ECCLAT: a New Approach of Clusters Discovery in Categorical Data

Nicolas Durand
IP and Network Laboratory, France Telecom R&D Caen
France

Bruno Crémilleux
GREYC CNRS-UMR 6072, University of Caen
France

Abstract

In this paper we present a new approach for the discovery of meaningful clusters from large categorical data (which is an usual situation, e.g., web data analysis). Our method called Ecclat (for Extraction of Clusters from Concepts LATtice) extracts a subset of concepts from the frequent closed itemsets lattice, using an evaluation measure. Ecclat is generic because it allows to build approximate clustering and discover meaningful clusters with slight overlapping. The approach is illustrated on a classical data set and on web data analysis.

1 Introduction

Discovering the structure and relationships within data is an important problem in many application areas. For instance, in analyzing market basket data, it is interesting to find clusters (i.e. groups) of customers having similar characteristics (or close to each other) while customers in different groups are dissimilar, or to find groups of similar products. On the other hand, for some years, there has been a considerable interest in web data analysis in order to create new applications and services (e.g. clusters of user accesses, clustering of web sessions). In such contexts, it is important to have knowledge discovery methods on categorical data that are both relevant and efficient to tackle the enormous amount of data. In this paper, we focus on a method to discover meaningful clusters from large categorical data sets. We will see below that this approach is quite different from usual clustering techniques. In the following discussion, each data record is called an *example* (or a *transaction* in the data mining context) and is described by *attributes*.

Let us recall that the general meaning of clustering is decomposing or partitioning examples into groups so that the examples in one group are similar to each other and are as different as possible from the examples in other groups.

The conceptual classification methods [4, 9, 19] produce clusters of examples described by categorical attributes. They do not produce directly a clustering. Indeed, these methods create a hierarchy of concepts, generally represented by a lattice [11]. Every concept can be seen as a cluster with its properties (i.e. items) and examples. Even if the number of concepts is smaller than the number of all combinations of items, this number remains very high in real-world applications. We will see in Section 3 the KDD's results (the frequency [2] and the efficiency to extract condensed representations like frequent closed itemsets [3, 20, 22]). Unfortunately, the number of concepts remains high and the hierarchy can not be used nor presented to an expert. Nevertheless, in Section 3, we will show that such clusters can provide a starting point to get a clustering. Furthermore, in many practical applications like web mining, this discovery of meaningful clusters (and not necessarily a clustering) may be of great help. For instance, in a clustering of web pages, it is understandable and useful to obtain a page in two (or more) clusters in order to retrieve it from several kind of queries. In fact, it corresponds to several points of view to classify pages. These clusters can also be used to browse and follow new paths through a hyper-document [5]. The situation is analogous to the discovery of visitor profiles or pre-fetching: several distinct groups (or clusters) may have overlaps between their interests. In the medical area, such meaningful clusters can be used to search for prognostic factors [7].

In this paper, we propose an efficient method to produce a set of clusters from a large set of categorical data with a minimum overlapping ("approximate clustering") or a slight overlapping to catch all the similarities between examples. The behavior of the method is parametrized by the user. Clusters are a subset of concepts from the frequent closed itemsets lattice and are selected according to an evaluation measure.

In the next section, we present related work. In Section 3, we introduce the minimal properties required for discovering interesting clusters from large databases. We will see that the frequent closed itemsets are relevant candidates to be clustered. In Section 4, starting from the frequent closed itemsets (seen as clusters), we propose a method called Ecclat (for Extraction of Clusters from Concepts LATtice) to select the most interesting concepts gathering similar examples. Ecclat is able to build a set of clusters with a minimum overlapping or a slight overlapping. In the latter case, the clusters which will be used, will depend on the point of view adopted. In Section 5, we give practical uses of Ecclat to produce clusters and clustering from categorical data and logs of a proxy server. Our conclusions are presented in Section 6.

