Discovering Frequent Itemsets Over Event Logs Using ECLAT Algorithm

A.S. Sundeep and G.S. Veena

Abstract Event logs are files that can record significant events that occur on a computing device. For example, when a user logs in or logs out, when the device encounters an error, etc., events are recorded. These events logs can be used to troubleshoot the device when it is down or works inappropriately. Generally, automatic device troubleshoot includes mining interesting patterns inside log events and classifying them as normal patterns or anomalies. In this paper, we are providing a sequential mining technique named ECLAT to discover interesting patterns over event logs.

Keywords Event logs · ECLAT algorithm · Sequence patterns

1 Introduction

Today, devices are getting more and more ubiquitous. It is difficult to troubleshoot these devices when they are affected by an internal error. One simple solution is to use event logs to record each and every event that occur in these devices. These event logs can be of various formats. Some can be in non-structured format like text files. And some can be in semi-structured formats like CDF files, XML files, or even files with log extension (i.e., '.log' formats). These files can be used to analyze the device functionality. For example, any windows system provides event logging. To view these event logs, open event viewer under System and security settings and admin istrative tools inside control panel.

Analysis of log events can be done in different ways. Traditionally, this is used to be a manual process. The analysis team used to get the data from the log files in the format that they require understands the device behavior and generates report

A.S. Sundeep (✉) · G.S. Veena
M S Ramaiah Institute of Technology, Bengaluru, India
e-mail: sundeepas4u@gmail.com

G.S. Veena
e-mail: veenags@msrit.edu

M.S. Reddy et al. (eds.), *International Proceedings on Advances in Soft Computing, Intelligent Systems and Applications*, Advances in Intelligent Systems and Computing 628, https://doi.org/10.1007/978-981-10-5272-9_6

on their findings. Automated techniques include tool-based analysis like SWATCH [1], LogSurfer [2], and SEC [3]. Some other automated process involves clustering techniques [4, 5], which includes calculating distance between log files and classifying them. Some other techniques involve mining internal patterns inside a single log file and classify these patterns. Few such approaches use state machine approach to analyze log files [6]. Generally, variants of apriori [7] algorithms are used in log analysis. We made a study on comparison of apriori over ECLAT [8] and decided to use ECLAT for this technique.

In this paper, we use a sequential mining algorithm named éclat and discover interesting rules over a pattern of events. The major objective of the work was to detect anomalies over event logs. However, scope of this paper is restricted to mining interesting patterns. We have considered an example to illustrate this process of generation of all possible subsets of event pattern over event logs.

2 ECLAT Algorithm Over Event Logs

ECLAT algorithm is used to perform itemset mining [9]. ECLAT stands for Equivalence Class clustering And bottom up Lattice Traversal. The algorithm uses tidsets (generally called as transaction ID sets) to avoid generation of unwanted subsets that does not occur in the current context. ECLAT algorithm is applicable for both sequential as well as normal patterns. For the proposed technique, we apply ECLAT algorithm over sequential patterns which are separated by bounded events. Let us illustrate how ECLAT can be modified over event logs. Let E indicate the set of all events of an event log if E_1, E_2, E_3, E_4 € E (Table 1).

When this input is given, the algorithm has to transform these transactions into the following table format format. The right-hand side of Table 2 shows the TID sets for a specific event E_i € E.

In the next step, the algorithm provides the support for every event and is shown in Table 3.

Here if we consider the minimum support as 2 $E3$ will be eliminated and other Events will be considered as Frequent item set. This is illustrated in Table 4.

Table 1 Initial input to ECLAT

TID (sequence ID)	Event sequences
1	E_1, E_3
2	E_1, E_4
3	E_2, E_3, E_4
4	E_2, $E4$

Table 2 Generating TID lists

Events E_i € E	TID list
$E1$	{1, 2}
$E2$	{3, 4}
$E3$	{3}
$E4$	{2, 3, 4}

Table 3 Events with support

Events E_i € E	Support
$E1$	2
$E2$	2
$E3$	1
$E4$	3

Table 4 ECLAT output

Events E_i € E	Support
$E1$	2
$E2$	2
$E4$	3

Table 5 Generating TID lists for two itemsets

Itemset $\{E_i, E_j\}$	TID list	Support
$\{E1, E2\}$	–	0
$\{E1, E4\}$	$\{2\}$	1
$\{E2, E4\}$	$\{3, 4\}$	2

Table 6 ECLAT output of two item set

Itemset $\{E_i, E_j\}$	Support
$\{E2, E4\}$	2

The above process shown is for single events over a set of sequences. The next step is to include item sets with more than one event. Table 5 illustrates this scenario.

By observing Table 6, it is not possible to perform this process for three item sets. Final output of the algorithm is $\{E_1\}$, $\{E_2\}$, $\{E_4\}$, and $\{E_2, E_4\}$. In the next section of this paper, we have discussed about the implementation process using a sample event log.

