Graph-Based Structures for the Market Baskets Analysis

Luís Cavique [*]

[*] ESCS-IPL / IST-UTL
Portugal
`lcavique@escs.ipl.pt`

Abstract

The market basket is defined as an itemset bought together by a customer on a single visit to a store. The market basket analysis is a powerful tool for the implementation of cross-selling strategies. Although some algorithms can find the market basket, they can be inefficient in computational time. The aim of this paper is to present a faster algorithm for the market basket analysis using data-condensed structures. In this innovative approach, the condensed data is obtained by transforming the market basket problem in a maximum-weighted clique problem. Firstly, the input data set is transformed into a graph-based structure and then the maximum-weighted clique problem is solved using a meta-heuristic approach in order to find the most frequent itemsets. The computational results show accurate solutions with reduced computational times.

Keywords: data mining, market basket, similarity measures, maximum clique problem.

1 Introduction

This paper addresses the problem of finding market baskets in large databases. Current database capacities associated with bar code technology and growth of the Internet has led to a huge collection of customer transaction data.

Companies in different sectors such as banking, insurance, telecommunications and airlines have now become more customer oriented than never. To obtain the customer's profile there are two main data sources using the customer's personal data and the product-oriented data. In order to gather demographic, social, geographic, personality or lifestyle data of the customer, costly surveys are needed. On the other hand, product-oriented data, about the frequency and the quantity each customer buys of a certain item, already exists in the companies' database. In order to establish customer relationship strategy one needs to find out whom the best customers are, how they respond to a campaign and be able to predict the next item each customer will buy, thus implementing a cross-selling strategy.

In section 2 we define the market basket problem and review the Apriori algorithm that solves this problem. Since this algorithm has a non-polynomial time complexity, we describe related work that tries to overcome this handicap.

In section 3 we present a swifter algorithm - Similis, which first of all transforms the input data set into a graph-based structure, and then the new problem, the weighted clique problem, is solved using a meta-heuristic approach. Each maximum-weighted clique corresponds to a quasi-most-frequent itemset.

In section 4 the Similis algorithm is validated with two real data sets and the computational results are presented.

Finally, in section 5 the conclusions point out the differences between the Apriori algorithm and the Similis algorithm.

In this paper we will refer to the market basket (or itemset) whenever it is related to physical data, on the other hand, if it refers to condensed data the terms graph-based structure (or clique) are used.

2 The Market Basket

2.1 The definition of the problem

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

Definition 1: The file with multiple transactions can be shown in a relational database table T(customer, item). Corresponding to each attribute there is a non-empty set called domain. The domain(customer) = {1, 2 , 3,... n} and the domain(item) = {a, b, c, ..., z}. The table T(customer, item) can be seen as a set of all customer transactions Trans $=\{t_1, t_2, t_3, \ldots, t_k\}$ where each transaction contains a subset of items $t_k = \{i_a, i_b, i_c \ldots\}$. The relational table T(customer, item) can also be seen as a set of item-clientele Itc$=\{i_1, i_2, i_3, \ldots\}$ where each item-clientele contains a subset of customers $i_k=\{c_1, c_2, c_3, \ldots\}$.

Based on the attributes (customer, 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 applications, 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 known market basket, the next item will be proposed.

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 [Hughes 2000]. 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.

2.2 The Apriori Algorithm

In opposition to the previous algorithm, the Apriori Algorithm [Agrawal et al. 1996] 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 L⇒R. For instance, given a itemset {A,B,C} the rule {B,C}⇒{A} should be read as follows: if a customer bought {B,C} he probably would 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" [Berry and Linoff 1997].

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⇒B be the association rule. The support measure is equal to the relative frequency or P({A,B}). The confidence measure is given by the conditional probability of B given A, P(B|A), which is equal to P({A,B})/P(A).