2 Related Work

The clustering methods developed in the literature can be identified in three types [15]: those based on an attempt to find the optimal partition into a specified number of clusters (for instance, the standard K-means method), those based on a hierarchical attempt to discover cluster structure (like the centroid-based agglomerative hierarchical clustering), and those based on a probabilistic model for the underlying clusters (there is an assumed probability model for each component cluster). Some methods have been developed to handle categorical data [10, 12, 14].

Usual criterion functions yield satisfactory results for numeric attributes but are not appropriate when examples include categorical attributes. It is not easy to define distances between values of categorical attributes [6]. The most common way of adapting such measures is to count the overlap of attributes having same values in two examples [18]. This technique may fail to capture the complexity and subtlety of the problem domain. For instance, Guha et al. [12] show that large clusters can be split even though examples in the cluster are well connected. There are several commonly used similarity measures such as the Cosine and the Jaccard similarity coefficients used for document clustering. Nevertheless, these measures have drawbacks. For instance, the Jaccard coefficient fails to capture the natural clustering of not so well-separated examples described with categorical attributes [12].

Among other things, due to the development of web usage mining (e. g., to identify strong correlations among user interests by grouping their navigation paths), there is a tendency for clustering with categorical attributes and recent research in Knowledge Discovery in Databases (KDD) revisits this question. More precisely, two main families of clustering methods based on association rules exist [1]. Association rules represent something like "It is frequent that when properties A_1 and A_2 are true within an example, then property A_3 tends to be true. As usual in this community, we refer to an attribute-value a pair as an *item* (for instance, `country = 'England'`).

Han et al. [13] present a method of association rules hypergraph k-partitioning. It takes a set of association rules and declares the items in the rules to be vertices, and the rules themselves to be hyperedges. Since each association rule has a direction, the algorithm combines all rules with the same set of items, and uses an average of the confidence of the individual rules as the weight for a hyperedge. Clusters are then found by a hypergraph partitioning method [16]. A clustering of items is obtained, but examples are not straightforwardly ranked into clusters. Ronkainen [21] defines similarities between items in large data sets to form hierarchies of clusters of items using agglomerative techniques. This family of methods produces a clustering of items and it may not be easy to rank examples within clusters.

So, the second family of methods starts by building a partition of examples. Wang et al. [24] suggest a clustering algorithm which groups examples in order to minimize intra-cluster and inter-cluster costs. Typically, an intra-cluster cost measures differences between elements within the cluster and an inter-cluster cost indicates the duplication of items among different clusters. This strategy gives a partition of the examples, but it is not able to derive easily a characterization of each cluster.

3 Understanding Required Properties for Clusters

In this section, we present the minimal properties required for discovering interesting clusters and how these properties are taken into account in large databases.

3.1 Concepts Lattice

Let $\mathcal{D} = (\mathcal{T}, \mathcal{I}, \mathcal{R})$ be a data mining context, $\mathcal{T}$ a set of examples (i. e. transactions), $\mathcal{I}$ a set of items (denoted by capital letters), and $\mathcal{R} \subseteq \mathcal{T} \times \mathcal{I}$ is a binary relation between examples and items. In the following, we use the term of *transaction* instead of *example* because it is the most common term in KDD. Each couple $(t, i) \in \mathcal{R}$ denotes the fact that the transaction t is related to the item i. A transactional database is a finite and non empty multi-set of transactions. Table 1 provides an example of a transactional database consisting of 8 transactions (each one identified by its Id) and 9 items denoted $A \dots I$. This table is used as support for the examples in the rest of the paper. An itemset is a subset of $\mathcal{I}$ (note that we use a string notation for sets, e.g., AB for $\{A, B\}$). A transaction t supports an itemset X iff $X \subseteq t$.

