5th SASTech Iran | Mashhad May 12 - 17 | 2011
5th Symposium on Advances in Science & Technology

New Algorithm For Mining Frequent Patterns Using Graph And Clique Algorithms

Ramin Shafei,MSc Student Of Department of Electrical & Computer Engineering, Qazvin Islamic Azad University Qazvin,Iran,
Email:R.Shafei@Gmail.com
Ali Nourollah, Department of Electrical & Computer Engineering, Qazvin Islamic Azad University, Qazvin, Iran
Email :nourollah@aut.ac.ir
Presenter:Ramin Shafei

Abstract

Frequent patterns are patterns such as sets of features or items that appear in data frequently. Finding such frequent patterns has become an important data mining task because it reveals associations, correlations, and many other interesting relationships hidden in a dataset. In this paper we present a new algorithm to discover large itemset pattern. In this approach, the condensed data is used and is obtained by transforming into a clique problem. Firstly, the input dataset is transformed into a graph-based structure and then we find cliques as candidate patterns. In this approach the number of candidate patterns is less than other algorithms, so this new algorithm is fast and accurate and because of using graph and it is easy and simple to update graph so this algorithm is more flexible. The computational results show large itemset patterns with good scalability properties.

Keywords: frequent pattern, data mining, clique

1. Introduction

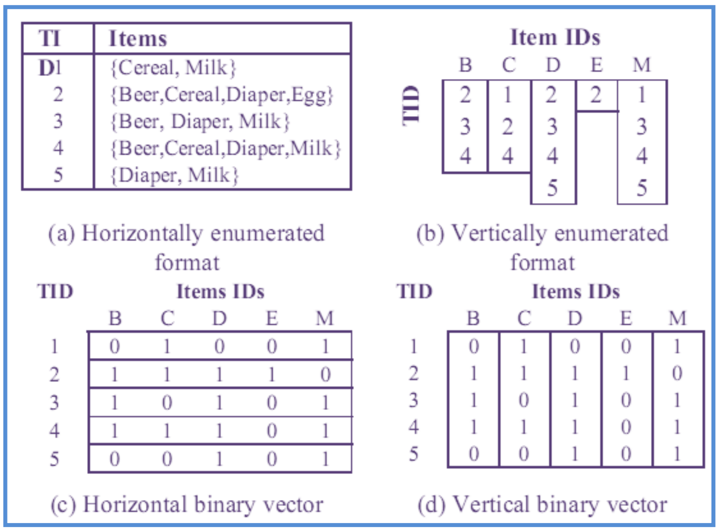
Data Mining, also known as Knowledge Discovery in Databases (KDD),is to find trends, patterns, correlations, anomalies in these databases which can help us to make accurate future decisions. Data Mining only helps experts to understand the data and lead to good decisions. Data Mining is an intersection of the fields Databases, Artificial Intelligence and Machine Learning.

Frequent pattern mining is the discovery of relationships or correlations between items in a dataset. A set of market basket transactions is a common dataset used in frequent pattern analysis. A dataset is typically in a table format. Each row is a transaction, identified by a transaction identifier or a TID. A transaction contains a set of items bought by a customer. A set of transactions might be organized in either an enumerated (dense), or a sparse binary vector format. In either format a dataset can be processed horizontally or vertically. Fig. 1 illustrates the data organization formats of a simple market basket dataset.

In a horizontally enumerated data organization (fig. 1a), each transaction contains only items positively associated with a customer purchase. It is a simplistic representation of market basket data because it ignores other information such as the quantity of purchased items or the profit of item sold. A horizontally enumerated format is sometimes referred to as a TidLists dataset organization. In a vertical organization of items bought enumeration (Fig.

1b), each column stores an ordered list of TIDs of the transactions that contain an item. This format of a dataset occupies that same space as the horizontally enumerated format.

Figs. 1c and 1d represent a binary vector format. A value in each vector cell is 1 if the item is present in a transaction and 0 otherwise. A binary vector format is referred to as a TidSets dataset organization.



