Those regular expressions are indexed and stored in suffix arrays. Suffix array used in this work is improved with the occurrence count. Therefore most common regex can be found easily. The groups represent episodes that have semantic meaning to understand the website user activities. Sessions are updated with episodes. Figure 1 shows the modular architecture of the implemented system.

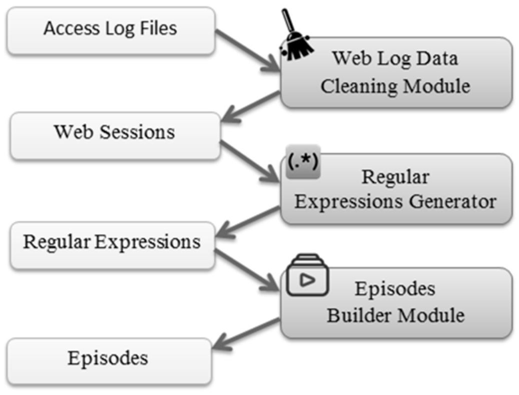

Figure 1: System architecture.

3.1 Web Log Data Cleaning Module

Web log data cleaning module is responsible for cleaning web log data and identifying web user sessions. We read the server access logs and filter the log records. There are access logs with different sizes that are created daily or weekly. Filtered and structured log records are appended to one file called ‘log’. Log data are cleaned using three steps as shown in Figure 2. Sessions are generated by grouping ‘IP’ or 'IP' and 'Agent'. These web user sessions are included in a session file. Each URL is given a unique number (integer). Mapping file contains the mapping of these integer numbers to the URLs.The mapping file is sorted and the similarities between the URLs are found by looking at argument length and argument name in the URLs. Once these arguments are removed, only one entry per URL is left on the mapping file.

The most requested pages for a time period can be retrieved by counting the number of web page requests. The new mapping file for most common pages (page occurrence count can be adjusted) is built.

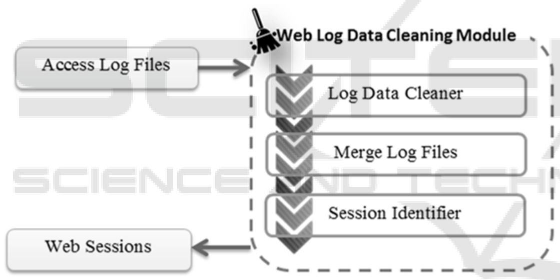

Figure 2: Web log data cleaning module.

3.2 Regular Expressions Generator

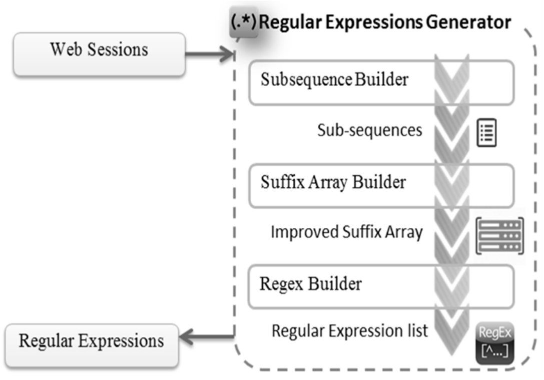

Figure 3: Regular Expressions Generator.

The regular expressions generator in Figure 3 contains three components. They are session subsequence builder, suffix array builder and regex builder.

Output of the session subsequence builder, i.e. sub-sequences (page sequences) are stored in a list similar to improved suffix array. The suffix array builder outputs an improved suffix array of sub-sequences and the regex builder outputs a list of regular expressions.

As explained in section 2, suffix arrays have been commonly used to identify the longest pattern in genetic algorithms(Pham and Duc, 2012) and on-line string searches (Wang et al., 2012). When using suffix arrays for web sessions, the limitation of suffix array approach is that it does not uniquely identify page sequences that are situated in the middle section of a session. Therefore session subsequence builder generates an sub-sequences (Sidorov et al., 2014) from sessions. Lengths of sub sequences are configured manually or in default mode it has a fixed range from 2 to the length of the sessions.

The sub-sequences are stored with a suffix. We introduce new feature column for suffix array called count that represents the occurrence count. When there is a suffix count for suffix array, it reduces the suffix array length and enables to find sub-sequences distribution as well. It is a data structure used, among others, in full text indices, data compression algorithms and within the field of bioinformatics (Arnau et al., 2014). In here we are using this technique for page sequences compression and to increase performance of our system.