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 the customer identification and the purchased items, as follows (customer, item). Input parameters are defined as the maximum number of acquired items (max_k) and the minimum support (min_sup) of a certain basket. The Enterprise Miner package from SAS Institute implemented this algorithm using max_k=4 and min_sup=5% as default values. The min_sup is of extreme importance in the algorithm since it will prune the useless branches in the search.

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 min_sup, 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 min_sup. 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 L⇒R. For each rule, the support measure and the confidence measure are calculated. In order to implement the cross-selling strategy the data analysts choose, firstly, the dimension of the basket, secondly, they choose the rules with the highest support measure. Finally, those with the highest confidence measure are chosen, among those with the highest support measure.

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 is the computational times, due to the exponential complexity of the algorithm.

2.3 Related Work

Apriori algorithm has an exponential time complexity, and several passes over the input table are needed. To overcome these handicaps some proposals have been made.

The Apriori algorithm performs as many passes over the data as the size of the itemsets. The Dynamic Itemset Counting, the DIC Algorithm, reduces the number of passes made over the data, using itemsets forming a large lattice structure [Brin et al. 1997]. DIC starts counting the 1-itemset and then adds counters 2, 3, 4, ..., k itemsets. Thus, after a few passes over the data it finishes counting all the itemsets. Running DIC and Apriori, DIC outperformed Apriori in the majority of cases.

In [Aggarwal, Wolf and Yu 1999] a method for indexing market basket data for similarity search is discussed. The index structure requires the partition of the set of all k-itemsets into subsets. They create a graph so that each node corresponds to an item, and for each pair of items a weight is added, which is the inverse of the support measure. Finally, the set of items is divided into k-sets. This algorithm shows good scalability with an increase in the number of transactions.

Just like the DIC algorithm, the MARC algorithm [Liu, Lu and Lu 2001] avoids several passes over the databases. MARC algorithm will use the summarized cluster information. The algorithm analyzes the similarities between transactions and creates clusters of similar transactions.

In the GCTD algorithm [Chen et al. 2002], the concepts of similarity relationships and the clustering problem appear together in order to discover connected components in an undirected graph.

The condensed data representation is extremely useful [Jeudy and Boulicaut 2002], taking into account that the Apriori algorithm has a better performance using sparse data rather than using highly correlated data. The latter is considered difficult or even intractable.

In the developing of recommender systems, i.e., systems that personalize recommendations of items based on the customer's preferences, in [Lin, Alvarez and Ruiz 2002] the authors present an algorithm that does not require the min_sup measure. Having previously defined min_sup, a negative result can be expected, by cutting down either too many or too few itemsets.

In order to obtain frequent market baskets in reduced computational times, in the next section we present the Similis Algorithm [Cavique e Themido 2001] [Cavique 2002]. This algorithm reuses some of the mentioned strategies, such as the reduction of passes over the database, the reduction of the number of parameters (e.g. min_sup) and the aggregate measures (e.g. similarity measures).

3 The Similis Algorithm

To overcome the performance problems of the Apriori algorithm we have developed a new algorithm, Similis, meaning similar. This term was chosen just like Apriori, since they are both Latin names.

The Similis algorithm is developed in two steps - the transformation step and the search step. In this section, we firstly describe the problem transformation and the similarity measures, just like in [Chen et al. 2002]. Secondly, we justify the graph-based structure and present the search step and finally we present the algorithm in its totality.

3.1 Problem Transformation

The input for the Market Basket Analysis is a table with two attributes T(customer, item) that represents the item bought by a customer in a single transaction.

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). In market basket case each vertex corresponds to an item and each arc has a weight which represents the distance between the adjacent vertices. The distance between two items is given by the frequency that the two items are bought together.

To find the values for the weighted graph G(V,E) some similarity measures can be used. The similarity value of the two items will be high if they are both included in frequent transactions.

Some authors study association measures between transactions [Liu, Lu and Lu 2001]. However, in this work the associations between items are analyzed. If two items are frequently bought in the same transactions, then they belong to a frequent market basket. In order to create sets of items, one association measure must be found, similarity or distance (dissimilarity) measures can be created.