TI	Items
D1	{Cereal, Milk}
2	{Beer,Cereal,Diaper,Egg}
3	{Beer, Diaper, Milk}
4	{Beer,Cereal,Diaper,Milk}
5	{Diaper, Milk}

(a) Horizontally enumerated format

Item IDs

B	C	D	E	M
2	1	2	2	1
3	2	3		3
4	4	4		4
		5		5

TID

(b) Vertically enumerated format

Items IDs

TID	B	C	D	E	M
1	0	1	0	0	1
2	1	1	1	1	0
3	1	0	1	0	1
4	1	1	1	0	1
5	0	0	1	0	1

(c) Horizontal binary vector

Items IDs

TID	B	C	D	E	M
1	0	1	0	0	1
2	1	1	1	1	0
3	1	0	1	0	1
4	1	1	1	0	1
5	0	0	1	0	1

(d) Vertical binary vector

2. The Market Basket

The input for the market basket analysis is a dataset of purchases. A market basket is composed of items bought together in a single trip to a store. The most significant attributes are the transaction identification and item identification. While ignoring the quantity bought and the price. Each transaction represents a purchase, which occurred in a specific time and place. This purchase can be linked to an identified customer (usually carrying a card) or to a non-identified customer.

The dataset with multiple transactions can be shown in a relational table (transaction, item). Corresponding to each attribute there is a set called domain. The table (transaction, item) is a set of all transactions T={T1, T2, T3, …, Tn} where each transaction contains a subset of items Tk = {Ia, Ib, Ic …}.

To exemplify this problem, an instance with 5 items and 7 transactions is given in table 1. The domain(item) is equal to {a, b, c, d, e} and the domain(transaction) is equal to {1, 2, 3, 4, 5, 6, 7}.

date	customer	transaction	item
05-Jan	1001	1	b, c, d
05-Jan	1003	2	a, b
05-Jan	1004	3	a, c, e
07-Jan	1001	4	b, c, d, e
07-Jan	1005	5	a, c, d
07-Jan	1003	6	a, d
07-Jan	1006	7	b, c

Table 1- Sample data to illustrate the market basket analysis

Based on the attributes (transaction, item), the market basket will be defined as the N items that are bought together more frequently. Once the market basket with N items is known, we can move on to cross-selling. The next step is to identify all the customers having bought N−m items of the basket and suggest the purchase of some m missing items. In order to make decisions in marketing, the market basket analysis is a powerful tool supporting the implementation of cross-selling strategies. For instance, if a specific customer's buying profile fits into a identified market basket, the next item will be proposed.

The cross-selling strategies lead to the recommender systems. A traditional Recommender System is designed to suggest new products to frequent customers based on previous purchase patterns. One of the first approaches, in order to find the market basket consists in using the Collaborative Filtering software developed by Net Perception that finds a "soul mate" for each customer. A customer’s "soul mate" is a customer who has identical tastes and therefore the same market basket. The software is installed in hundreds of companies. However, its disadvantage is that it merely compares two individuals and this does not allow an overall view. For example: customer X bought four books about Data Mining and a book about Cooking, and customer Y bought the same four books about Data Mining. Therefore, the software will suggest the Cooking book as the next item for customer Y, leading to possible mistakes.

2.1 The Apriori Algorithm

In opposition to the "soul mate" algorithm, the Apriori Algorithm takes all of the transactions in the database into account in order to define the market basket. The market basket can be represented with association rules, with a left and a right side Left $\Rightarrow$ Right. For instance, given an itemset {A, B, C} the rule {B, C}$\Rightarrow${A} should be read as follows: if a customer bought {B,C} he would probably buy {A}. This approach was initially used in pattern recognition and it became popular with the discovery of the following rule: "on Thursdays, grocery store customers often purchase diapers and beer together".

To evaluate the association rules two measures can be used, the support measure and the confidence measure. Let {A,B} be an itemset and the let $A \Rightarrow B$ be the association rule. The support measure is equal to the relative frequency or the probability $P(A \cap B)$. The confidence measure is given by the conditional probability of B given A, $P(B|A)$, which is equal to $P(A \cap B)/P(A)$. The rule “diapers=>beer” with a support measure of 50% and a confidence measure of 85% means that diapers and beer are bought together in 50% of the transactions and in 85% of cases those who buy diapers will also buy beer, i.e. P(beer|diapers)=85%.

The Apriori algorithm was implemented in commercial packages, such as Enterprise Miner from the SAS Institute [SAS 2000]. As input, this algorithm uses a table with purchase transactions. Each transaction contains its identification and the purchased items, as follows (transaction, item). Input parameters are defined as the maximum number of acquired items (max_k) and the minimum support (minsup) of a certain basket. The Enterprise Miner package from SAS Institute implemented this algorithm using max_k=4 and minsup=5% as default values. The minsup is of extreme importance in the algorithm since it will prune the useless branches in the search.

The minsup is a parameter used to control the combinatorial expansion of the exponential algorithm. As Apriori uses only a single value for the minsup, some basket sizes may be oversized while others may be undersized.