Table 1 represents a sample session (session no. 101) in a normal sorted suffix array. Table header i represents the string charter index and here it is the session page index. If we used sorted suffix array, we would not be able to find all the patterns in sessions. This is because some page sequences such as {34,34,23}, {34,12,11} are missed from the middle part of the session string.

Session no. 101: 12, 34, 34, 23, 11, 45

Table 1: Sorted Suffix Array.

Suffix	i
$	7
11, 45 $	5
12, 34, 34, 23, 11, 45 $	1
23, 11, 45 $	4
34, 23, 11, 45 $	3
34, 34, 23, 11, 45 $	2
45 $	6

A suffix is part of a user page access sequence in sessions. Table 2 shows an improved sorted suffix array with index; a unique number for suffix is assigned and it is built from sub-sequences. Suffix count is an important part in our system as it represents the commonness of user accesses. From count we can get the most common and uncommon user access patterns.

Session no 101: 12, 34, 34, 23, 11, 45

Table 2: Sorted suffix array from sub-sequences.

Index	Suffix	Count
1	11, 45	1
2	12, 34	1
3	12, 34, 34	1
4	12, 34, 34, 23	1
5	12, 34, 34, 23, 11	1
6	12, 34, 34, 23, 11, 45	1
7	23, 11	1
8	23, 11, 45	1
9	34, 23, 11, 45	1
10	34, 23, 11	1
11	34, 23	1
12	34, 34, 23, 11, 45	1
13	34, 34, 23, 11	1
14	34, 34, 23	1
15	34, 34	1

When a new subsequence is picked from a session, it is added to the sorted suffix array if it does not exist there. But if it exists, count of suffix is incremented by one. The suffixes (page sequences) are processed in the regular expression engine and regular expressions are its output.

We have used page sequences as strings. There are few syntax patterns for regex. Here we used traditional UNIX egrep regular expression syntax (Sidhu and Reetinder, 2007).

3.3 Regular Expression Episode Representation

Let web page P be denoted as P and Session S as S=P[1]P[2]...P[n] and let P[*i,j*] denote an episode of S ranging from *i* to *j*. A regular expression can be mapped to any page sequence and it can be called an episode.

When we map episodes to sessions, there can be alternations, repetitions and concatenations. Suppose e_1 ande_1 are two regexes from sub-sequences *n* of session *s*, and n_1 and n_2 are sub-sequences of session *s*. if e_1 matches n_1 and e_2 matches n_2, then $e_1|e_2$ matches n_1 or n_2, and e_1e_2 matches $n_1.n_2$.

For example, Figure 4 shows a session with page order accessed by a user. There are page repeating patterns as shown in sections (a) and (b). Regex for *a* is $[P_2,P_3](2)$ and for *b* is $P_8[2]$. Session s can be updated as in Figure 5. There are no overlapping episodes in the session in Figure 5. Regexes in the middle are also considered.

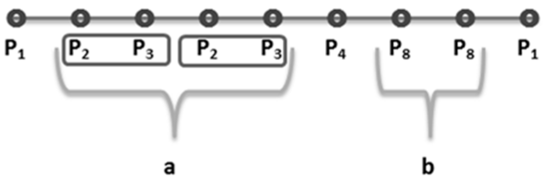

Figure 4: Sample session with page sequence.

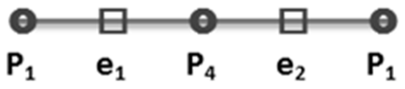

Figure 5: Session with episode representation.

4 EVALUATION

Here we present two quantitative and one qualitative experiment done during the system evaluation. A university website, a financial institution website and a non-profit organization website are considered in our evaluations. These website are denoted by U, F and N respectively from here onwards.

4.1 Results

Firstly we discuss the two quantitative experiments. In the first experiment, we cluster the three sites U, F and N using DBSCAN and k-means. First we run the algorithms with web page occurrences in the session. Then the algorithms are run with episode based sessions. Completeness is a standard way to verify the effectiveness of clustering mechanisms(Andrew, 2007). A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster.

Figure 6 shows the results on a bar chart. We can see that our episode-based approach gives a better performance with respect to completeness.

