Association Rule Mining in System Event Logs to Discover Patterns

Rahul Gaikwad[1], Dr. Santosh Deshpande[2]
[1]Research scholar, MES'S Institute of Management and Career Courses Pune, India
[2]Director, MES'S Institute of Management and Career Courses Pune, India

Article Info
Volume 83
Page Number: 10544 - 10550
Publication Issue:
March - April 2020

Article History
***Article Received**: 24 July 2019*
***Revised**: 12 September 2019*
***Accepted**: 15 February 2020*
***Publication**: 13 April 2020*

Abstract

This research paper provides an overview of the current state of logs analysis in IT systems. Initial part covers some fundamental theory and summarises basic goals and techniques about system logs. The current software systems have been drastically evolving which are increasing in scale and complexity of software systems, that leads to a flood of logs. The traditional manual log inspection and analysis became impractical and almost impossible. As logs are unstructured in nature, the first important step is to parse the text log messages into structured and meaningful data for further processing and analysis. Correlation of diverse data and uncovering patterns and relationships in the data is a backbone of Artificial intelligence for IT operations (AIOps) field.

In this research paper, we present a comprehensive evaluation study on log events and discovering best association rules in logs to better understand and get more insight of logs events. More specifically, we evaluate more than a hundred log events spanning across distributed IT systems, hosts, customised services and application servers. We report the pattern discovery results in terms of association rules which gives practical importance when investigating and troubleshoot system issues.

***Keywords:** Log Analysis, Data Mining, AIOps, Pattern Mining, Association rules.*

1. INTRODUCTION

Logs are very important and play a crucial role in the software development and operations area. It is a standard practice to write detailed system runtime information into log files which allows developers and system administrators to understand the system behaviours and investigate problems.

1.1. Logs

Log file keeps recording the events that occur in OS and different system application(s). Logging is the act of keeping a log (record). In short, messages are written to a single log file which may consist of many events [1].

1.2. Common Types of logs [1]

1. **Application logs:** Developers have good control over Application logs. It can contain all types of events, error messages, warnings written by the application.
2. **Web and application server logs**: This log file record the activity of the client and all HTTP requests (called hits) made by web browsers.
3. **Garbage collector logs**: The garbage collector logs provide information about garbage collector activities.
4. **System logs**: Operating system writes specific events to System logs. These logs are also a right place to get details of external events.

1.3. Logging Levels[1]

1. **FATAL** - It indicates a critical service failure.
2. **ERROR** - It represents a disruption in a request or the ability to service a request.
3. **WARN** - It shows a non-critical service error.
4. **INFO** - It represents the state of the service.
5. **DEBUG** - It conveys extra information regarding life-cycle events.
6. **TRACE** - It is directly associated with activity that corresponds to requests.

2. RELATED WORKS

2.1 Log Mining

This paragraph summarizes the desired meaningful information which can get from the log files and where it can be applicable.

- General statistics (like average or max values, mean, deviations) which is useful for setting hardware requirements and accounting purposes.
- Program or system warnings (e.g. power or hardware failure, low memory, disk or CPU utilization) which helps in system maintenance or administration.
- Security related warnings can be leveraged for security testing / audits.
- Validation of program runs are helpful in software testing.
- Time related characteristics can be used for software profiling and benchmarking.
- Patterns and trends are getting applied for different data mining purposes.
- Behavioural trends used to determine performance and reliability

2.2 Association Rule used for Web Mining

In Website usage mining several data mining techniques are used. Association rules are used to discover the pages which are visited together even if they are not directly connected. It reveals associations between groups of users with specific interest or need. Using this insight, the trends of the activity of the users can be determined and predictions to the next visited pages can be calculated. [3]

2.3 Log Pattern Mining

In the log event files, many useful associations or patterns can be discovered using different data mining techniques, as described below and shown in *Figure 1*.

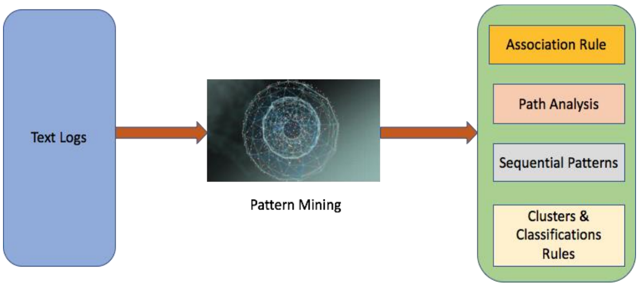


Figure 1: Pattern Mining in text log file

2.3.1 Association Rule:

It is used to predict the correlation of items where the presence of one set of items in a transaction or event implies the presence of other items.