Given two item-clientele A and B, the first two similarity functions that come up are the number of matches and the hamming distance. The number of matches is given by the cardinality of $(A \cap B)$, while on the other hand, the hamming distance is given by the sum of the cardinalities of the sets (A-B) and (B-A). A third measure is the euclidian distance, which is given by the square root of the sum of the squares of the differences.

Even though a variety of measures can be used, normalized measures are preferable. The most common ones are those based on vector comparisons.

In the generation of the weighted graph we will use the analogy with the Information Retrieval Systems. The Information Retrieval systems were developed in the middle of the 20^{th} century to manage the vast amount of scientific information produced in universities, research centers and by the academic press. The commercial Information Retrieval systems have millions of recorded documents each containing many terms. Information Retrieval systems are very efficient in many fields that deal with documents and semi-structured data.

At this stage we will make an analogy between the Information Retrieval techniques and the market basket problem. Just as a document consists of many terms, a market basket contains several items. In the same way, the study of terms in a document is identical to the study of items in a basket. For each pair of items (A,B) a similarity measure SIM(A,B) can be found, if the items are bought together many times they have a strong similarity, but they have a weak similarity if they are not usually bought together. For all items, an item similarity matrix is generated, which can be represented by the adjacent matrix of the weighted graph G(V,E).

To present the Information Retrieval System, in [Salton and McGill 1983] the authors describe the following similarity measures. All measures use binary matrices and return normalized values between 0 and 1. The Dice (sim1), Jaccard (sim2) and Cosine (sim3) coefficients are widely used given their simplicity.

$$sim1(\mathrm{A,B}) = \frac{2.\sum_{\mathrm{k}}(\mathrm{A_k.B_k})}{\sum_{\mathrm{k}}\mathrm{A_k}+\sum_{\mathrm{k}}\mathrm{B_k}}$$

$$sim2(\mathrm{A,B}) = \frac{\sum_{\mathrm{k}}(\mathrm{A_k.B_k})}{\sum_{\mathrm{k}}\mathrm{A_k}+\sum_{\mathrm{k}}\mathrm{B_k}-\sum_{\mathrm{k}}(\mathrm{A_k.B_k})}$$

$$sim3(\mathrm{A,B}) = \frac{\sum_{\mathrm{k}}(\mathrm{A_k.B_k})}{\sqrt{\sum_{\mathrm{k}}(\mathrm{A_k})^2.\sum_{\mathrm{k}}(\mathrm{B_k})^2}}$$

Definition 2: A multiplicative model will de used to express the weight of an edge (A,B). The weight of the edge (A,B) takes into account the similarity and frequency of items, such as:

$$weight(A,B) = sim(A,B)\ .\ frequency(A,B)$$

The similarity value of two items will be high if they are both included in the same transactions. The frequency of the item must be considered to guarantee a correspondence between high-weighted edges and items that appear in many transactions.

There are several ways to define item frequency. In this work we opt for the average of the relative frequency of the two items, given by:

$$frequency(A,B) = \frac{\sum_k A_k + \sum_k B_k}{2.n}$$

Using the Dice similarity measure, it is interesting to notice that the weight is equal to the support measure of two items.

$$weight1(A,B) = \frac{\sum_k (A_k.B_k)}{n}$$

However, based on the computational results the Jaccard similarity measure was chosen.

Numeric Example 1:
To exemplify the first step in the Similis algorithm, a problem with 5 items and 7 customers is given, the domain(Item)= {a, b, c, d, e} and the domain(Customer)= {1, 2, 3, 4, 5, 6, 7}.

The item-clientele Itc={Ia, Ib, Ic, Id, Ie}where Ia={2, 3, 5, 6}, Ib={1, 2, 4, 7}, Ic={1, 3, 4, 5, 7}, Id={1, 4, 5, 6} and Ie={3, 4}.