Id	Items
1	$A\ B\ C$
2	$A\ B\ C$
3	$A\ B\ C$
4	$D\ E$
5	$D\ E\ H$
6	$A\ D\ E\ F\ G\ H$
7	$A\ F\ G\ I$
8	$H\ I$

Table 1: A transactional database

Given $\mathcal{D}$, there is a unique ordered set which describes the inherent lattice structure defining natural groupings and relationships among the transactions and their items. This structure is known as a concepts lattice or Galois lattice [11][25]. Each element of the lattice is a couple (T, I) composed of a set of transactions and an itemset. Each couple (named *concept* by Wille [25]) must be a complete couple with respect to $\mathcal{R}$, which means that the following two properties (noted f and g) are satisfied. For $T \subseteq \mathcal{T}$ and $I \subseteq \mathcal{I}$, we have:

$$f(T) = \{i \in \mathcal{I} | \forall t \in T, (t, i) \in \mathcal{R}\}$$
$$g(I) = \{t \in \mathcal{T} | \forall i \in I, (t, i) \in \mathcal{R}\}$$

$f(T)$ associates with T, items common to all transactions $t \in T$, and $g(I)$ associates with I, transactions related to all items $i \in I$. In other words, T is the largest set of transactions described by the items found in I, and symmetrically, I is the largest set of items common to the transactions supporting I. For instance, in Table 1, the couple composed of transactions 1, 2 and 3 on one side and the items ABC on the other side is a concept of the lattice whereas there is no couple composed of transactions 1, 2 (since transaction 3 shares the same items as transactions 1 and 2). The idea of maximally extending the sets is formalized by the mathematical notion of *closure* in ordered sets and the operators $h = f \circ g$ and $h\prime = g \circ f$ are the Galois closure operators.

The idea of maximally extending the sets is on the core to highlight meaningful clusters. Indeed, for a group of transactions, we prefer to simply produce the single itemset which is composed of the maximal number of items shared by the group. The key point is to capture the maximum amount of similarity among the data. The next section defines the notion of closed itemsets and shows that only frequent closed itemsets are relevant in the discovery of meaningful clusters.

3.2 Frequent Closed Itemsets

Let X be an itemset. X is a closed itemset iff $h(X) = X$. In other words, a closed itemset is a maximal set of items (with respect to set inclusion) shared by a set of transactions. Then, a candidate cluster is a concept of the closed itemsets lattice and is composed of an itemset X and its transactions $g(X)$.

In our example (see Table 1), ABC is a closed itemset because it is the maximum set of items shared by transactions containing at least ABC. On the contrary, AB is not a closed itemset since it is not a maximal group of items common to the data: all transactions having the items AB also get the item C. In other words, we can add item C to AB without decreasing its frequency.

The frequency of X is $\mathcal{F}(X) = |g(X)|$ (where as usual $|\dots|$ denotes the cardinality of a set). Note that we use here the absolute frequency (a number of examples $\leq |\mathcal{T}|$) instead of the relative frequency $|g(X)|/|\mathcal{T}|$ in $[0, 1]$. An itemset X is *frequent* if its frequency is at least the frequency threshold $minfr$ fixed by the user. For the following, X denotes a frequent closed itemset, and L the set of the frequent closed itemsets. The frequency is fundamental to extract reliable clusters. It allows to take into account the "importance" (in term of "weight") of a candidate cluster and discard clusters which do not rely on sound relationships within data. A cluster with too few transactions would not be kept by a user. From large databases, there are efficient algorithms [3, 20, 22] to compute the frequent closed itemsets and give the associated transactions.

Let us come back to our example (Table 1). If we set *minfr* to 2, DE is frequent because its frequency is 3 (transactions 4, 5 and 6 contain DE) and DEF is not frequent (its frequency is 1 since only transaction 6 contains DEF). Seven frequent closed itemsets are extracted from Table 1 (Table 2 shows these frequent closed itemsets, noted FCI, with their frequencies).

These two points (the capture of the maximum amount of similarity - i.e. closed itemsets - and the notion of frequency) are the basis of the method of selection of meaningful clusters that we present below.

