*International Journal of Computer Applications (0975 – 8887)*
*Volume 135 – No.9, February 2016*

# Logical Itemset Mining Implementation on Hadoop

Karan Jawalkar
137, Somwar Peth
Pune-11

Avinash Patil
115, Sahakarnagar
Pune-09

Shreemay Panhalkar
24,Shivteertha
Pune-38

Raj Pande
115, Sahakarnagar
Pune-09

## ABSTRACT

Frequent Itemset Mining (FISM) finds the large and fre-quently occurring items from the datasets using Apri-ori algorithm. The FISM framework does not addresses two major properties that are Mixture-of property(more than one customer intent) and Projection-of property. To overcome the problems of irrelevant and non ac-tionable data and also to address the properties men-tioned above, Logical Itemset Mining (LISM) frame-work is introduced. LISM finds logical itemsets from the data which helps in eliminating non actionable data but at the same time keeps data which is log-ically connected. LISM not only finds logically con-nected items but aso items which are rarely occurring but logically connected are also discovered. LISM also addresses the Mixture of property and Projection of property which are not very well addressed in FISM.

## General Terms

Market Basket, Mixture of property, Projection of property

## Keywords

FISM, LISM, M/R Job, FLASK, LUCENE

## 1. INTRODUCTION

Market basket is the collection of items or things purchased by a customer in the market. When any customer visits a store or market and buys some items then the stuff bought by him/her forms the market basket. Data mining helps to find groups of related items from such market baskets.

Frequent Itemset Mining[1] was discovered in 1990s, which used Eclat[2] or Apriori algorithm[3] to find the large and frequent item-set in data input. This algorithm fits for limited data size but as data size increased day by day this algorithm was not scalable. FISM typically generates very large number of maximal frequent item-sets, most of which tend to be noisy or meaningless.

To tackle with this problem, different techniques were developed which helped to expand the application of FISM framework. FISM generates noisy itemsets in which very few are useful and action-able. The properties like Mixture-of and Projection-of are also not very well addressed in FISM. Mixture of property means that in a market basket there could be more than one customer intent. Pro- jection of property means that market basket contains subset of items associated with customer intent.

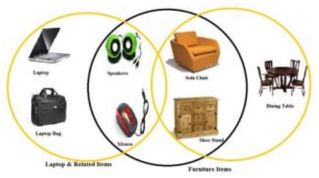

**Fig. 1. Market Basket**

These properties are addressed in Logical Itemset Mining(LISM). LISM[4] also helps in finding data which is really useful and ac-tionable .For understanding all this consider the figure in which solid black circle shows the market basket for a customer. In the fol-lowing example both properties can be viewed. The Logical Itemset Mining framework in which logically connected as well as rarely occurring data can be obtained and also both the properties of the data can be addressed in best way.

## 2. ALGORITHM

Logical Itemset Mining (LISM) is used to find the logically con-nected itemsets from the given bag of data. The properties like Mixture-of and Projection-of properties are very well addressed here. Unlike of the FISM , LISM stores rarely occurring data which sometimes might be logical. In short, LISM also takes into consid-eration rare itemset mining[5]. LISM framework has four stages:

1. Counting Stage: At this point, pair and single occurrences are calculated for each item in the input data. This step is implemented to find the co-occurrence counts between all pair of items. Here the co-occurrence counts are made which can be given as an input to the Consistency stage. In the counting stage we calculated two types of counts

a.Co-occurrence counts: in this step the count is made of items of type in which both items co-occur i.e the counting is done by seeing number of datasets in which both items co-occur. To see mathematically, f(a,b) = f(b,a).

b.Marginal Counts: in this step the count is done by seeing the number of pairs in which item occurred with some other item in the data. Here number of pairs are counted by seeing which item occurred with some other item in that dataset.