To obtain the adjacent matrix of graph G(V,E), in the calculation of the weight of the edges the Jaccard similarity measure (sim2) is use. For instance, for items *a* and *b*, the weight (a,b) is given by:

$$\text{sim2 (Ia,Ib)} = \frac{\sum_{k} (\text{Ia}_k.\text{Ib}_k)}{\sum_{k} \text{Ia}_k + \sum_{k} \text{Ib}_k - \sum_{k} (\text{Ia}_k.\text{Ib}_k)} = \frac{1}{4+4-1} = \frac{1}{7}$$

$$\text{frequency (Ia,Ib)} = \frac{\sum_{k} \text{Ia}_k + \sum_{k} \text{Ib}_k}{2 \cdot \text{n}} = \frac{(4+4)}{2 \cdot 7} = \frac{4}{7}$$

weight2(a,b)= sim2 (Ia,Ib) . frequency (Ia,Ib)= 1/7 x 4/7 = 0.082

For items *a* and c, the weight (a,c) is given by:

weight2(a,c)= sim2(Ia,Ic) . frequency (Ia,Ic)= 2/7 x 4.5/7 = 0.184

The process is repeated for all the pairs of items, returning the adjacent matrix of the weighted graph G, presented in table 1.

Table 1: Adjacent matrix of the weighted graph G(V,E)

G(V,E)	**b**	**c**	**d**	**e**
a	0.082	0.184	0.190	0.086
b		0.321	0.190	0.086
c			0.321	0.200
d				0.086

In this way the data is condensed in a graph G(V,E) using a procedure with time complexity O($|V|^2$), where |V| is the number of vertexes i.e. the number of items.

3.2 Searching for the Maximum-weighted Clique

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 weighted subgraph, and the weight of the subgraph is given by the sum of the weights of the edges.

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 most weighted clique corresponds to the common-interest group whose elements communicate the most among themselves. This structure allows the representation of sets of elements strongly connected.

We use the same principle in the market basket analysis, where the items bought together in a single transaction have a higher degree of connection than items bought in different transactions. In the last section we presented the data transformation of table T(customer, item) into a weighted graph G(V,E). In other words, the market basket problem can be transformed into the maximum-weighted clique problem, using a procedure with quadratic complexity.

Definition 3: As a way of comparing the weighted clique, given a size k, the clique weight must be calculated. Although it can be defined in several ways, the sum of the weighted edges was chosen, as follows:

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

The Maximum Clique Problem is an important problem in combinatorial optimization with many applications, which include: market analysis, project selection and signal transmission [Berge 1991]. The Maximum Clique Problem is a hard combinatorial problem, classified as NP-Hard [Garey and Johnson 1979]. The interest in this problem led to the algorithm thread challenge on experimental analysis and algorithm performance promoted by Second DIMACS Implementation Challenge [Johnson and Trick 1996]. A recent state of the art survey is found in [Bomze et al. 1999].

In order to deal with large data volumes, a number of heuristic algorithms have been recently proposed: Genetic Algorithms [Balas and Niehaus 1998], Neural Networks [Jagota, Sanchis and Ganesan 1996], GRASP with Simulated Annealing [Feo, Resende and Smith 1994], Tabu Search [Soriano and Gendreau 1996] [Cavique, Rego e Themido 2002b] and Scatter Search [Cavique, Rego and Themido 2002a] [Cavique 2002].

To find the Maximum-weighted Clique with size k, the Primal-Tabu algorithm [Cavique, Rego and Themido 2002b] [Cavique 2002] was adapted by adding the restriction related to the clique size and the objective function was changed.

Three possible moves were identified: the related neighborhood structures are N^+, N^-, and $N^0$for addition, removal and swap of a vertex of the subgraph. To avoid cycling in the iterative search process, a short-term memory is incorporated. The short-term memory is usually implemented using a Tabu list, made up of a set of (reverse) moves, which are forbidden for a certain number of iterations.