FCI	Transactions	Homogeneity	Concentration	Interestingness
I	7 8	0.333	0.416	0.375
H	5 6 8	0.272	0.344	0.308
AFG	6 7	0.6	0.266	0.433
DE	4 5 6	0.545	0.511	0.528
ABC	1 2 3	1	0.5	0.75
A	1 2 3 6 7	0.263	0.406	0.334
DEH	5 6	0.666	0.266	0.466

Table 2: Interestingness of frequent closed itemsets (FCI)

4 ECCLAT: Extraction of Clusters from Concepts LATtice

Starting from the frequent closed itemsets (seen as clusters), we propose here a method, based on the definition of a cluster evaluation measure, to select the most interesting concepts gathering similar transactions.

4.1 Cluster Evaluation Measure

As we have seen, a relevant cluster has to be as homogeneous as possible and should gather "enough" transactions. Translated into the usual clustering framework, it means that we have to maximize the intra-cluster similarity (which we call *homogeneity*) and minimize the inter-clusters similarity. We use a *concentration* measure to limit the overlapping of transactions between clusters (a relevant cluster should concentrate some transactions). We will see that a slight overlapping may be understandable and useful in some applications.

Let us start by defining homogeneity and concentration. For homogeneity, we want to favor clusters having many items shared by many transactions. Homogeneity of a cluster X is computed from its size, $\mathcal{F}(X)$ (i.e. its number of transactions) and a divergence measure. The *divergence* is the number of items not in X, for each transaction of $g(X)$.

$$homogeneity(X) = \frac{\mathcal{F}(X) \times |X|}{divergence(X) + (\mathcal{F}(X) \times |X|)}$$

where $divergence(X) = \sum_{t \in g(X)} |f(t) - X|$.

We have $0 \leq homogeneity(X) \leq 1$. If a cluster is pure (i.e. $\forall t \in g(X)$ $f(t) = X$), its divergence is equal to 0, and its homogeneity equals 1. The more a cluster supports transactions with items not belonging to X, the more its homogeneity tends towards 0. Let us remark that the homogeneity of a cluster X depends only on X and can be computed simultaneously to X.

Following our previous example, Table 2 gives all frequent closed itemsets with their transactions, homogeneity and concentration measures.

We have $homogeneity(ABC) = (3 \times 3)/((0+0+0)+(3 \times 3)) = 1$: all items of each transaction supporting ABC belong to ABC. For DE, $homogeneity(DE) = (3 \times 2)/((0+1+4)+(3 \times 2)) = 0.545$: transaction 5 has an item which diverges (i.e. does not belong to the closed DE) and four items diverge for transaction 6.

For concentration, we want to favor clusters having transactions appearing the least in the whole set of clusters. Concentration limits the overlapping of transactions between selected clusters. Concentration of a cluster X is defined by taking into account the number of clusters where each transaction appears.

$$concentration(X) = \frac{1}{|g(X)|} \times \sum_{t \in g(X)} \frac{1}{\mathcal{F}'(t)}$$

where $\mathcal{F}'(t)$ is the number of clusters where t occurs (i.e. absolute frequency of t in L).

We have $0 < concentration(X) \leq 1$. If all transactions $g(X)$ occur only in X, then $concentration(X) = 1$. The more frequent are the transactions of $g(X)$ in the whole set of clusters, the more $concentration(X)$ tends towards 0.

In our example, we have $concentration(ABC) = 1/3 \times (1/2 + 1/2 + 1/2) = 0.5$ (each transaction supporting ABC belongs to two closed frequent itemsets). For DE, $concentration(DE) = 1/3 \times (1/1 + 1/3 + 1/5) = 0.511$. Transaction 4 only supports the closed itemset DE while transaction 5 supports three closed itemsets and transaction 6 supports five closed itemsets.

Finally, we define the *interestingness* of a cluster as the average of its homogeneity and concentration. We have $0 \leq interestingness(X) \leq 1$.

$$interestingness(X) = \frac{homogeneity(X) + concentration(X)}{2}$$

We have $interestingness(ABC) = (1 + 0.5)/2 = 0.75$ (the most interesting cluster in our example) and $interestingness(DE) = (0.545 + 0.511)/2 = 0.528$.