3 Implementation of ECLAT Algorithm

In this paper, we are considering a sample device log which is shown in Fig. 1. Our initial step is to generate pattern boundaries, to separate out different sequential patterns. This can be done in different ways. One such technique is to use temporal relationship between patterns. Another approach is to manually identify boundary patterns and generate pattern sequences. In our case, we are using the second approach to generate pattern sequences. Figure 2 shows set of bounded patterns indicating start and end of events.

Implementation of the proposed technique requires three steps mentioned as follows:

1. Creating the event transaction matrix;
2. Deciding minimum support;

	V1	V2	V3	V4	V5	V6
1	236533	2015-07-20T15:34:30	581	Command Requested. CommandName : System log fil...	Developer	None
2	236532	2015-07-20T15:34:19	965	Command Completed. CommandName : Log viewer.	Developer	None
3	236531	2015-07-20T15:33:02	457	*List count: logContainerList[3998] : staticWorker[39...	Developer	None
4	236530	2015-07-20T15:33:02	123	*ListCount:3998 Start:0 End:3997*	Developer	None
5	236529	2015-07-20T15:33:01	817	*Inital Loading done event received*	Developer	None

Fig. 1 Sample event log

	V1	V2	V3	V4	V5	V6	V7	V8
1	236342	2015-07-20T15:31:53	133	Sound on: *TubeOverload.mid*	Service	DebugLowLevel	Information	50010700
2	236321	2015-07-20T15:31:53	31	Sound on: *EndOfExposure.mid*	Service	DebugLowLevel	Information	50010700
3	236316	2015-07-20T15:31:52	474	Sound on: *RadiationOn.mid*	Service	DebugLowLevel	Information	50010700
4	236281	2015-07-20T15:31:47	449	Sound on: *TubeOverload.mid*	Service	DebugLowLevel	Information	50010700
5	236256	2015-07-20T15:31:47	85	Sound on: *EndOfExposure.mid*	Service	DebugLowLevel	Information	50010700
6	236250	2015-07-20T15:31:46	510	Sound on: *RadiationOn.mid*	Service	DebugLowLevel	Information	50010700

Fig. 2 Boundary events

3. Applying ECLAT;

A. *Creating the event transaction matrix*

From the above data, every event is identified by an identifier under column V8. These identifiers categorize events into a number of categories. For example, two events indicating user log in and log out can have the same identifier. These two events can be considered as events under user session category. Similarly, if user enters a set of commands to input variables into memory, increment a memory pointer, etc., such events can be categorized as memory events and can have a same identifier. Using the help of these identifiers, we can generate boundary events as shown in Fig. 2.

Now by considering this bounded events, sequence of events can be generated from the above data and can be included inside a single table as shown below. Each sequence is assigned with a sequence ID or pattern ID; in the order, they are generated indicated by column V1. Figure 3 illustrates the generation of these sequences. Each sequence has a different length. Hence, they are called variant patterns.

The above patterns are generated by considering the identifiers present in the original data, which can indicate the event flow. After obtaining these event sequences, they should be converted into transaction matrix to avoid "NA" characters in the item sets. Figure 4 shows the transaction matrix that can be obtained from the above log sequence.

B. *Deciding support*

Before applying the ECLAT over transaction matrix, we need to decide the support value. The support value can be decided by an item frequency plot. If the plot indicates a specific support value to be used by the algorithm. This plot is

1	500107001025	500107001030	500610003532	500610003534	500610003587	501200033120	50120000000
2	500107001025	500100002041	502100000259	501000000230	502100000002	500107001025	NA
3	500107001025	501401304510	500100002036	501200000002	501200000001	501400002010	50230000053
4	500107001025	500610003532	500610003534	501000000230	501000000230	501200000002	50120000000
5	500107001025	500610003360	500100002035	500100002041	501000000230	502100000002	50010700102
6	500107001025	501401304510	501200000002	501200000001	500100002036	501400002010	50010000204
7	500107001025	500610003360	500100002035	500100002041	502100000002	501000000230	50010700102
8	500107001025	501200000002	501200000001	500100002036	500100002046	501401304510	50140000201
9	500107001025	502100000259	502100000002	501000000230	500107001025	NA	NA

Fig. 3 Sequence of events

```
 500100002035  500100002041  500107001025  500107001030  500610003111
"500100002041" "500107001025" "501000000230" "502100000002" "502100000259"
"500100002034" "500100002036" "500100002046" "500100002047" "500107001025"
"500107001025" "500610003111" "500610003112" "500610003532" "500610003534"
"500100002035" "500100002041" "500107001025" "500610003360" "501000000230"
"500100002034" "500100002035" "500100002036" "500100002046" "500100002047"
"500100002035" "500100002041" "500107001025" "500610003360" "501000000230"
"500100002034" "500100002035" "500100002036" "500100002040" "500100002041"
"500107001025" "501000000230" "502100000002" "502100000259"
"500100002034" "500100002036" "500100002041" "500100002046" "500100002047"
"500107001025"
```