The original version of Primal-Tabu combines the neighborhood structures N^+, N^0 and N^-. At each step one new solution S' is chosen from the neighborhood N(S) of the current solution S. If it is possible to increase the clique value, one vertex is added to the clique using N^+, or else try to swap vertices using the neighborhood structure N^0, otherwise the heuristic moves backwards removing one vertex using N^-. At each iteration the current solution S and the best solution found S* are updated whenever the clique value is increased.

The weighted version of the algorithm restricts the growth of $N^+(S)$ by using the condition $|S| <k$; the objective function was changed, instead of using the vertex number, the function for the clique weight is now used. The local search will run through the solution space searching for the maximum-weighted clique with a given dimension k. The Primal-Tabu Heuristic can be sketched as follows:

S is the current solution and S' the trial solution
S* is the best solution found
Tabu is the tabu list
$N^+(S)$, $N^-(S)$, $N^0(S)$ are the neighborhood structures

Primal-Tabu Meta-Heuristic for the Maximum-weighted Clique

```
input: weighted graph G(V,E), size k;
output: maximum-weighted clique S* with size k;
initialize S, S* e Tabu;
    while not end condition
        if $N^+(S)\setminus Tabu \neq \emptyset$ and $|S| < k$ choose the best S';
        else if $N^0(S)\setminus Tabu \neq \emptyset$ choose the best S';
            else choose the best S' in $N^-(S)$ and update Tabu;
        update S←S';
        if Clique_weight(S) > Clique_weight(S*) then S*←S;
    end while;
return S*;
```

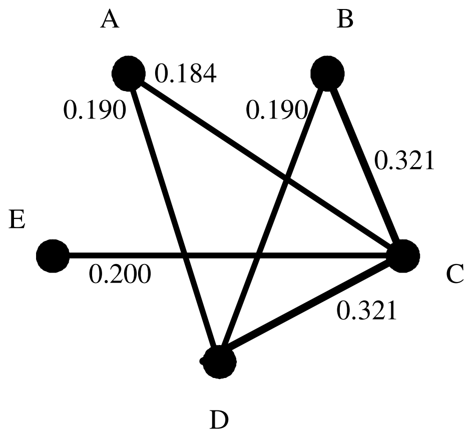

Figure 1: Weighted Graph

Numeric Example 2:
Following the previous numeric example, based on the found weighted graph, a chart presentation is given in figure 1, where only the edges which weigh more than 0.100 are shown, in order to clarify the figure. 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 0.832

and for clique (A,C,D) the value is 0.695 is obtained. The most frequent market basket with 3 items occurs twice, is (B,C,D) and is also the most weighted clique, showing the aim was reached. The information given by the weighted graph seems to be richer than the frequency of a market basket with k items, since it also includes information from smaller markets baskets. The formal validation is given in section 4, where computational results show accurate solutions.

3.3 The Algorithm Description

To find the market basket patterns, i.e. the most frequent itemsets, the Similis algorithm is described in two steps - the data transformation step and the search step.

For the first step, the input is the table T(customer, item) and the output 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. The search step can run more than once, depending on the market basket dimensions one is looking for.

Firstly a weighted graph G(V,E) is generated based on the similarities of the items. In the graph G(V,E) the vertex set V represents the itemset in the market basket. The weighted edge (i,j)∈E represents the similarity between item *i* and item *j*. Two items are similar if they were bought together in many transaction.

Finally, to find the maximum-weighted clique that corresponds to the most frequent market basket, we adapted the Primal-Tabu Meta-heuristic. The main algorithm can be sketched as follows:

```
The Similis Algorithm
STEP 1 – Data Transformation
input: table T(customer, item)
□ Generate graph G(V,E) using the similarities between items
output: weighted graph G(V,E)
STEP 2 – Finding the maximum-weighted cliques
input: weighted graph G(V,E) and size k
□ Find in G(V,E) the clique S with k vertexes with the maximum weight, using the Primal-Tabu Meta-heuristic.
output: weighted clique S of size k that correspond to the most frequent market basket with k items.
```

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.