User sessions are built from log data of website U, F and N. We ran the k-means and DBSCAN clustering several times with different parameters. We got results with completeness. Figure 6 presents the best count for completeness for each cluster technique. Domain experts labelled the clusters and data records. Labelled data set is used to evaluate the system for completeness. Frequent pattern mining algorithm, given large amount of user groups (clusters) those groups are meaningless. Sequential pattern mining gave good results for the sample data (one week period), but for the whole dataset (four month dataset) gave performance issues due to the pattern growing problem.

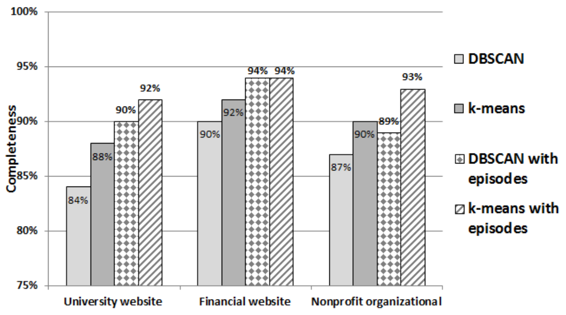

Figure 6: Completeness of Clustering algorithms and our system.

In the second experiment, we improved the suffix array by introducing the suffix count. This improvement reduced the length of the array. This is due to the array only containing unique suffixes with session count. So we can identify the most common suffixes of all the sessions.

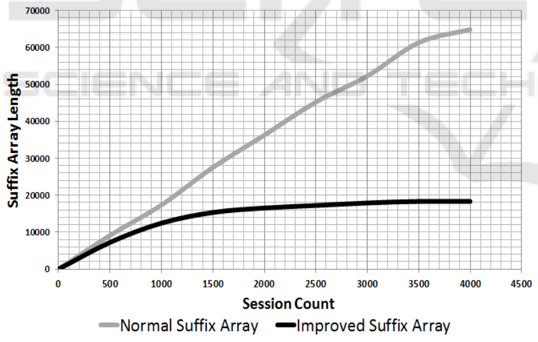

Figure 7: Suffix Array length growth for the two suffix array versions in website N.

As shown in Figure 7, the normal suffix array represents a linear relationship with session count and suffix array length. The improved suffix array comes to a steady state when session count is increased.

4.2 Identifying Attacks on a Website

In the qualitative experiment we demonstrate how we found two attacks and some other interesting user patterns in website N.

We have a list of episodes with regex representation and each episode has the occurrence count stored in the improved suffix array. The most common episodes can be identified from the count. Also we can find the most similar episode for any given episode.

By looking at count distribution we can find interesting patterns. If a particular episode has a less count value in the suffix array, and the similarity between this episode and others is less, then this episode could most probably be an attack. If regex similarity is high, count is low and episode repeating count is very high, then this also could be an attack. After inspecting the particular session and log record references (where the user was previously browsing) we can confirm whether this is an attack or not. Time gap between web requests sent can be used to confirm whether the interesting pattern belongs to a human user or a malicious tool.

Using the approach just discussed above, we detected some attacks and below we will explain them. An attack was detected and the attacker's web session sequence was like a normal user access pattern. In Figure 8 (a), normal user access pattern contains 1 to 4 episodes repeating and (b) attacker pattern also repeats the same episode (page sequences) but the repeating count was very high from 30 to 60.

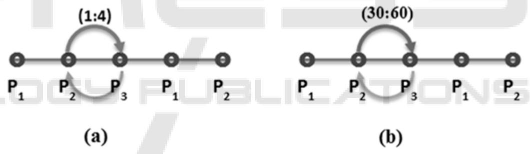

Figure 8: (a) Normal user pattern. (b) Attacker pattern.

The second attacker sends few requests to common pages such as home page and contact page denoted by p_1 and p_2 as shown in Figure 9. Then he sends requests to admin pages (p_{23}, p_{24}) and login pages (p_{25}), which a normal user does not request in this sequence. Web pages p_{26} and p_{27} are also admin pages that normal website user does not access. He repeats this process again for many times. Figure 9 shows the page request sequences.

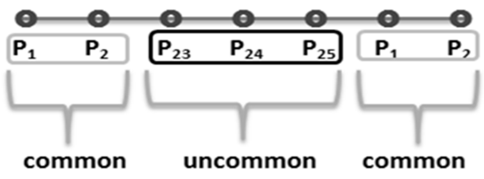

Figure 9: Sample web session attack.

Therefore we can see that the attackers camouflage themselves as common users but examining their user patterns can expose their malicious behaviour.