The idea is to select clusters with high interestingness and the next section presents an algorithm for this task.

4.2 Clusters Procedure: Selection Algorithm

We use the interestingness defined above to select clusters from the frequent closed itemsets lattice. An innovative feature of ECCLAT is its ability to produce a clustering with a minimum overlapping between clusters (which we call "*approximate clustering*") or a set of clusters with a slight overlapping. This functionality depends on the value of a parameter called M. M is an integer corresponding to a number of transactions not yet classified that a new selected cluster must classify. If $minfr$ is low, all transactions appear at least in one frequent closed itemset. So with $M = 1$, we assure to classify all transactions in the selected cluster. Nevertheless, a slight overlapping between clusters may appear. M should be set near 1 if we are interested in discovering meaningful clusters. The more the value of M increases, the more the overlapping decreases but some transactions may not belong to any cluster. We refer to these unclustered transactions as *trash* (i.e. remaining transactions are grouped in a trash cluster). Experimental results (see Section 5) show that the choice of a value of M near $minfr$ performs good results in approximate clustering.

The sketch of the algorithm is the following. First, the interestingness of each cluster of L is computed (L is the set of the frequent closed itemsets, see Section 3.2). The cluster having the highest interestingness is selected. Then as long as there are transactions to classify (i.e. which do not belong to any selected cluster) and clusters remain, we select the cluster having the highest interestingness and containing at least M transactions not yet classified.

The results of the method are linked to the value of M. Let us illustrate these different behaviors with our example. Let us start with $M = 1$. The cluster which has the highest interestingness (see Table 2) is ABC, so it is selected. Transactions 1, 2 and 3 are classified. Then cluster DE is chosen and transactions 4, 5 and 6 belong to this cluster. Transactions 7 and 8 remain. Cluster DEH is skipped because neither transaction 7 nor transaction 8 supports it. Then cluster AFG is selected because it classifies transaction 7. Transaction 8 is the only remaining one. Finally, cluster I is selected. All transactions are classified in four clusters. We get the following overlapping: transaction 6 is both in DE and AFG and transaction 7 belongs to AFG and I. Intuitively, when we observe transactions 6 and 7 (see Table 1), there is no sound reason to classify them in one cluster or the other. Note that item A appears in two clusters.

With $M = 2$, we get clusters ABC, DE and I. AFG is not selected because it only classifies one remaining transaction. There is no overlapping and we obtain a partition. Note that here $M = minfr$.

With $M = 3$, only two clusters are selected (ABC and DE) and a trash cluster is built with transactions 7 and 8.

From the computational point of view, it is obvious that the lower $minfr$, the more there are frequent closed itemsets and more time is required. With regard to M, the higher M, the number of constraints that needs to be checked between clusters increases and this reduces system efficiency.

5 Experiments

We have tested our method on the well-known data set Mushroom[1], and on proxy server logs coming from France Telecom R&D.

5.1 Clustering the Mushroom Data Set

The Mushroom data set includes 8124 transactions and 116 items. Each transaction has a class value (`edible` or `poisonous`). For experimentation purposes, the class is ignored, but it is used afterwards for the assessment of

[1]http://www.ics.uci.edu/∼mlearn/MLRepository.html

the results. To simplify presentation of results, we use a relative value (i.e. percentage) for $minfr$ and M. For instance, since the total number of data is 8124, $minfr = 5\%$ means 406 transactions using an absolute frequency ($5\% = 0.05 \simeq 406/8124$).

$minfr$ %	No. of frequent closed itemsets	No. of selected clusters	ratio %
50	44	10	22.7
40	113	13	11.5
30	301	21	6.9
20	888	30	3.4
10	3581	61	1.7
5	9738	144	1.5

Table 3: Number of selected clusters according to the $minfr$ value (Mushroom, $M = 1$)

Table 3 shows the number of closed frequent itemsets and the number of selected clusters for different values of $minfr$. In order to discover all meaningful clusters, we set M to 1. The number of selected clusters is much lower than the number of frequent closed itemsets. When $minfr$ decreases, we see that the number of selected clusters does not increase as much as the number of frequent closed itemsets (in other words, ratio decreases).