2.3.2 Path Analysis:

Graph models are used for Path Analysis, which represents data in nodes and relationships format.

2.3.3 Sequential Patterns:

The sequence of items or events occurring in transaction has a particular order between the items or the events.

2.3.4 Clusters and Classification rule:

This technique groups profiles of items or objects with similar characteristics. This discovers the relationships.

3. PROBLEM:

DevOps or system administrators need to closely monitor various IT systems, application stacks and infrastructure. They have to go through all different systems, applications and database logs, to investigate and resolve all kinds of system issues like performance degradation or outages. Manually checking all logs is a very difficult and critical task. Also, it’s not possible to correlate the log events to understand the cause of system problems. A logging is an essential part of application support. By nature, logs are in unstructured text format. Most of the time developers simply use printf statements and concatenate strings to generate log messages. This logging has some drawbacks, it needs to parse the text message first to do log analysis, which is very complicated and expensive work. Collecting and combining through log data to identify a system issue is equivalent to searching for a needle in a haystack. In this case, someone may use a magnet to find that needle, likewise IT teams also need an easy way to search log files, correlate and interpret the log events. In this research, we are focusing on analysing IT computing system logs and finding the association (pattern) between different log events and their impacts on the system.

4. EXPERIMENTAL SETUP:

For this research work, we shall consider a sample set of 108 log events that have generated from different live hosts and services. Each log has a specific list of events. Here we have demonstrated the implementation of Apriori algorithm for association rule mining using a tool.

4.1 Dataset Insight

Hosts (Servers) - Collected sample sets of logs which are generated from 108 hosts (servers), out of that 32 hosts are distinct, shown in *Figure 2.* If we consider it as graph, X-axis represents different hosts whereas Y-axis represents number of instances.

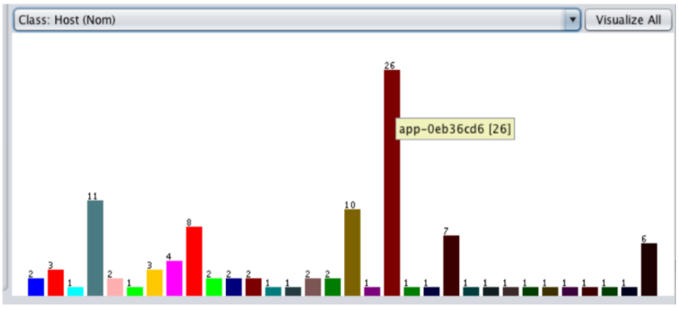


Figure 2: Hosts - From were log events generates

Services: There are distinct 10 different customized developed services which have generated these logs from different hosts (servers), shown in *Figure 3.* X-axis represents different customized services whereas Y-axis represents a number of instances in log data.

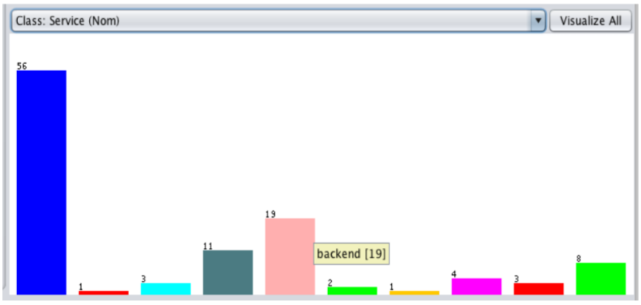


Figure 3: Services - who generates log events

Log Event: Collected 108 log data events as a sample set which are considered for this research experiment and out of that 32 are distinct events. As shown in *Figure 4,* X-axis represents different log events whereas Y-axis represents a number of instances in log data.

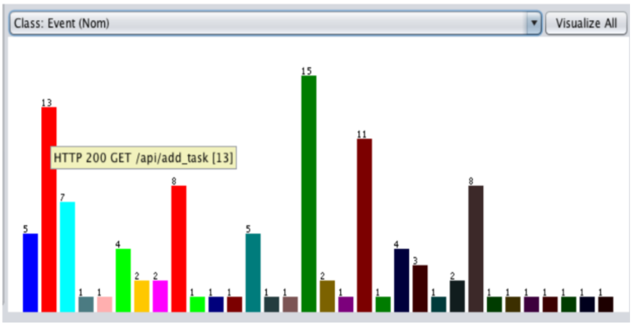


Figure 4: Events -which generated in log files

Status: Based on host, service and event, developers have given some labels to these events like error, debug, info, notice and warn. As shown in *Figure 5*, X-axis represents statuses whereas Y-axis represents a number of instances in log data.