The Apriori algorithm firstly finds the most frequent itemsets and secondly generates the association rules. We follow the same procedure by finding solutions and then creating association rules. To implement a cross-selling strategy, frequent itemsets with dimensions 2, 3, . . . , k must be found, then association rules are generated and evaluated with the support and confidence measures.

4 Computational Results

In the validation of an algorithm some choices must be made such as the data files, the compiler, the computer and the performance measures. To validate the Similis algorithm, two data sets were chosen. The first includes the result of a survey carried out at inns named "Pousadas de Portugal" involving 43 inns (items) and 2500 customers [Cardoso, Themido e Pires 1999]. The second regards the distribution of frozen food items throughout Portugal by Nestle enterprise involving 158 items and 450 consumer centers. The computer program was written in C language and the Microsoft Visual C++ compiler was used. The computational results were obtained from a Pentium 200 MHz. The performance measures that we are going to use are the quality of the solutions and the computational times. The Apriori algorithm that has been implemented in the SAS Enterprise Miner will be used in the computational result comparisons.

The Apriori algorithm has two possible measures: the support and the confidence measure. The Similis algorithm retrieves only the clique weights as the performance measure, so the use of the market basket support seems pertinent for measuring the quality of both algorithms.

For each algorithm the top five solutions were reached. For the Apriori algorithm the solutions are ordered by decreasing support and in the Similis algorithm they are ordered by decreasing clique weight.

Definition 4: For each market basket with k items, found by the Similis algorithm, an accuracy measure is required. The Accuracy function is given by the average support of the top five solutions found by the Similis algorithm, divided by the average support of the top five solutions found by the Apriori algorithm, that is:

$$Accuracy = \frac{AverageSupport(Similis)}{AverageSupport(Apriori)}.100$$

The "Pousadas" data set includes the original file and two simulated files SX and SY with the same dimension. The computational time (in seconds) for the Apriori algorithm (TA), the Similis algorithm (TS) and the Accuracy obtained (Acc), are reported in table 2. The average accuracy is close to 97%. The Similis algorithm computational times are almost constant, while the computational times for the Apriori algorithm are exponential. What makes the real difference between the two algorithms is the computational time.

Table 2: Computational results of Pousadas data set

Basket Size	Acc Real	Acc SX	Acc SY	TA (sec.)	TS (sec.)
2	100	100	100	3	3
3	100	94	95	42	2
4	94	96	94	440	3

Acc- Accuracy; TA- Time Apriori; TS- Time Similis

The second data set was handed in by the Nestle enterprise and we generated some reduced files from the original one. The data set includes the Nestle20 with 20 items, the Nestle40 with 40 items and Nestle80 and Nestle120 with respectively 80 and 120 items.

The accuracy obtained by the Similis algorithm (Acc), the computational time in seconds for the Apriori algorithm (TA) and Similis algorithm (TS) are reported in table 3. The computational results are limited by the non-existence of cliques, in the Nestle20 case, but the great limitations are the Apriori computational times for Nestle40, Nestle80, Nestle120 and NestleReal.

Table 3: Computational results for Nestle data set for different basket sizes

basket	Nestle20			Nestle40			Nestle80			Nestle120			NestleReal		
size	Acc	TA	TS	Acc	TA	TS	Acc	TA	TS	Acc	TA	TS	Acc	TA	TS
2	100	0	18	100	1	56	100	5	204	100	12	384	100	23	573
3	97	1	17	92	8	43	100	89	159	100	347	293	100	1476	442
4	98	3	19	95	72	43	100	3771	159	99	25512	291	98	35235	455
5	86	10	17	88	486	44	93	21332	159						
6	93	26	16	92	2761	36									
7	95	51	16	97	13842	35									
8	90	80	16												
9	93	139	16												
10															

Acc- Accuracy; TA- Time Apriori; TS- Time Similis

The Similis algorithm has a good performance in the Acc with an average of 96%, and has constant computational times for baskets with different sizes for the Nestle data set.