The first step of the Apriori algorithm generates sets of market baskets. I_k is defined as the set of frequent items with k items bought together. Firstly, the algorithm filters the items with a frequency that is higher than the minsup, generating I_1. In the following stages, for each I_k it generates the I_{k+1} candidates, such as $I_k \subseteq I_{k+1}$. For each I_{k+1} candidate, the algorithm removes the baskets, which are lower than the minsup. The cycle ends when it reaches I_{max_k}.

In the second step, the Apriori algorithm generates sets of market baskets and then generates association rules Left $\Rightarrow$ Right. For each rule, the support measure and the confidence measure are calculated.

The outputs of the Apriori algorithm are easy to understand and many new patterns can be identified. However, the sheer number of association rules may make the interpretation of the results difficult. A second weakness of the algorithm is the computational times when it searches for large itemsets, due to the exponential complexity of the algorithm.

To overcome the difficulty of finding large itemsets, we have developed a new algorithm and called it SHN[1] algorithm. In this section, firstly we describe some graphs concepts required for the understanding of the algorithm. Secondly, the SHN algorithm is presented in two steps − the transformation step and the search step. Finally, the similarities and differences between this method and other methods are discussed.

3 Graph Concepts

A possible way to condense the data is by transforming it into a graph structure. A graph is a pair G=(V,E) of sets satisfying $E \subseteq [V]^2$ where elements of E are 2-element subsets of V. The elements of V are the vertices (or vertexes, or nodes, or points) of the graph G(V,E) and the elements of E are its edges (or arcs, or lines).

Given the undirected graph G(V,E), then G1(V1,E1) is called a subgraph of G if $V1 \subseteq V$ and $E1 \subseteq E$, where each edge of E1 is incident in the vertices of V1. A subgraph G1 is said to be complete if there is an arc for each pair of vertices. A complete subgraph is also called a clique. A clique is maximal if it is not contained in any other clique. In the maximum clique problem the objective is to find the largest complete subgraph in a graph. The clique number is equal to the cardinality of the largest clique of the graph. If a weight is given to each edge, the subgraph is known as a weighted subgraph, and the weight of the subgraph is given by the sum of the weights of the edges, as follows:

$$\text{Clique\_weight} = \sum \text{weight}(i,j), \ \forall \ \text{edge}(i,j) \in \text{Clique} \qquad (1)$$

A clique can represent a common interest group. Given a graph representing the communication among a group of individuals in an organization, each vertex represents an individual, while edge (i,j) shows that individual i regularly communicates with individual j. Finding a common interest group where each individual communicates with all of the other group members, corresponds to finding a clique. In French "la clique" is defined as a closely knit group of individuals who share common interests. Finding the maximum clique means finding the largest common-interest group possible.

If a graph with weights in the edges is used, the highest weighted clique corresponds to the common-interest group whose elements communicate the most among themselves. This structure allows the representation of sets of elements that are strongly connected. The Maximum Clique Problem is an important problem in combinatorial optimization with many

[1] Shafei-Nourollah

applications, which include: market analysis, project selection and signal transmission. The Maximum Clique Problem is a hard combinatorial problem, classified as NP-Hard. The algorithm which we use to find all cliques is Bron Kerbosch algorithm. It is a recursive backtracking algorithm that searches for all maximal cliques in a given graph *G*. More generally, given three sets *R*, *P*, and *X*, it finds the maximal cliques that include all of the vertices in *R*, some of the vertices in *P*, and none of the vertices in *X*. Within the recursive calls to the algorithm, *P* and *X* are restricted to vertices that form cliques when added to *R*, as these are the only vertices that can be used as part of the output or to prevent some clique from being reported as output.

The recursion is initiated by setting R and X to be the empty set and P to be the vertex set of the graph. Within each recursive call, the algorithm considers the vertices in P in turn; if there are no such vertices, it either reports R as a maximal clique (if X is empty), or backtracks. For each vertex v chosen from P, it makes a recursive call in which v is added to R and in which P and X are restricted to neighbors of v, N(v), which finds and reports all clique extensions of R that contain v. Then, it moves v from P to X and continues with the next vertex in P.

