

# Rare Association Rule Mining via Transaction Clustering

Yun Sing Koh<sup>1</sup>

Russel Pears<sup>2</sup>

School of Computing Science and Mathematics  
Auckland University of Technology, New Zealand,  
Email: ykoh@aut.ac.nz<sup>1</sup>, rpears@aut.ac.nz<sup>2</sup>

## Abstract

Rare association rule mining has received a great deal of attention in the recent past. In this research, we use transaction clustering as a pre-processing mechanism to generate rare association rules. The basic concept underlying transaction clustering stems from the concept of large items as defined by traditional association rule mining algorithms. We make use of an approach proposed by Koh & Pears (2008) to cluster transactions prior to mining for association rules. We show that pre-processing the dataset by clustering will enable each cluster to express their own associations without interference or contamination from other sub groupings that have different patterns of relationships. Our results show that the rare rules produced by each cluster are more informative than rules found from direct association rule mining on the unpartitioned dataset.

*Keywords:* Rare Association Rule Mining, Transaction Clustering, Apriori-Inverse

## 1 Introduction

The main goal of association rule mining is to discover relationships among sets of items in a transactional database. Association rule mining was introduced by Agrawal et al. (1993). It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in transaction databases or other data repositories. The relationships are not based on inherent properties of the data themselves but rather based on the co-occurrence of the items within the database. The associations between items are also known as association rules. In the classical association rule mining process, all frequent itemsets are found, where an itemset is said to be frequent if it appears with minimum frequency  $s$ , called minimum support. Association rules are then derived from frequent items and are represented in the form  $A \rightarrow B$  where  $AB$  is a frequent itemset. Strong association rules are those that meet the minimum confidence  $c$  threshold (the percentage of transactions containing  $A$  that also contain  $B$ ).

A much less explored area in association mining is infrequent itemset mining or rare association rule mining. Items that rarely occur are in very few transactions and are normally pruned out. One limitation of common association rule mining approaches, i.e. Apriori, are that they rely on there being a

meaningful minimum support level that is reasonable (sufficiently strong) to reduce the number of frequent itemsets generated to a manageable level. However, in some data mining applications relatively infrequent associations are likely to be of great interest as they relate to rare but crucial cases. Examples of mining rare itemsets include identifying relatively rare diseases, predicting telecommunication equipment failure, and finding associations between infrequently purchased supermarket items. Indeed, infrequent itemsets warrant special attention because they are more difficult to find using traditional data mining techniques.

In this paper we first pre-process the dataset by clustering transactions before performing association rule mining. The rationale behind clustering transactions prior to mining association rules is that the latter is performed on partitions that are essentially distinct from each other. Each cluster would be expected to contain associations without interference or contamination from other sub groupings that have different patterns of relationships. We thus adopt a two phased approach. The first phase comprises the transaction clustering phase and adopts the clustering method proposed by Koh & Pears (2008). In the second phase we generate rare rules based on the clusters generated in the initial phase.

The basic concept underlying transaction clustering stems from the concept of large items as defined by association rule mining algorithms. Currently, none of the techniques proposed offer a good solution to scenarios where large items overlap across clusters. A further limitation with some of the existing algorithms is that they rely on some form of domain specific knowledge, thus limiting their range of applicability. Koh & Pears (2008) overcome the aforementioned limitations by using cluster seeds that represent initial centroids. Seeds are generated from sets of transaction items that occur together above a certain threshold and such seeds may overlap in their itemsets across clusters.

In the second phase we run Apriori-Inverse (Koh & Rountree 2005) on the clusters generated. In this approach we consider itemsets that are above a minimum absolute support requirement (Koh et al. 2008) and below a maximum support threshold. We show that we find more informative rules compared with Apriori-Inverse on the unclustered dataset.