Above attack patterns are contained in one session or few sessions. Therefore DBSCAN or density techniques will not pick them as interesting patterns as they have low densities. In k-means those sessions are inside a cluster as they are similar to normal user patterns. Therefore above explained attacks or interesting patterns are not identified by density based technique, nearest neighbour analysis technique or cluster technique.

Common user patterns are nicely highlighted when we go through the access patterns.

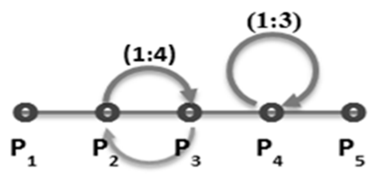

Figure 10: Search user access pattern, P_1 = home page, P_2 = feature list page, P_3 = search page, P_4 = Result page, P_5 = result detail page.

Figure 10 shows the search user access pattern. User comes to P_1(home page) then he moves to feature list page, P_2 where it lists down web site features. He picks the search feature from feature list page and he navigates to search page, P_3. He loops for 1 to 4 times between pages P_2 and P_3 as in Figure 10. After he finds the interesting results, he sorts and filters the results in result page, P_4. The Page P_4 is looped for 1 to 3 times. Then he finds the correct result and goes to the detail view of the result item P_5. Figure 10 represents the regular expression by P_1, $(P_2,P_3)\{1:4\}$,$P_4\{1:3\}$,P_5

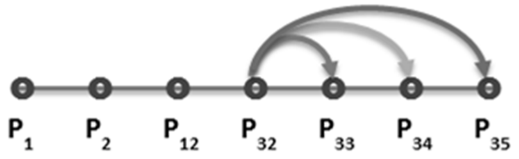

Figure 11: Article readers access patterns in website N, P_{32} = article list, P_{33}, P_{34} and P_{35} are article pages.

Figure 11 shows the access pattern of alternation. He navigates through P_1, P_2, P_{12} and P_{32}. P_{32} is page that list articles. User can select from the alternatives P_{33}, P_{34} and P_{35}, the article he prefers.P_1, P_2, P_{12}, $P_{32}(P_{33}|\ P_{34}|P_{35})$ is the regex of Figure 11.

Most of the users use only about 5% of the website functionality and some users loop through few of the web pages over and over again. This leads to better understanding of human interaction and interfaces of a web site. It brings about better designs in HCI and resolves navigational and other access issues in a website.

5 CONCLUSIONS

This paper presented an episode- based approach to identify website access patterns of users. Although regular expressions have been commonly used in many other domains to identify string patterns, this is the first time for them to be applied in identifying website access patterns. The use of regular expressions not only identifies the common access patterns, but also the rare access patterns, which could refer to anomalies such as attacks. We demonstrated that the completeness of this approach is considerably higher than the same of some popular clustering algorithms.

Currently we have not considered the '*not*' notation in regex. Therefore we cannot include them in our representations. In future we are going to implement the 'not' notation in our solution and cover all meta characters in regex such as [^], ?and \.

ACKNOWLEDGEMENTS

This research was supported by the LK Domain Registry, Sri Lanka. We thank our colleagues from Research Division of LK Domain Registry who provided insight and expertise that greatly assisted the research.

REFERENCES

Arnau, V. et al., 2014. Acceleration of short and long DNA read mapping without loss of accuracy using suffix array. , 30(23), pp.3396–3398.

Aye, T., 2011. Web log cleaning for mining of web usage patterns. *Computer Research and Development (ICCRD), 2011*.

Chandola, V., Banerjee, A. & Kumar, V., 2009. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, p.74.

Cooley, R., Mobasher, B. & Srivastava, J., 2013. Data Preparation for Mining World Wide Web Browsing Patterns. *Knowledge and Information Systems*, 1(1), pp.5–32.

EH Han, G Karypis, V Kumar, B.M., 1998. Hypergraph Based Clustering in High-Dimensional Data Sets : A Summary of Results. *IEEE Data Eng*, 21.1, pp.15–22.

Facca, F.M. & Lanzi, P.L., 2005. Mining interesting knowledge from weblogs: a survey. *Data & Knowledge Engineering*, 53(3), pp.225–241.

Goldberg, D.E., 2006. *Genetic Algorithms in Search , Optimization , and Machine Learning*, Pearson Education India.

Han, J. et al., 2007. Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery*, 15(1), pp.55–86.