That is, in pseudo code, the algorithm performs the following steps:

```
    P := V(G)
    R := X := empty
BronKerbosch (G):
    If  P&X = =empty
        report R as clique
    else
    for each vertex v in a degeneracy ordering of G:
         BronKerbosch (R ∪ {v}, P ∩ N(v), X ∩ N(v))
         P := P \ {v}
         X := X ∪ {v}
```

4 The Algorithm Description

To find the market basket patterns, i.e. the most frequent itemsets, the Shn algorithm is described in two steps − the data transformation step and the search step. For the first step, the input is the table T(transaction, item) and the output is a weighted graph G(V,E). For the second step, the input is the graph G(V,E) and the size of the market basket k and the output is the market basket with k-items. For the same transformation step, the search step can run more than once, depending on the market basket dimension one is looking for.

Firstly, if the number of items is too large, a filter can be used to discard the infrequent 1-itemsets, equivalent to the minsup cut. The weighted graph G(V,E) is generated, using the information of the 2-itemset frequency. In the graph G(V,E) the vertex set V represents the number of items. The weighted edge $(i,j) \in E$ represents the times that item i and item j were bought together in the transactions. The procedure to load the graph is the following: given a transaction, create a clique with the given itemset and add one to each edge of the clique in the graph. The process is repeated for all the transactions, returning the adjacent matrix of the weighted graph G, presented in table 2.

In the second step, to find the maximum-weighted clique that corresponds to the most frequent market basket, we adapted the Bron-Kerbosch algorithm. The main algorithm can be sketched as follows:

The SHN Algorithm

```
STEP 1 – Data Transformation
input: support measure, table T(transaction, item);
output: weighted graph G(V,E);
  Discard the infrequent 1-itemset using a filter;
  Generate graph G(V,E) using the 2-itemset frequency;

STEP 2 – Finding the Maximum-Weighted Cliques
input: weighted graph G(V,E) and size k;
output: weighted clique S of size k that corresponds to the most frequent k-itemset;
  Find in G(V,E) the clique S with k vertexes with the maximum weight
```

In the first step, the condensed data is created, in such a way that the second step can run as many times as needed, altering the market basket size, thus showing one of the advantages of having condensed data.

In the transformation step, to create the graph, the time complexity is Θ(N) where N is the number of transactions. In the search step, to find the maximum-weighted clique with size k.

To illustrate the algorithm, two numeric examples are presented as follows:

Numeric Example 1 - Problem Transformation:
Using the data present in table 1, the transactions set is T={T1,T2,T3,T4,T5,T6,T7} where transaction T1={b,c,d}, T2={a,b}, T3={a,c,e}, T4={b,c,d,e}, T5={a,c,d}, T6={a,d} and T7={b,c}. The problem transformation returns de adjacent matrix of the weighted graph G(V,E) in table 2.

G(V,E)	b	c	d	e
a	1	2	2	1
b		3	2	1
c			3	2
d				1

Table 2 - Adjacent matrix of the weighted graph G(V,E)

Numeric Example 2 - Searching for the Maximum-Weighted Clique:
Following the previous numeric example, based on the weighted graph, a chart is presented in figure 2, where only the edges which weigh more than 1 are shown, in order to clarify the figure.

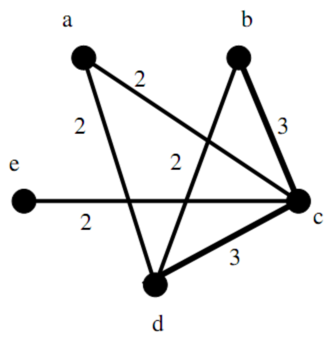


Figure 2 – Weighted Graph

Two cliques with three vertexes stand out from the graph: the clique (b,c,d) and (a,c,d). For the clique (b,c,d) the sum of the edges is equal to 8 and for clique (a,c,d) the value 7 is obtained.

5 Similarities and Differences

The Shn algorithm is based on the conjecture that a clique is similar to a k-itemset. However, some differences can be found. The 3-itemset (a,c,d) in figure 1 occurs twice, while in the clique (a,c,d) the weight of edge (c,d) is equal to 3, so the problem transformation creates a false weighted itemset in (c,d). The drawback of this approach is the fact that in the final graph some specific weighted cliques can appear and they don't correspond to any k-itemset transaction. Nevertheless, the graph structure offers a different type of information.

The soul-mate algorithm compares a k-itemset with an identical k-itemset of a single customer. On the other hand, Apriori-like algorithms find all the exact k-itemsets. In a more general way, the Shn algorithm, mixes all the preferences in the graph G(V,E) going further than a single comparison or an exact itemset count, by creating a pattern.