The remainder of this paper is organised as follows. Section 2 provides a review of related research. In Section 3 we introduce the notion of transaction clustering by seeding. Section 4 describes how rare association rule mining is applied on the clusters produced with transaction clustering. Experimental results of applying the method on several real-world datasets is presented in Section 5. The paper concludes in Section 6 with a summary of the contributions made in this research.

## 2 Related Work

In this section we look at three related areas, clustering transaction, association rule mining, and the combination of the two.

### 2.1 Clustering Transactions

In the recent past there has been an increasing level of interest in transaction clustering. All such approaches have employed quite different methods when compared to traditional clustering methods. Wang et al. (1999) utilised the concept of large items (Agrawal et al. 1993) to cluster transactions. Their approach measures the similarity of a cluster based on the large items in the transaction dataset. Each transaction is either allocated to an existing cluster or assigned to a new cluster based on a cost function. The cost function measures the degree of similarity between a transaction and a cluster based on the number of large and small items shared between that transaction and the given cluster.

To speed-up the method proposed above, Yun et al. (2001) introduced a method called SLR (Small-Large Ratio). Their method essentially uses the measurement of the ratio between small to large items to cluster transactions. Both the large item (Wang et al. 1999) and SLR (Yun et al. 2001) method suffers a common drawback. In some cases, they may fail to give a good representation of the clusters. Suppose that A and B are large items in a transaction dataset, with A and B occurring individually 60% of the time and AB occurring together 40% of the time. If the support threshold is set at 40%, then the cost function that they use results in two clusters, one having transactions that contain the item A and the other that contains item B. However, in this case, it is clear that the optimal cluster configuration requires an additional cluster containing the itemset AB. Their approach thus tends to discourage bonding between itemsets already occurring in other clusters. This in turn forces transactions to choose between sub-optimal clusters when deciding what cluster that they should belong to.

Xu et al. (2003) proposed a method using the concept of a caucus. The basic idea of introducing a caucus to cluster transactions is motivated by the fact that cluster quality is sensitive to the initial choice of cluster centroids (Xu et al. 2003). Fundamentally different from most other clustering algorithms, their approach attempts to group customers with similar behaviour. In their approach they first determine a set of background attributes from the dataset that are significant. A set of caucuses, consisting of different subsets of items is then constructed to identify the initial cluster centroids.

The main drawback of this method is that it requires the user to define the initial centroids which is difficult as it requires some form of prior knowledge about the dataset. The cluster seeding method Koh & Pears (2008) overcomes the two main issues with the current approaches in that it copes well with overlapping centroids and does not require background domain specific knowledge.

### 2.2 Rare Association Rule Mining

Detecting sporadic association rules, rules with low support but high confidence efficiently is a difficult data mining problem. To find these rules in traditional approaches, such as the Apriori algorithm, minimum support (minsup) has to be set very low, which results in a large amount of redundant rules. As a specific example of the problem, consider the

association mining problem where we want to determine if there is an association between buying a food processor and buying a cooking pan (Liu et al. 1999). The problem is that both items are rarely purchased in a supermarket. Thus, even if the two items are almost always purchased together when either one is purchased, this association may not be found. Modifying the minsup threshold to take into account the importance of the items is one way to ensure that rare items remain in consideration. To find this association minsup must be set low. However setting this threshold low would cause a combinatorial explosion in the number of itemsets generated. Frequently occurring items will be associated with one another in an enormous number of ways simply because the items are so common that they cannot help but appear together. This is known as the rare item problem (Liu et al. 1999). It means that using the Apriori algorithm, we are unlikely to generate rules that may indicate rare events of potentially dramatic consequence.

Liu et al. (1999) note that some individual items can have such low support that they cannot contribute to rules generated by Apriori, even though they may participate in rules that have very high confidence. They overcome this problem with a technique called MSApriori whereby each item in the database can have a minimum item support (MIS) given by the user. By providing a different MIS for different items, a higher minimum support is tolerated for rules that involve frequent items and a lower minimum support for rules that involve less frequent items. Yun et al. (2003) proposed the RSAA algorithm to generate rules in which significant rare itemsets take part, without any "magic numbers" specified by the user. This technique uses relative support: RSup is used in place of support. Thus, this algorithm decreases the support threshold for items that have low frequency and increases the support threshold for items that have high frequency.