M	No. of selected clusters	No. of transactions in trash	No. of common transactions (avg.)
1 (0.01%)	144	0	40
101 (1.25%)	43	60	29
203 (2.5%)	28	44	28
304 (3.74%)	21	60	15
406 (5%)	17	340	7
507 (6.2%)	13	442	16
609 (7.5%)	11	572	16
812 (10%)	9	479	73

Table 4: Results of Ecclat according to the M value (Mushroom, $minfr = 5\%$)

Table 4 gives the number of selected clusters, the number of transactions in the trash cluster and the average number of common transactions between all pairs of selected clusters (i.e. overlapping) according to M ($minfr = 5\%$). With $M = 1$ all transactions are classified (because $minfr$ is low enough), but some overlapping remains. One observes that with $M = 5\%$ (which is also the

$minfr$ value) the overlapping is minimal. Too large a value of M leads to a trash cluster with many transactions and an increase of overlapping. Experimentally, $M = minfr$ is a good choice to achieve an approximate clustering.

For $minfr = 5\%$ (9738 frequent closed itemsets) and $M = minfr$, we obtain 16 clusters and a trash cluster (see Table 5)[2]. Slight overlapping involves only clusters 14 and 16.

cluster	#poisonous	#edible
1	0	432
2	0	432
3	0	432
4	0	432
5	648	0
6	648	0
7	432	0
8	432	0
9	432	0
10	432	0
11	0	768
12	0	512
13	352	96
14	288	896
15	0	416
16	72	560
trash	180	160

Table 5: An approximate clustering with ECCLAT (Mushroom, $minfr$=5%)

cluster	#poisonous	#edible
1	0	94
2	0	13
3	0	6
4	26	682
5	30	2631
6	37	121
7	61	69
8	287	0
9	3388	61
10	77	372
11	0	9
12	10	19
13	0	21
14	0	110

Table 6: A clustering with the method of Wang et al. (Mushroom)

It is interesting to compare these results with those provided by the method of Wang et al. [24] (see Table 6). This method produces 14 clusters which are obtained hierarchically (by splitting trash clusters) and using several $minfr$ values. Results given by our approach seem more understandable: 13 clusters among 17 are pure and we did not use a hierarchical decomposition requiring several values for $minfr$. Let us recall that we used a low value for $minfr$, which is possible because ECCLAT is based on efficient algorithms for frequent closed itemsets mining with regard to the step of extraction of candidate clusters.

5.2 Proxy Server Logs

In this experiment, we used some proxy server logs coming from France Telecom R&D. This data contained 136 transactions and 17,270 items. Items are

[2]The fact that several clusters have the same number of transactions is an amazing coincidence.

keywords of the HTML pages browsed by 136 users of a proxy-cache, over a period of 1 month. 18,162 pages were viewed. For every page, we extracted at most 10 keywords with an extractor based on the frequency of significant words. This extractor was developed at France Telecom R&D, and used in several studies, notably the variability of the users' thematic profile [17].

$minfr$ %	No. of frequent closed itemsets	No. of selected clusters	ratio %
60	84	8	9.52
55	295	8	2.71
50	981	10	1.01
45	4482	13	0.29
40	21507	16	0.07
35	109237	25	0.02

Table 7: Number of selected clusters according to the $minfr$ value (Proxy server logs, $M = 1$)

Table 7 represents the number of selected clusters with $M = 1$. As previously, the number of selected clusters is much lower than the number of frequent closed itemsets.