Hartigan, John A., and M.A.W., 1979. A K-Means Clustering Algorith. *Applied statistics*, pp.100–108.

Hipp, Jochen, Ulrich Güntzer, and G.N., 2000. Algorithms for Association Rule Mining – A General Survey and Comparison. *ACM sigkdd explorations newsletter 2.1*, pp.58–64.

Holland., J., 1992. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, MIT press.

Huo, H. et al., 2014. A Practical Implementation of Compressed Suffix Arrays with Applications to Self-Indexing. *2014 Data Compression Conference*, (2), pp.292–301.

Iváncsy, R. & Vajk, I., 2006. Frequent Pattern Mining in Web Log Data. , 3(1), pp.77–90.

J Srivastava, R Cooley, M.D., 2000. Web Usage Mining : Discovery and Applications of Usage Patterns from Web Data. *ACM SIGKDD Explorations Newsletter 1.2*, pp.12–23.

Kasai, T. et al., Computation in Suffix Arrays and Its Applications. , pp.181–192.

Kim, H.R. & Chan, P.K., 2003. Learning implicit user interest hierarchy for context in personalization. *Proceedings of the 8th international conference on Intelligent user interfaces - IUI '03*, p.101.

Knorr, E.M., Ng, R.T. & Tucakov, V., 2000. Distance-based outliers: algorithms and applications. *The VLDB Journal The International Journal on Very Large Data Bases*, 8(3-4), pp.237–253.

Langhnoja, S.G., Barot, M.P. & Mehta, D.B., 2013. Web Usage Mining to Discover Visitor Group with Common Behavior Using DBSCAN Clustering Algorithm. , 2(7), pp.169–173.

Liang, Tianyi, et al., 2014. A DPLL (T) Theory Solver for a Theory of Strings and Regular Expressions. *Computer Aided Verification. Springer International Publishing*, pp.1–22.

Manber, U. & Myers, G., 1991. Suffix arrays: A new method for on-line string searches.

Mannila, Heikki, Hannu Toivonen, and A.I.V., 1995. Discovering frequent episodes in sequences Extended abstract. In *The first Conference on Knowledge Discovery and Data Mining*.

MN Garofalakis, R Rastogi, K.S., 1999. SPIRIT : Sequential Pattern Mining with Regular Expression Constraints. *VLDB. Vol. 99*, pp.223–234.

Nazeer, K.A.A. & Sebastian, M.P., 2009. Improving the Accuracy and Efficiency of the k-means Clustering Algorithm. , I, pp.1–5.

Pham, Duc, and D.K., 2012. *Intelligent optimisation techniques- genetic algorithms, tabu search, simulated annealing and neural networks.pdf*, Springer Science & Business Media.

Pokrajac, D. & Hartford, E., 2007. Incremental Local Outlier Detection for Data Streams. , (April).

Rosenberg, Andrew, and J.H., 2007. V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. *EMNLP-CoNLL. Vol. 7*, pp.1–2.

Sidhu, Reetinder, and V.K.P., 2007. Regular expression matching can be simple and fast. *Perls Dev. Conf.*

Sidorov, G. et al., 2014. Syntactic sub-sequences as machine learning features for natural language processing. *Expert Systems with Applications 41.3*, (Cic), pp.853–860.

Sun, K. & Bai, F., 2008. Mining Weighted Association Rules without Preassigned Weights. , 20(4), pp.489–495.

Wang, L. et al., 2012. A Complete Suffix Array-Based String Match Search Algorithm of Sliding Windows. *2012 Fifth International Symposium on Computational Intelligence and Design*, pp.210–213.

Wang, Jason Tsong-Li, et al., 1994. Combinatorial pattern discovery for scientific data: Some preliminary results. *ACM SIGMOD Record. Vol. 23. No. 2*, pp.1–2.

Y Fu, K Sandhu, M.S., 1999. Clustering of Web Users Based on Access Patterns. *KDD Workshop on Web Mining. San Diego, CA. Springer-Verlag*.

Yu, X. & Korkmaz, T., 2015. Heavy path based super-sequence frequent pattern mining on web log dataset. *Artificial Intelligence Research*, 4(2), pp.1–12.

Yun, U. & Leggett, J.J., 2006. WSpan: Weighted Sequential pattern mining in large sequence databases. *2006 3rd International IEEE Conference Intelligent Systems*, pp.512–517.