In the top five solutions, the best solution found with the Apriori algorithm is also the one that is generally found by the Similis algorithm. The difficult problem is finding the most frequent baskets, since the frequency calculation corresponds to polynomial complexity procedures. After getting the maximum-weighted cliques (or baskets) using the Similis algorithm, it is possible to re-order the baskets based on the frequency in polynomial time.

When we need quasi-most-frequent market baskets involving limited computational effort, the Similis algorithm is very competitive as shown in the computational results of both data sets.

5 Conclusions

In this paper after having described the usefulness of the market basket problem, the Apriori algorithm was presented. The main disadvantage of the Apriori algorithm is the exponential time complexity, since it performs many passes over the data. Using few items or sparse data the algorithm is efficient, while when using correlated data the performance degrades significantly. In order to reduce the number of items in the input data, the min_sup parameter is used. However, since the min_sup is independent from the data table, it may lead to unpredictable data reductions. Finally, the Apriori algorithm generates a huge number of associative rules, of which only a few ones are used in cross-selling strategies.

Considering all of the handicaps of the Apriori algorithm, we developed the Similis algorithm because of its lower computational complexity, thus allowing the resolution of a greater number of real problems. In this innovative approach, the condensed data is obtained by transforming the market basket problem in a maximum-weighted clique problem. Firstly, the input data set is transformed into a graph-based structure and then the maximum-weighted clique problem is solved using a meta-heuristic approach in order to find the most frequent itemsets. The computational results show accurate solutions with reduced computational times.

Both algorithms have very different characteristics. Whereas the Apriori algorithm uses a transaction file, the Similis algorithm organizes its condensed data in a weighted graph. They also differ in the procedure to obtain the solutions. The Apriori algorithm uses an exact counting method with exponential complexity, whereas the Similis algorithm uses a meta-heuristic approach. The output of the Apriori algorithm reports the exact market basket frequencies, whereas the Similis algorithm works with a transformed problem, using the "imperfect" information of the clique weights and reports quasi-most-frequent market baskets. Each algorithm can treat different data volumes. The Apriori algorithm can only work with limited data volumes, whereas the Similis algorithm can treat large data volumes.

The time complexity of the Apriori algorithm is dependent on the number of items, the number of transactions and the market basket size, whereas the time complexity of the Similis algorithm is only dependent on the number of items (that correspond to the number of vertexes of the graph) showing an excellent scalability with database size.

The Apriori algorithm performs efficiently with few items or with sparse data, as is already the case in banking and insurance implementations. Let us not forget that the default value for the maximum number of items in the Enterprise Miner of SAS is max_k= 4. The Similis algorithm expands the use of the market basket to hundreds of items in highly correlated data.

The Similis algorithm has some advantages when compared to data-condensed algorithms such as MARC and GCTP, since it doesn't require any additional parameters (like min_sup), making it easier to use.

To validate the Similis algorithm two real-case studies were presented. The first is from the "Pousadas de Portugal" with 43 items and the second is from Nestle enterprise with 158 items. The Similis algorithm performs well, the average of both data sets being equal to 96% for the accuracy function obtained with reduced computational times.

In this innovative approach, the ability to condense the data without using too many parameters and the capacity to find several market baskets with different sizes using reduced computational times, makes Similis algorithm very competitive. The market basket analysis can therefore be implemented in retail markets such, as supermarkets and on-line shopping. Its application can be extended to recommender systems due to its condensed data.

6 References

[1] C.C. Aggarwal, J.L. Wolf and P.S. Yu, A New Method for Similarity Indexing of Market Basket Data, *in Proceedings of the 1999 ACM SIGMOD Conference*, Philadelphia PA, 1999, pp.407-418.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. Verkamo, 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, 1996.