Koh et al. (2008) proposed an approach to find rare rules with candidate itemsets that fall below a maxsup (maximum support) level but above a minimum absolute support value. They introduced an algorithm called Apriori-Inverse to find sporadic rules efficiently: for example, a rare association of two common symptoms indicating a rare disease. They later proposed another approach called MIISR. In their approach, the consequent of these rules is an item below maxsup threshold and the antecedent has support below maxsup but may consist of individual items above maxsup. In both approaches they use minimum absolute support (minabssup) threshold value derived from an inverted Fisher's exact test to prune out noise. At the low levels of co-occurrences of candidate itemsets that need to be evaluated to generate rare rules, there is a possibility that such co-occurrences happen purely by chance and are not statistically significant. The Fisher test provided a statistically rigorous method of evaluating significance of co-occurrences and was thus an integral part of their approach.

Like Apriori and MSApriori, RSAA is exhaustive in its generation of rules, so it spends time looking for rules which are not sporadic (i.e. rules with high support and high confidence). If the minimum-allowable relative support value is set close to zero, RSAA takes a similar amount of time to that taken by Apriori to generate low-support rules in amongst the high-support rules.

### 2.3 Combining Clustering and Association Rule Mining

Recently, Plasse et al. (2007) proposed a method of analysing links between binary attributes in a large sparse data set. Initially the variables are clustered to obtain homogeneous clusters of attributes. Association rules are then mined in each cluster. Plasse et al. (2007) used several clustering methods and compared the resulting partitions. They generated their clusters based on hierarchical methods which are divided into two groups: ascendant methods based on an agglomerative algorithm and descendant methods performed by a divisive algorithm. The similarity coefficients used in their clustering technique includes Russel and Rao, Jaccard, Ochiai, and Dice. Once the clusters have been generated by the different techniques, association rules were produced on the different clusters. While their method did succeed in finding association rules that could not be discovered without clustering, the inherent weakness was in the clustering algorithms that they employed. None of the methods proposed offered a good solution to scenarios where large items overlap across clusters.

A further limitation with some of the existing transaction clustering algorithms is that they rely on some form of domain specific knowledge, thus limiting their range of applicability. Executing numerous different clustering methods and then generating rules based on each of the clusters produced becomes prohibitively expensive in certain cases. This is especially true when clustering is employed over a variety of datasets from different domains.

In the next section we examine in detail the transaction clustering approach that we adopt. We show that the process of transaction clustering is fundamentally different from that of traditional clustering and discuss the specific concepts and methods that are required to generate high quality clusters containing a high degree of homogeneity of transactions.

The clustering algorithm that we describe achieves a much higher degree of homogeneity of items within a cluster, with either all or a large percentage of its items falling into the frequent category. Furthermore, the degree of heterogeneity across clusters was also significantly greater than with the Large Item approach, with very few frequent items being duplicated across clusters (Koh & Pears 2008). Results on a range of real world datasets showed that it significantly outperformed its Large Item counterpart in both respects. We believe that cluster quality is crucial in discovering association rules that would otherwise be undetectable via mining on an unclustered dataset, and this motivated us to choose the approach proposed by (Koh & Pears 2008).