M	No. of selected clusters	No. of transactions in trash	No. of common transactions (avg.)
1 (0.7%)	16	0	50
3 (2.2%)	9	3	40
7 (5.1%)	5	14	31
10 (7.3%)	4	21	30
13 (9.5%)	4	12	45
27 (20%)	3	14	28
40 (30%)	2	45	0
54 (40%)	2	45	0

Table 8: Results of ECCLAT according to the M value (Proxy server logs, $minfr = 40\%$)

Table 8 gives the number of selected clusters, the number of transactions in the trash cluster and the average number of common transactions between all pairs of selected clusters according to M ($minfr = 40\%$). When the value of M increases, note that the number of selected clusters lowers drastically. All transactions are ranked in a cluster with $M = 1$ with an average overlapping of 50 transactions.

In this experiment, we obtain clusters of users according to their interests (consulted keywords). We think that it is useful that all users are classified in at least one cluster (which is allowed with $M = 1$) and the overlapping is suitable for some applications. For instance, a user may be included in a cluster corresponding to `fishing` and `England` and also in a cluster characterized by `bike` and `mountain`. The overlapping allows to retrieve such a user from several kind of queries (i.e. taxonomies corresponding to several points of view). Note that a "pure" clustering algorithm (i.e. producing a partition of transactions) assigns the user to a single cluster, that means a single point of view. The obtained clusters can also be used for personalization (e.g., web pages), for recommendation of web sites or documents to users, and also for the prefetching of information on a proxy-cache [8, 23] (for example, for the selected clusters, we can load in the cache documents indexed by the keywords of clusters).

6 Conclusion

We propose a new method (Ecclat) to build approximate clustering and discover meaningful clusters from categorical data. Such clusters correspond to concepts selected from the frequent closed itemsets lattice. With regard to the step of extraction of candidate clusters, efficiency of algorithms for frequent closed itemsets mining allows to tackle large databases. Furthermore, we think that it is an original use of the frequent closed itemsets. Unlike existing techniques, our approach does not use a global measure of similarity between transactions but is based on an evaluation measure of a cluster. The number of clusters is not fixed beforehand.

Ecclat is generic because it allows to build an approximate clustering where overlapping is minimized or to discover a set of clusters with a slight overlapping. We claim that in some situations, like web mining applications, a set of clusters (and not necessarily a partition) is required. For instance, overlapping is suitable to cluster pages or web page suggestion.

We have tested our method on the well-known Mushroom data set where we looked for an approximate clustering, and on web data coming from France Telecom R&D where we searched clusters of users. In all experiments, we show that the number of selected clusters is very low with respect to the number of frequent closed itemsets.

As exceptions are common in real-world data, further work will be to relax the constraint of closure to allow for some exceptions in a cluster.

References

1. Agrawal, R. & Imielinski, T. & Swami, A. Mining Association Rules between Sets of Items in Large Database. In Proceedings of ACM SIGMOD 93, pages 207-216, ACM Press, 1993

2. Agrawal, R. & Srikant, R. Fast Algorithms for Mining Association Rules. In Proceedings of the 20th VLDB Conference, Santiago, Chile, 1994

3. Boulicaut, J.F. & Bykowski, A. Frequent Closures as a Concise Representation for Binary Data Mining. In Proceedings of the Fourth Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD 00, volume 1805 of LNAI, pages 62-73, Kyoto, Japan, 2000. Springer-Verlag

4. Carpineto, C. & Romano, G. Galois: An Order-Theoretic Approach to Conceptual Clustering. In Proceedings of the 10th International Conference on Machine Learning, ICML 93, pages 33-40, Amherst, USA, June 1993

5. Crémilleux, B. & Gaio, M. & Madelaine, J. & Zreik, K. Discovering Browsing Paths on the Web. In Proceedings of the 7th International Conference on Human-System Learning, pages 9-18, Paris, France, 2000. Europia

6. Das, G. & Mannila, H. Context-based Similarity Measures for Categorical Databases. In Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD 00, volume 1910 of LNAI, pages 201-210, Lyon, France, 2000. Springer-Verlag

7. Durand, N. & Crémilleux, B. & Henry-Amar, M. Discovering Associations in Clinical Data: Application to Search for Prognostic Factors in Hodgkin's Disease. In Proceedings of the 8th Conference on Artificial Intelligence in Medecine in Europe, AIME 01, volume 2101 of LNAI, pages 50-54, Cascais, Portugal, July 2001. Springer-Verlag