Fig. 4 Transactions sets

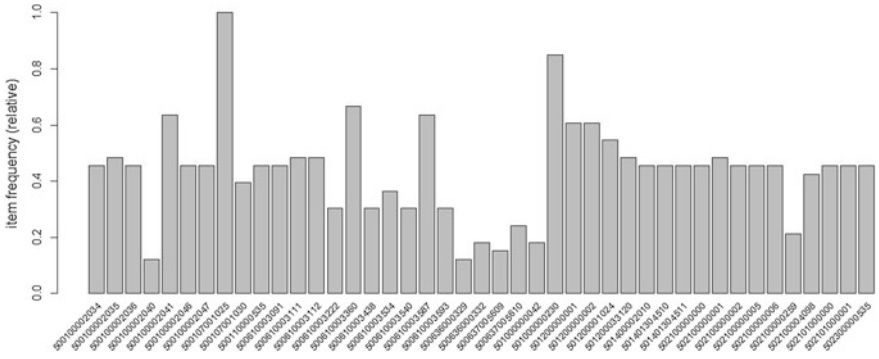

Fig. 5 Item frequency plot

shown in Fig. 5 under result section. Here, all the events appear in about 50% of the sequences. Hence, support can be kept nearer to 0.5. Hence in our paper, we have selected a support equal to 0.4.

C. *Applying ECLAT*

ECLAT is applied over the transaction matrix with a minimum support from the previous step, and the process follows the steps mentioned in previous section. The result of the algorithm also is shown in Fig. 5 in the result section.

items	support
{500107001025,501000000230,502100000002}	0.4545455
{500107001025,502100000002}	0.4545455
{501000000230,502100000002}	0.4545455
{500100002035,500100002041,500107001025,500610003360,501000000230}	0.4242424
{500100002035,500100002041,500107001025,500610003360}	0.4545455
{500100002035,500100002041,500610003360,501000000230}	0.4242424

Fig. 6 Frequent itemsets with minimum support

4 Results

In the previous section, we discussed about transaction matrix and an item frequency plot was generated from the transactions to decide the minimum support. This plot is shown in Fig. 5.

From Fig. 5, it can be noticed that most of the events appear in 30–50% of the mined patterns. Hence, minimum support can be selected anywhere between 0.3 and 0.5. In our case, we have selected 0.4.

After selecting the support, the transaction matrix and minimum support can be fed to the algorithm and the output result would be itemsets of various lengths. The result of our implementation is shown in Fig. 6.

5 Conclusion and Future Work

In this paper, we present how to use ECLAT algorithm over event logs. The result in Fig. 6 shows that the subsets of patterns that appear around 40–45% of the total events in the log file. However, the result shown is the head part of the result class. Many other subsets have around 50% and above support. Thus, our approach to select minimum support is valid. ECLAT algorithm provides appearance of every possible subset in terms of support. Thus, the output of the ECLAT algorithm suggests strong patterns that appear frequently in the event logs with respect to the minimum support selected.

After obtaining frequent patterns from the event logs, our further job would be to classify the patterns as anomalies or normal events. One way this can be done is by using Naïve Bayes filter and classify the events into anomalous or normal events which is a supervised learning technique, wherein we feed the system with known error patterns and normal patterns and classify the results. Another way is to consider time series analysis for anomaly detection. In our approach by observing the data anomaly detection over time series is difficulty as the difference between time is very less. Thus, we are using Naïve Bayes approach.

References

1. Hansen, Stephen E., and E. Todd Atkins. 1993. Automated System Monitoring and Notification With Swatch. In *Proceedings of the USENIX 7th System Administration Conference*.
2. Prewett, J.E. 2003. Analyzing Cluster Log Files Using Logsurfer. In *Proceedings of Annual Conference on Linu x Clusters*.
3. Rouillard, John P. 2004. Real-Time Log File Analysis Using the Simple Event Correlator (SEC). In *Proceedings of LISA XVIII*, 133–149 (Print).
4. Dickinson, W., D. Leon, and A. Podgurski. 2001. Finding Failures by Cluster Analysis of Execution Profiles. In Proceedings of ICSE.
5. Vaarandi, Risto. 2003. A Data Clustering Algorithm for Mining Patterns from Event Logs. In *Proceedings of the 2003 IEEE Workshop on IP Operations and Management*, 119–126.
6. Tan, J., X. Pan, S. Kavulya, R. Gandhi, and P. Narasimhan. 2008. SALSA: Analyzing Logs as State Machines. In *Proceedings of WASL*.
7. Pei, Jian, Jiawei Han, Behzad Mortazaviasl, and Hua Zhu. 2000. Mining Access Patterns Efficiently from Web Logs. In *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 396–407.
8. Kotiyal, Bina, Ankit Kumar, Bhaskar Pant, R.H. Goudar, Shivali Chauhan, Sonam Junee. 2013. User Behavior Analysis in Web Log Through Comparative Study of Eclat and Apriori. In *2013 7th International Conference on Intelligent Systems and Control (ISCO)*, 421, 426, January 4–5, 2013.
9. Kaur, Manjitkaur, Urvashi Grag. 2014. ECLAT Algorithm for Frequent Itemsets Generation. *International Journal of Computer Systems* 01 (03). ISSN: 2394-1065.