### 3 Transaction Clustering By Seeding

Clustering is the process of finding naturally occurring groups in data. Clustering is one of the most widely studied techniques in the context of data mining and has many applications, including disease classification, image processing, pattern recognition, and document retrieval. Traditional clustering techniques deal with horizontal segmentation of data, whereby clusters are formed from sets of non-overlapping instances. Many efficient algorithms exist for the traditional clustering problem (Jain et al. 1999, Ganti et al. 1999, Guha et al. 2000). In contrast, transaction clustering has fundamentally different requirements, and has been gaining increasing attention in recent years. Unlike traditional clustering, transaction clustering requires that transactions be partitioned across clusters in such a manner that instances within a clus-

ter share a common set of large items, where the concept of large follows the same meaning attributed to frequent items in association rule mining (Agrawal et al. 1993). Thus it is clear that transaction clustering requires a fundamentally different approach from the traditional clustering techniques. Compounding the level of difficulty is the fact that transaction data is known to have high dimensionality, sparsity, and a potentially large number of outliers (Xu et al. 2003).

Current research in both data mining and information retrieval suggests that transaction clustering functionality needs to extend well beyond a near neighbourhood search for similar instances (Wang et al. 1999, Cutting et al. 1992). This form of clustering provides a natural solution to many applications such as targeted marketing/advertising, discovering causes of diseases, and others.

In this paper we adopt a recent approach used by (Koh & Pears 2008) for transaction clustering that is based on an initial seeding of cluster centroids. Their approach consists of two phases: a seed generation phase followed by a transaction allocation phase. In the seed generation phase the seeds are identified in a progressive manner by a candidate generation process based on Apriori (Agrawal et al. 1993). Large items are extended in precisely the same manner as Apriori. The chi square significance test is used to ensure that only strongly associated items are joined together into an itemset. The improvement constraint restricts the growth of a seed to ensure that it only consists of items that increase the value of an improvement function. Once seeds are generated, the next phase assigns transactions to clusters. Each transaction is allocated to a cluster centroid with the highest similarity. Once all transactions have been allocated, the centroid is recalculated for each cluster. The new centroid consists of large items that reside in the cluster. Transactions are then reallocated to clusters on the basis of proximity to the new centroids that were defined. In order to determine the optimal value for the number of clusters, the allocation phase is repeated until the value of a fitness function reaches a plateau.

Let  $D = \{t_1, \dots, t_n\}$  be a set of transactions. Each transaction is a set of items  $\{i_1, \dots, i_m\}$ .  $C$  is a partition of the transaction,  $\{C_1, \dots, C_k\}$  of  $\{t_1, \dots, t_n\}$ . Each  $C_i$  is called a cluster. Overall the clustering is divided into two main phases: seed generation and allocation phases.

#### 3.1 Seed Generation Phase

We start by describing the method used for finding the optimal number of clusters. The initial choice of seeds are the large items in the dataset. A minimum support threshold,  $\theta$  is used to identify large items, where  $0 < \theta < 1$ . Any item in the dataset that has support above  $|D| * \theta$  is considered a large item. Let  $L_i$  denote the set of large items or large itemsets. The items  $L_i$  are extended to itemsets  $L_{i+1}$  in the same way as Apriori generates candidate frequent itemsets. For a large itemset to be considered a cluster seed the frequency of co-occurrence of all pairs of subsets within the seed must occur together with a frequency above a threshold value at a given significance level. This effectively ensures that cluster seeds of size  $\geq 2$  have items that co-occur together at a frequency that is statistically significant. In addition, all cluster seeds satisfy an improvement constraint when they are extended. This constraint is based on the concept of relative support.

**Definition 1 (Relative Support).** The relative support of an itemset  $X_k$  of size  $k$  is defined to be the ratio of the support of  $X_k$  to the support of  $Y_{k-1}$

which is that  $(k-1)$ -sized subset of  $X_k$  with the maximum support. Thus,

$$RS(X_k) = \frac{\text{supp}(X_k)}{\text{supp}(Y_{k-1})}$$

**Definition 2 (Extension of a Seed).** Given two existing seeds,  $X_{k-1}$  and  $Y_{k-1}$ ,  $X_{k-1}$  is extended to a new seed  $X_{k-1} \cup Y_{k-1}$  if and only if:

$$\phi(X_{k-1}, Y_{k-1}) > \chi_c^2,$$

$$RS(X_{k-1} \cup Y_{k-1}) - RS(X_{k-1}) > \sigma, \text{ and}$$

$$RS(X_{k-1} \cup Y_{k-1}) - RS(Y_{k-1}) > \sigma$$

where  $\phi$  denotes the chi square correlation coefficient,  $\chi_c^2$ , the chi square cut-off threshold at the  $c\%$  confidence level and  $\sigma$  is a user-defined threshold.