[3] E. Balas, W. Niehaus, Optimized Crossover-Based Genetic Algorithms will be the Maximum Cardinality and Maximum Weight Clique Problems, Journal of Heuristics, Kluwer Academic Publishers, 4, 1998, pp. 107-122.

[4] C. Berge, Graphs, 3^{rd} edition, North-Holland, 1991.

[5] M. Berry and G. Linoff, Data Mining Techniques for Marketing, Sales and Customer Support, John Wiley and Sons, 1997.

[6] I.M. Bomze, M. Budinich, P.M. Pardalos and M. Pelillo, Maximum Clique Problem, in *Handbook of Combinatorial Optimization, D.-Z. Du and P.M. Pardalos Eds*, 1999, pp.1-74.

[7] 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.

[8] M. Cardoso, I. Themido and F. Moura Pires, Evaluating a Clustering Solution: an Application in the Tourism Market, Intelligent Data Analysis, 3, 1999, pp. 491-510.

[9] L. Cavique and I Themido, A New Algorithm for the Market Basket Analysis, Internal Report CESUR-IST, UTL, Portugal, 2001.

[10] L. Cavique, C. Rego and I. Themido (a), A Scatter Search Algorithm for the Maximum Clique Problem, *in Essays and Surveys in Meta-heuristics, C. Ribeiro and P. Hansen Eds,* Kluwer Academic Publishers, 2002, pp. 227-244.

[11] L. Cavique, C. Rego and I. Themido (b), Estruturas de Vizinhança e Procura Local para o Problema da Clique Máxima, Revista de Investigação Operacional, 2002, vol.22, pp. 1-18.

[12] L. Cavique, Meta-heurísticas na Resolução do Problema da Clique Máxima e Aplicação na Determinação do Cabaz de Compras, dissertação de Doutoramento em Engenharia de Sistemas no Instituto Superior Técnico da Universidade Técnica de Lisboa, 2002.

[13] N. Chen, A. Chen, L. Zou and L. Lu, A Graph-based Clustering Algorithm in Large Transaction Databases, Intelligent Data Analysis, 5, 2002, pp.327-338.

[14] T.A. Feo, M.G.C. Resende and S.H. Smith, A Greedy Randomized Adaptive Search Procedure for Maximum Independent Set, Operations Research, 42, 1994, pp. 860-878.

[15] M.R. Garey and D.S. Johnson, Computers and Intractability: a Guide to the Theory of NP-Completeness, W.H. Freeman and Company, New York, 1979.

[16] A.M. Hughes, Strategic Database Marketing, McGraw-Hill, 2000.

[17] A. Jagota, L. Sanchis, R. Ganesan, Approximately Solving Maximum Clique using Neural Network Related Heuristics, *in Clique, Coloring and Satisfiability, Second Implementation Challenge DIMACS, D.S. Johnson and M.A. Trick Eds, AMS,* 1996, pp. 169-203.

[18] B. Jeudy and J.-F. Boulicaut, Optimization of Association Rule Mining Queries, Intelligent Data Analysis, 6, 2002, pp.341-357.

[19] D.S. Johnson and M.A. Trick Eds, Clique, Coloring and Satisfiability, Second Implementation Challenge DIMACS, AMS, 1996.

[20] W. Lin, S.A. Alvarez and C. Ruiz, Efficient Adaptive-Support Association Rule Mining for Recommender Systems, Data Mining and Knowledge Discovery, 6, Kluwer Academic Publishers, 2002, pp.83-105.

[21] F. Liu, Z. Lu and S. Lu, Mining Association Rules Using Clustering, Intelligent Data Analysis, 5, 2001, pp.309-326.

[22] G. Salton and M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, 1983.

[23] SAS Software, Enterprise Miner Documentation, SAS Institute, 2000.

[24] P. Soriano and M. Gendreau, Tabu Search Algorithms for the Maximum Clique, *in Clique, Coloring and Satisfiability, Second Implementation Challenge DIMACS, D.S. Johnson and M.A. Trick Eds, AMS,* 1996, pp. 221-242.