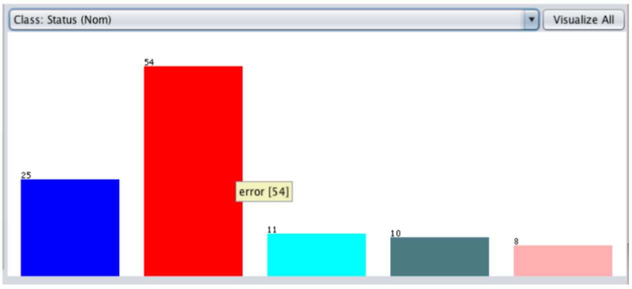


Figure 5: Status - Impact of log events on system

4.2. Log Parsing:

As an example, illustrated in *Table 1*, each system log event from sample dataset is printed by an application logging and records system events with its message header and content. The message header is generated by the logging framework therefore it can be easily extracted, such as date, service name, environment type, hostname, event (message) content and verbosity level (e.g., ERROR/INFO/DEBUG). [2]

Table 1: An Illustrative example of Log Parsing

date	2020-02-06T12:03:08.267Z
service	cont-link
env	prod
Host	app1.web.abc.com
event	executing query: ROLLBACK
status	debug

5. ASSOCIATION RULE MINING

Association (pattern) Mining is the most important data mining technique. Extracting association rules is the core fundamental of data mining [5]. The benefits of association rules are detecting and discovering unknown relationships or patterns, producing results based on decision making and prediction can be performed [5]. The discovery of association rules is divided into two phases [7] [8] - detection of the frequent itemset and generation of association rules. To find patterns (sequence) in the log events data, we focus on implementing below important association rule mining algorithms for the research experiment. [6]

Algorithms [10]**:**

There are four main important association rule algorithms. Out of that we will be focusing on Apriori algorithm 1. Apriori Algorithm:

2. FilteredAssociator algorithm
3. Predictive apriori algorithm
4. Tertius algorithm

Measures:

Support- It measures how often rules occur in the database. Formula to calculate support [4].

Number of occurrences {x, y}

Support =

Total Transaction in DB

Confidence - Support measures how often the rules occur in the database while confidence measures the strength of the rules. Formula to calculate confidence [4]:

Total occurrence for item X and Y

Confidence =

Total occurrence for item X

6. EXPERIMENTS

As shown in *Figure 6*, we implemented the Apriori association rule algorithm and found the best 20 association rules with *Support=50%* and *confidence=50%*. Below are the parameters, values and their description which has been applied during algorithm implementation. N = 20 =>Number of rules as an output