The rationale behind extension lies in the fact that the new itemset to be added to the seed has a statistically strong correlation with the existing seed and that the inclusion of the new itemset will improve the relative support of the seed above a user defined minimum threshold. The algorithm for the seed clustering phase is shown below in Figure 1.

**Algorithm for Seed Generation Phase**

Input: Transaction database  $D$ ,  $\theta$  value,  $\sigma$  value, universe of items  $I$

Output: Cluster Seeds,  $S = \{s_1 \dots s_k\}$

$k \leftarrow 1$

$s_k \leftarrow \{\{i\} | i \in I, \text{count}(\{i\}) \geq |D| * \theta\}$

while  $l_k \neq \emptyset$  do

$k \leftarrow k + 1$

$l_k \leftarrow \{x \cup y | x, y \in s_{k-1}, |x \cap y| = k - 2\}$

$s_k \leftarrow \{x \cup y | x \cup y \in l_k, \phi(x, y) \geq \chi_c^2, RS(x \cup y) - RS(y) > \sigma, RS(x \cup y) - RS(x) > \sigma\}$

end while

return  $\bigcup_{t=1}^{k-1} s_t$

Figure 1: Algorithm for Seed Generation Phase

### 3.2 Allocation Phase

The seeds produced in the initial phase are considered as the initial centroids for the clusters. In this phase, transactions are assigned to clusters on the basis of similarity to cluster centroids. In order to measure similarity we modified the Jaccard similarity coefficient. For each transaction,  $t$ , we calculate the similarity between  $t$  and the existing centroid,  $c_k$ . The similarity,  $sim$ , is between  $t$  and the  $c_k$  is calculated as:

$$\text{sim}(t, c_k) = \frac{|t \cap c_k|}{|t \cup c_k| - |t \cap c_k| + 1}$$

Given  $t_1 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}\}$  and  $c_1 = \{\{b\}, \{c\}\}$ , here  $t_1 \cap c_1 = \{\{b\}, \{c\}\}$  and  $t_1 \cup c_1 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}\}$ . Using our measure, the similarity between  $t_1$  and  $c_1$  is calculated as  $2/(5-2+1) = 0.5$ . The greater the overlap between  $t$  and  $C_k$ , the greater the value of  $sim$  coefficient.

Once all transactions are allocated to clusters, further refinement is accomplished by recomputing the centroids which may need to be updated with large items belonging to transactions allocated to a given cluster but not currently part of its centroid. The updating of centroids will result in the need for reorganisation of the clusters, thus the process of centroid update and cluster reorganisation will need to be repeated in tandem until a suitable point of stabilisation is reached. In order to determine the point at which stabilisation is reached, a fitness function adapted from particle swarm optimisation approach was used to find the optimal clusters. For all cluster  $\{C_1, \dots, C_k\}$ , the fitness function is calculated as:

$$J = \frac{1}{k} \sum_{j=1}^k \frac{\sum_{t \in C_j} d(t, c_j)}{|C_j|}$$

**Algorithm for Allocation Phase**

Input: Transaction database,  $D = \{t_1, \dots, t_n\}$ ,

Cluster Seed,  $S = \{s_1, \dots, s_k\}$

Output: Cluster,  $C = \{C_1, \dots, C_k\}$

$J_{prev} \leftarrow 0$

$C \leftarrow \{C_k \leftarrow \emptyset | k \in S\}$

/\* Assign transactions to clusters with the highest similarity \*/

$C \leftarrow \{C_k \cup t | \arg \max\{k \mapsto \text{sim}(t, s_k) | s_k \in S\}, t \in D\}$