8. Durand, N. & Lancieri, L. Study of the Regularity of the Users' Internet Accesses. In Proceedings of the 3rd International Conference on Intelligent Data Engineering and Automated Learning, IDEAL 02, volume 2412 of LNCS, pages 173-178, Manchester, UK, August 2002. Springer-Verlag

9. Fisher, D.H. Knowledge Acquisition via Incremental Conceptual Clustering. Machine Learning, 2:139-172, 1987

10. Ganti, V. & Gehrke, J. & Ramakrishnan, R. CACTUS: Clustering Categorical Data Using Summaries. In Proceedings of the 5th ACM SIGMOD International Conference on Knownledge Discovery and Data Mining, pages 73-83, New-York, August 1999

11. Godin, R. & Missaoui, R. & Alaoui, H. Incremental Concept Formation Algorithms based on Galois (concept) Lattices. Computational Intelligence, 11(2):246-267, 1995

12. Guha, S. & Rastogi, R. & Shim, K. ROCK: A Robust Clustering Algorithm for Categorical Attributes. In Proceedings of the 15th International Conference on IEEE Data Engineering, ICDE 99, pages 512-521, 1999

13. Han, E.H. & Karypis, G. & Kumar, V. & Mobasher, B. Hypergraph Based Clustering in High-Dimensional Data Sets : a Summary of Results. Bulletin of the Technical Commitee on Data Engineering, 21(1), 1998

14. Huang, Z. Extensions to the K-Means Algorithm for Clustering Large Data Sets with Categorical Values. Data Mining and Kownledge Discovery, 2(3):283-304, 1999

15. Jain, A.K. & Dubes, R.C. Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs, New Jersey, 1988

16. Karypis, G. & Han, E.H.S. Concept Indexing: a Fast Dimensionality Reduction Algorithm with Applications to Document Retrieval and Categorization. Technical Report TR-00-0016, University of Minnesota, 2000

17. Legouix, S. & Foucault, J.P. & Lancieri, L. A Method for Studying the Variability of Users' Thematic Profile. In Proceedings of WebNet2000, Association for the Advancement of Computing in Education (AACE), San Antonio, 2000

18. Liu, W.Z. & White, A.P. Metrics for Nearest Neighbour Discrimination with Categorical Attributes. In Proceedings of the Seventh International Annual International Conference of the British Computer Society Specialist Group on Expert Systems (ES 97), Cambridge, UK, 1997

19. Michalski, R.S. & Stepp, R.E. Learning from Observation: Conceptual Clustering. In Michalski, R. S. & Carbonell, J. G. & Mitchell, T. M. (eds), Machine Learning, An Artificial Intelligence Approach, volume 1, pages 331-363. Morgan Kauffmann, 1983

20. Pasquier, N. & Bastide, Y. & Taouil, R. & Lakhal, L. Efficient Mining of Association Rules Using Closed Itemset Lattices. Information Systems, 24(1):25-46, Elsevier, 1999

21. Ronkainen, P. Attribute Similarity and Event Sequence Similarity in Data Mining. Technical Report C-1998-42, University of Helsinki, October 1998

22. Stumme, G. & Taouil, R. & Bastide, Y. & Pasquier, N. & Lakhal, L. Computing Iceberg Concept Lattices with TITANIC. Journal on Knowledge and Data Engineering, 2002

23. Wang, J. A Survey of Web Caching Schemes for the Internet. ACM Computer Communication Review, 29(5):36-46, October 1999

24. Wang, K. & Chu, X. & Liu, B. Clustering Transactions Using Large Items. In Proceedings of the ACM Conference on Information and Knowledge Management, CIKM 99, USA, 1999

25. Wille, R. Ordered Sets, chapter Restructuring Lattice Theory: an Approach based on Hierachies of Concepts, pages 445-470. Reidel, Dordrecht, 1982