2. Consistency Stage: Now from the Counting stage we have two counts like Co-Occurrence and marginal count. Like FISM which depends on some threshold to find the most frequent items in the dataset, LISM also does the same thing. However, FISM decides threshold based on co-occurrence counts, while LISM uses a con-sistency value as well, which is derived from co-occurence counts and marginal counts. There are four measures given for the pur-pose of finding Consistent items. They are: 1.Cosine 2.Jaccard co-efficient 3.Point-wise Mutual Information 4.Normalized Point-wise Mutual Information Cosine formula has been used for the imple-mentation. It is as follows - f(a,b) = phi(a,b) / s qrt ( phi(a) * phi(b) ) where phi(a,b) = Co-occurrence count between items a and b phi(a) = Marginal count of item a The consistency value from the above formula is always between 0 to 1. If the value is 1 then it means that whenever the two items occur, they occur together. The greater the consistency value, the greater are their occurrences to-gether.

3. Denoising: In the denoising stage we reduce the itemsets by us-ing some threshold values for the consistency value, co-occurence count and marginal count. Those pairs which have any count or consistency value below the threshold are considered as noisy pairs and are thus discarded for further calculations. In FISM, only the two counts are used as threshold. Consistency is the main step in LISM, which helps to create logically related groups.

4. Discovery: Discovery is the important part in the LISM frame-work. In the above 3 stages we have reduced the mixture-of (multiple intents) noise in the data by ignoring low frequency co-occurrence counts, converting these counts into consistencies and then eliminating these consistencies by giving some threshold value. The remaining pairs are then considered for graph creation. The graph is represented in the form of adjacency list, which is then used to find cliques[5]. A clique is a set of maximum num-ber of nodes or a subset of the graph in which all the nodes are connected to each other. All such cliques are then discovered.

The bridge node plays an important part in the further part of LISM. If bridge node between two cliques is formed or obtained then cus-tomer can be redirected from one clique of items to another clique of items[6]. Bridge node and its example can be seen in the figure2.

## 3. IMPLEMENTATION

The implementation part is done accoridng to the LISM frame-work.The input file is a transactional data in which each line is one transaction containing comma-separated list of items.The input file is processed in preprocessing module.

Single occurence and pair occurence of different items are counted in the counting stage.Both these count values are stored in Re-dis database.In Consistency stage, the data from Redis database is taken and converted into consistency values.For finding Consis-tency values, cosine formula is taken.

In denoising stage we decide some threshold value manually to dis-card the item pairs having consistency value below it.

In Discovery stage, the denoised data is used to build a graph, which is represented in the form of adjacency list. This graph is the used to generate cliques. M/R job is run iteratively for this purpose, till all the cliques are generated.Once the clique and bridge nodes are found the resultant files are copied to local from the hadoop. A bridge node is a node which is present in more than one clique. This data is stored in Redis. Data is read from the resultant files and written to lucene database before first request comes to flask server. Then user can enter the product of his choice through web based UI. This request goes to the flask server and then flask server retrieves data from lucene database. Then resultant data will be sent to UI through AJAX.