/\* Removes the empty clusters \*/

$C \leftarrow \{C_k | C_k \neq \emptyset, C_k \in C\}$

$J_{curr} \leftarrow \frac{1}{|C|} \sum_{j=1}^{|C|} \frac{\sum_{t \in C_j} \text{sim}(t, C_j)}{|C_j|}$

/\* Refine clusters \*/

while  $J_{prev} < J_{curr}$  do

$J_{prev} \leftarrow J_{curr}$

$c \leftarrow \{c_k | \{i\} \in C_k, \text{count}(\{i\}, D) \geq |D| * \theta\}, C_k \in C\}$

$C \leftarrow \{C_k \cup t | \arg \max\{k \mapsto \text{sim}(t, c_k) | c_k \in c\}, t \in D\}$

$C \leftarrow \{C_k | C_k \neq \emptyset, C_k \in C\}$

$J_{curr} \leftarrow \frac{1}{|C|} \sum_{j=1}^{|C|} \frac{\sum_{t \in C_j} \text{sim}(t, c_j)}{|C_j|}$

end while

return  $C$

Figure 2: Algorithm for Allocation Phase

The fitness measure calculates the average similarity between every transaction in a cluster to its centroid and thus the intention is to maximise the fitness value generated. The algorithm for the allocation phase is shown in Figure 2 above.

### 4 Rare Association Rule Mining via Transaction Clustering (AICluster)

The following is a formal statement of association rule mining for transaction databases. Let  $I = \{i_1, i_2, \dots, i_m\}$  be the universe of items and  $D$  be a set of transactions, where each transaction  $T$  is a set of items. An association rule is an implication of the form  $X \rightarrow Y$ , where  $X \subset I$ ,  $X \cap Y = \emptyset$ , and  $X \cup Y \subset I$ .  $X$  is referred to as the antecedent of the rule, and  $Y$  as the consequent. The rule  $X \rightarrow Y$  holds in the transaction set  $D$  with confidence  $c\%$  if  $c\%$  of transactions in  $D$  that contain  $X$  also contain  $Y$ . The rule has support  $s\%$  in the transaction set  $D$ , if  $s\%$  of transactions in  $D$  contain  $XY$ . Throughout this article we shall use  $XY$  to denote an itemset that contains both  $X$  and  $Y$ .

**Apriori Inverse**

Input: Transaction Clusters (cluster), maxsup value

Output: Rare Itemsets,  $R$

$N \leftarrow |\text{cluster}|$

$Idx \leftarrow \text{invert}(\text{cluster}, I)$

$k \leftarrow 1$

$R_k \leftarrow \{\{i\} | i \in \text{dom } Idx, \text{count}(\{i\}, Idx) \geq 1\}$

while  $(L_k \neq \emptyset)$

$k \leftarrow k + 1$

$C_k \leftarrow \{x \cup y | x, y \in R_{k-1}, |x \cap y| = k - 2\}$

$R_k \leftarrow \{c | c \in C_k, \text{count}(\{i\}, Idx) > \text{minabssup}, \text{count}(\{i\}, Idx) < \text{maxsup}\}$

end while

return  $\bigcup_{t=2}^{k-1} R_t$

Figure 3: Algorithm for Apriori Inverse

Initially we cluster the transactions into different partitions and then mine each of the partitions for rare association rules. Our rule mining approach is based on the Apriori Inverse algorithm introduced by Koh & Rountree (2005). Apriori Inverse inverted the downward-closure principle of the Apriori algorithm; rather than all subsets of rules satisfying the minsup

lower bound support threshold, all subsets are under maximum support threshold (maxsup). Since making a candidate itemset larger cannot increase its support, all extensions are viable except those that fall under the minimum absolute support requirement. The minimum absolute requirement is necessary to detect noise. Those exceptions are pruned out, and are not used to extend itemsets in the next round. Figure 3 gives the pseudo code for Apriori Inverse.