T =0 =>Rank rules

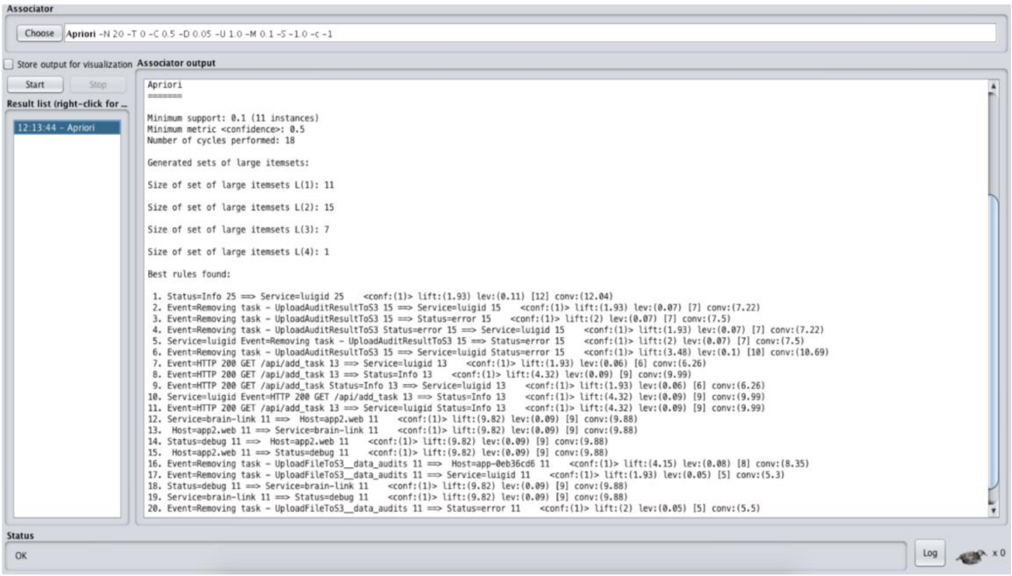
C= 0.5 =>Score of rule

D= 0.05 =>Delta for minimum support

U = 1.0 =>Upper bound for minimum support

M = 0.1 =>lower bound for minimum support

S = -1.0 =>significance levelc = -1 =>Class index



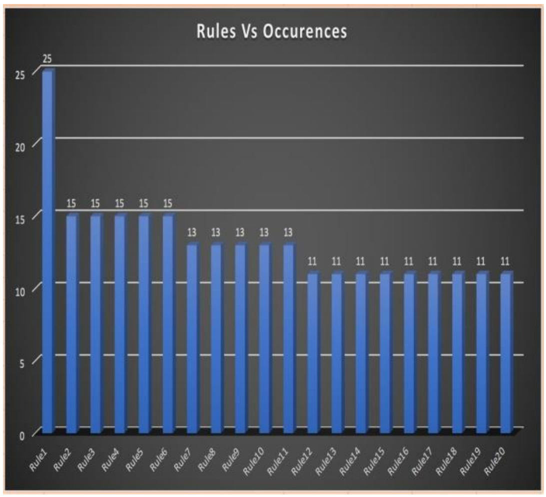
```
Apriori -N 20 -T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
```

```
Apriori
=======

Minimum support: 0.1 (11 instances)
Minimum metric <confidence>: 0.5
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 11

Size of set of large itemsets L(2): 15

Size of set of large itemsets L(3): 7

Size of set of large itemsets L(4): 1

Best rules found:

 1. Status=Info 25 ==> Service=luigid 25    <conf:(1)> lift:(1.93) lev:(0.11) [12] conv:(12.04)
 2. Event=Removing task - UploadAuditResultToS3 15 ==> Service=luigid 15    <conf:(1)> lift:(1.93) lev:(0.07) [7] conv:(7.22)
 3. Event=Removing task - UploadAuditResultToS3 15 ==> Status=error 15    <conf:(1)> lift:(2) lev:(0.07) [7] conv:(7.5)
 4. Event=Removing task - UploadAuditResultToS3 Status=error 15 ==> Service=luigid 15    <conf:(1)> lift:(1.93) lev:(0.07) [7] conv:(7.22)
 5. Service=luigid Event=Removing task - UploadAuditResultToS3 15 ==> Status=error 15    <conf:(1)> lift:(2) lev:(0.07) [7] conv:(7.5)
 6. Event=Removing task - UploadAuditResultToS3 15 ==> Service=luigid Status=error 15    <conf:(1)> lift:(3.48) lev:(0.1) [10] conv:(10.69)
 7. Event=HTTP 200 GET /api/add_task 13 ==> Service=luigid 13    <conf:(1)> lift:(1.93) lev:(0.06) [6] conv:(6.26)
 8. Event=HTTP 200 GET /api/add_task 13 ==> Status=Info 13    <conf:(1)> lift:(4.32) lev:(0.09) [9] conv:(9.99)
 9. Event=HTTP 200 GET /api/add_task Status=Info 13 ==> Service=luigid 13    <conf:(1)> lift:(1.93) lev:(0.06) [6] conv:(6.26)
10. Service=luigid Event=HTTP 200 GET /api/add_task 13 ==> Status=Info 13    <conf:(1)> lift:(4.32) lev:(0.09) [9] conv:(9.99)
11. Event=HTTP 200 GET /api/add_task 13 ==> Service=luigid Status=Info 13    <conf:(1)> lift:(4.32) lev:(0.09) [9] conv:(9.99)
12. Service=brain-link 11 ==>  Host=app2.web 11    <conf:(1)> lift:(9.82) lev:(0.09) [9] conv:(9.88)
13.  Host=app2.web 11 ==> Service=brain-link 11    <conf:(1)> lift:(9.82) lev:(0.09) [9] conv:(9.88)
14. Status=debug 11 ==>  Host=app2.web 11    <conf:(1)> lift:(9.82) lev:(0.09) [9] conv:(9.88)
15.  Host=app2.web 11 ==> Status=debug 11    <conf:(1)> lift:(9.82) lev:(0.09) [9] conv:(9.88)
16. Event=Removing task - UploadFileToS3__data_audits 11 ==>  Host=app-0eb36cd6 11    <conf:(1)> lift:(4.15) lev:(0.08) [8] conv:(8.35)
17. Event=Removing task - UploadFileToS3__data_audits 11 ==> Service=luigid 11    <conf:(1)> lift:(1.93) lev:(0.05) [5] conv:(5.3)
18. Status=debug 11 ==> Service=brain-link 11    <conf:(1)> lift:(9.82) lev:(0.09) [9] conv:(9.88)
19. Service=brain-link 11 ==> Status=debug 11    <conf:(1)> lift:(9.82) lev:(0.09) [9] conv:(9.88)
20. Event=Removing task - UploadFileToS3__data_audits 11 ==> Status=error 11    <conf:(1)> lift:(2) lev:(0.05) [5] conv:(5.5)
```

```
OK
```