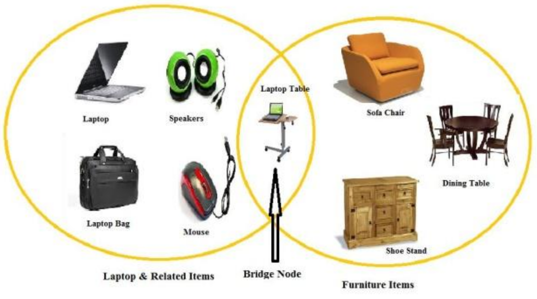

**Fig. 2. Clique with bridge node**

## 4. EXPERIMENTS

A Datasets used The dataset which has been used is the actual mar-ket basket dataset of the customers, which has total of 46,605 differ-ent transactions and 126 different products. Most items have high occurrence and thus following preprocessing has been done for it: Compute frequency of all items (here, products) Remove multiple occurrence of a item in one transaction Remove low frequence key-words from each itemset B. Logical Itemsets Discovered Our main objective is to find logically related itemsets which are of high qual-ity and less noise. There are some key properties of logical itemsets discovered: Large sizes: The number of frequent itemsets generated grow exponentially with the growth of itemset size. Here, LISM has been proved to be efficient for the purpose. Whereas, FISM consid-ers all the frequent itemsets and thus becomes more time complex. Meaningful itemsets: The itemsets discovered are meaningful as well as less noisy. And as the dataset size grows, it becomes more and more meaningful and perfect. Low frequncy: FISM considers only highly occuring pairs. But in LISM, rarely occurring, but log-ically related (occurring together) item pairs are also addressed and used to form cliques. Thus overall, we conclude that LISM gen-erateshigh quality, desirable itemsets while FISM produces a very large number of noisy, undesirable itemsets from the same data.

| **Occurence Count in Cliques** | **Bridge Node .** |
|---|---|
| 51 | display storage |
| 41 | drawers storage |
| 35 | computer tables study tables |
| 35 | shoe racks |
| 33 | kitchen storage |

| Size | Itemset . |
|---|---|
| 13 | adam book case oak,center corner tables,computer tables study tables,display storage,drawers storage,entire solid wood range,eon 2 door shoe cabinet wenge,eon 3 door shoe cabinet wenge,kitchen storage,shoe racks,shoe racks (web exclusive),sofa,tv cabinets units,wardrobes |
| 10 | accessories,beds,carpets,chairs,collections,dining tables sets,lighting,sofas,storage,tables |
| 7 | drawers storage,nigel storage unit mahogany,nigel storage unit oak,nigel storage unit wenge,roger storage unit mahogany,roger storage unit oak,roger storage unit wenge |
| 6 | eon 2 door shoe cabinet wenge,mac 3 door storage unit wenge,royce tv unit black glass,shoe racks,sofa,tv cabinets units,zona kitchen cabinet wenge |
| 6 | eon 2 door shoe cabinet wenge,pearl center table,royce tv unit black glass,shoe racks,sofa,tv cabinets units |
| 4 | brooke tv unit wenge,roxx wall unit oak,roxx wall unit wenge,zona kitchen cab-inet oak |
| 3 | shelton side board mahogany,shelton side board oak,shelton side board wenge |
| 3 | racks,tv cabinets units,wardrobes |
| 2 | bar units stools (web exclusive),upholstery stitching |
| 2 | stool and chairs (web exclusive),swing chairs (web exclusive) |

## 5. CONCLUSION

Frequent Itemset Mining is used for generating groups of items which are frequently occurring together. It can be used in vari-ous different fields like products, text, tags, documents, biology, etc. Logical Itemset Mining is an alternative proposed for Frequent Itemset Mining. It is robust to inconsistencies (noise) and effec-tively addreses mixture-of and projection-of properties which are not very effectively addressed in Frequent Itemset Mining. LISM is simple in implementation. It uses only two passes through the in-put data and stores relatively less data and thus it is highly scalable. The output which is generated can stored using inverted index to be accessed easily.

## 6. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami, Mining associa-tion rules between sets of items in large databases, SIGMOD, pp. 207216, 1993

[2] M. J. Zaki, Scalable algorithms for association mining, IEEE TKDE, vol. 12, pp. 372-390, 2000

[3] R. Agrawal and R. Srikant, Fast algorithms for mining m asso-ciation rules in large databases , VLDB, pp. 487499,1994.

[4] Shailesh Kumar, Chandrashekar V and C V Jawahar, Logical Itemset Mining , 2013.

[5] L. Szathmary, A. Napoli, and P. Valtchev, Towards rare itemset mining, ICTAI (1), pp. 305312, 2007.

[6] https://hasgeek.tv/skumar0127/speaking-in/638-mapreduce-and-the-art-of-thinking-parallel

[7] https://hasgeek.tv/skumar0127/speaking-in/640-co-occurrence-analytics-a-versatile-framework-for-finding-interesting-needles-in-crazy-haystacks