Data mining can be defined as the science and the art of pattern discovery. A pattern is a shape, a form, a template or more abstractly a model. A pattern is created by the combination of repeated single templates, resulting in a final form with a specific character in nature we can easily identify a crystal, a tree or a fish pattern and fractals are artificial patterns produced by mathematical models. To discover a pattern is to discern a relevant template, with the desired structure in a general set. In our approach, it seems that the information given by the weighted graph is richer than the frequency of a market basket with k-items, since it includes the combined information of different market baskets, as shown by the pattern definition.

6 Computational Results and Analysis

In the validation of an algorithm some choices must be made such as the datasets, the computational environment and the performance measures. The chosen performance measures are the accuracy of the Shn algorithm and the computational times associated to the minsup. The FP-growth algorithm implemented by Borgelt will be used in the computational result comparisons.

dataset	Size(KB)	# items	Average #item/trans	Max # item/trans
Frozen food	169	159	8	59
retail	4,156	18,000	11	77

Table3-Datasets

The two datasets in table 3 were chosen keeping in mind different sizes and different number of items. The same table shows the size in KB, the number of items, the average number of items per transaction and the maximum number of items per transaction for each dataset. Given this data, it is important to study large market baskets within the 10 - 70 item range.

The computer program was written in C#.NET language. The computational results were obtained from a Pentium 4 CPU 2.4 GHz with 512 MB of main memory running in Windows XP.

dataset	SHN algorithm				FP growth	
	filter	vertex	itemset	Time(sec)	minsup	Time(sec)
Frozen food	0.05%	159	5	33	1%	1
	0.05%	159	10	34	1%	1
	0.05%	159	15	35	0.1%	>900
	0.05%	159	20	39	-	-
retail	0.5%	222	5	309	0.1%	1
	0.5%	222	10	331	0.01%	3
	0.5%	222	15	352	0.005%	6
	0.5%	222	20	381	0.001%	>900

table 4-Computational Results

In table 4 the computational results are presented for the datasets frozen-food and retail. Firstly, we run the Shn algorithm, with different itemset sizes: 5, 10, 15 and 20. Then using the FP-growth algorithm, we search for the adequate minsup measure that returns the k-itemset.
In the Apriori-like algorithms the desired k-itemset and the computational time are dependent on the minsup measure. So, some trials must be carried out. In these trials we use the minsup measure in following the decreasing order: 10%, 5%, 1%, 0.5%, 0.1%, 0.05%, 0.01%, 0.005% and 0.001%.

The Shn algorithm finds patterns of large itemsets that don’t correspond to real large itemsets. FP-growth cause very low support measures that Shn can’t reach for the retail dataset with 18,000 items. On the other hand, Shn can obtain large k-itemsets in every datasets.

7 Conclusions

In this approach, the condensed data is obtained by transforming the market basket problem into a maximum-weighted clique problem. There are three important variables that influence the computational time: the number of transactions, the basket size and the number of items. The time complexity of the original Apriori algorithm is dependent on the number of items, the number of transactions and the market basket size. Using a condensed tree data structure, the FP-growth is dependent on the number of items and the market basket size, and it scales better than Apriori. Finally, the time complexity of the Shn algorithm is only dependent on the number of items (that correspond to the number of vertexes of the graph) showing a good scalability. The number of transactions is not relevant for the FP-growth and Shn algorithm due to the transformation into a tree or into a graph problem, allowing the processing of a huge number of transactions. In the Shn algorithm the computational results show the discovery of large itemset patterns with good scalability.

References

R. Agrawal, R. and R. Srikan,(1994) ”Fast algorithms for mining association rules”, Proceedings of the 20th international conference on very large data bases, 478-499.

R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. Verkamo,(2000) Fast Discovery of Association Rules, in Advances in Knowledge and Data Mining, U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy Eds, MIT Press,.

C. Anderson,(2006) The Long Tail: why the future of business is selling less of more, Hyperion Books.

C. Berge, (1991) Graphs, 3rd edition, North-Holland, 1991.

M. Berry and G. Linoff, (1997)Data Mining Techniques for Marketing, Sales and Customer Support, John Wiley and Sons.

C. Borgelt,(2005) An Implementation of the FP-growth Algorithm, Workshop Open Source Data Mining Software, OSDM'05, Chicago, IL, 1-5.ACM Press, USA.

S. Brin, R. Motwani, J.D. Ullman and S.Tsur, Dynamic Itemset Counting and Implication Rules for Market Basket Data, in Proceedings of the 1997 ACM SIGMOD Conference, Tucson, Arizona, 1997, pp.255-264.