Figure 6: Implementation of Apriori algorithm on log event data

After implementation of apriori algorithm and pattern discovery, we compared the generated rules and their occurrences as shown in *Graph 1*. We can observe that infrequent log events have been excluded by algorithm and gives only matching association rules as per given thresholds of support and confidence variables.



Graph 1: Rules and occurrences

7. RESULTS AND CONCLUSION

The main objective of this research paper is mining association (pattern) rules in log events. From the experiment and observation, we could solve the problem of manual log monitoring and manually finding the patterns in log events to understand the relationships in various events. Based on the experiment and results of association rules, we could understand the relationships and association between different log events (messages). It also provides more insights into how each host, services and events are correlated with each other and how it generates status results. This research work can be further enhanced by implementing more association rules algorithms, statistical measures and comparing with different algorithms on different parameters. Below is some conclusion which drawn from the experiment and results:

1. **Rule 1**: There is an association between status and service attributes, when status is info and service is luigid. Apriori algorithm found 25 matching instances.
2. **Rule 2-6**: Result shows, there are 15 matching instances, where *service:luigid* trigger *event:'Removing task - UploadAuditResultToS3'* which results into *status:error.*
3. **Rule 7-11**: It interprets from matching 13 instance that *event:'HTTP 200 GET /api/add_task'* get generated from *service:luigid* which has *status:info.*
4. **Rule 12-13**: There are 11 matching instances which represent association of *service:brain-link* and *host:app2.web.*In other words, *brain-link* service is hosted on *app2.web* server.
5. **Rule 14-15** : We can interpret that debug (status) mode has enabled on host app2.web, as there are 11 matching instances which generate this association rule.
6. **Rule 16-17:** There are matching 11 instances which represents, a *service:luigid* is hosted on *host:app-0eb36cd6* and it triggers *event: 'Removing task -*

   UploadFileToS3__data_audits'.
7. **Rule 18-19:** Service brain-link has debug mode enabled as there are 11 instances.
8. **Rule 20** : When there is an *event:'Removing task - UploadFileToS3__data_audits'* generated it triggers *status:error* , we have 11 matching instances.

REFERENCES

[1] Wikipedia https://en.wikipedia.org/wiki/ andhttps://stackify.com/java-logs-types/

[2] Jan Valdman, "Log File Analysis" , Technical Report No. DCSE/TR-2001-04 July, 2001

[3] S.VijayaKumar,A.S.Kumaresan, U.Jayalakshmi, Frequent Pattern Mining in Web LogData using Apriori Algorithm , International Journal of Emerging Engineering Research and Technology Volume 3, Issue 10, October 2015, PP 50-55 ISSN 2349-4395 (Print) & ISSN 2349-4409 (Online)

[4] L.K. Joshila Grace , V.Maheswari, Dhinaharan Nagamalai , ANALYSIS OF WEBLOGS AND WEB USER IN WEB MINING, International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.1, January 2011.

[5] F. H. AL-Zawaidah, Y. H. Jbara, and A. L. Marwan, "An Improved Algorithm for MiningAssociation Rules in Large Databases," Vol. 1, No. 7, 311-316, 2011

[6] Mohammed Al-Maolegi , Bassam Arkok, AN IMPROVED APRIORI ALGORITHM FORASSOCIATION RULES,International Journal on Natural Language Computing (IJNLC) Vol. 3, No.1, February 2014.

[7] R. Agrawal, T. Imieliński, and A. Swami, “Mining association rules between sets of items inlarge databases,” in ACM SIGMOD Record, vol. 22, pp. 207–216, 1993

[8] R. Srikant, “Fast algorithms for mining association rules and sequential patterns,” UNIVERSITY OF WISCONSIN, 1996.

[9] T. C. Corporation, “Introduction to Data Miningand Knowledge Discovery”, Two Crows Corporation, Book, 1999.

[10] Mukesh Sharma, Jyoti Choudhary, Gunjan Sharma,Associate.Professor, Assi.Professor, Mtech Scholar , Evaluating the performance of apriori and predictive apriori algorithm to findnew association rules based on the statistical measures of datasets , International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 6, August - 2012 ISSN: 2278-0181