Our approach is able to produce interesting rules which are not detected in Apriori Inverse. For example, consider a case where we are looking at diagnosis which leads to mortality in a medical scenario. When we partition the datasets into clusters, a trend we may notice is treatment which leads to a much higher mortality rate in the cluster corresponding to the intensive care section when compared to the rest of the dataset. This is however not very interesting. On the other hand, if we detect rules in clusters from the outpatient unit which refer to mortality, this instead would be considered more interesting as it represents relatively rare and unexpected events which deserve closer examination as to the circumstances that led to the fatalities. These types of rules may never have manifested with Apriori Inverse on the unclustered dataset.

We now offer a formal proof of AICluster’s rule coverage vis-a-vis the Apriori Inverse algorithm.

**Lemma 1.** *If a rule  $R$  exists on the unclustered data set then that rule must have confidence  $C_i > C_{min}$  on at least one cluster  $cl_i$ .*

*Proof.* Let  $N$  be the total number of clusters. Suppose that the Lemma was false and hence for all clusters,  $cl_i = 1, \dots, N$  we have  $C_i \leq C_{min}$  for rule  $R$ . (1)

Let the number of instances of the antecedent of the rule be  $L_i$  for the  $i$ th cluster,  $cl_i$ . Let the number of instances of the antecedent and the consequent occurring together in the rule be  $LR_i$  for the  $i$ th cluster. We now have  $L_i P_i = LR_i$  where  $P_i$  is essentially the confidence for  $i$ th cluster.

We now have  $\sum_{i=1}^N L_i P_i = \sum_{i=1}^N LR_i \leq C_{min} \times \sum_{i=1}^N L_i$  from (1) above. Let us denote  $\sum L_i P_i$  by  $LR$ . We thus have  $LR \leq C_{min} \times \sum_{i=1}^N L_i$ . (2)

Now consider the rule on the unclustered data set. We have  $LC = GLR$  where  $LC$  is the support of the antecedent of the rule and  $GLR$  is the number of instances of the antecedent and consequent occurring together in the rule. We also have  $C > C_{min}$  since the rule  $R$  exists on the unclustered data set and so we have  $GLR > C_{min} \times \sum_{i=1}^N L_i$  from (2) above.

We also have  $GLR = LR$  as the total number of occurrences taken across all clusters must be the same as the total number of instances across the unclustered data set, as the instances in the unclustered data set is the union of all instances in the clusters.

Substituting for  $GLR$  in the expression above we have  $LR > C_{min} \times \sum_{i=1}^N L_i$  (3), which leads to a contradiction with (2) above. Thus our initial assumption that the Lemma is false is untrue and this proves the Lemma.  $\square$

**Theorem 1.** *AICluster together with traditional frequent mining has a coverage that is greater than a combination of Apriori Inverse with traditional association rule mining.*

*Proof.* Once again consider a rule  $R$  that exists on the unclustered data set. Now suppose that

the rule  $R$  does not exist on any of the clusters  $cl_i$  where  $i = 1, \dots, N$ . Let us consider the case where rule  $R$  is not picked across any of the clusters.

$C_i \leq C_{min}$  or  $S_i \geq S_{max}$  (4) for all clusters  $i = 1, \dots, N$ ; where  $S_i$  denotes the support of rule  $R$  on cluster  $cl_i$  and  $S_{max}$  is the upper bound support threshold for finding rare rules.

According to the Lemma above there must exist at least one cluster where the confidence of the rule exceeds  $C_{min}$ . Let us pick one of these clusters at random, say  $cl_j$ . We now have  $S_j > S_{max}$  for this cluster from (4) above since  $C_j > C_{min}$ .

This means that the rule meets the confidence value on cluster  $cl_j$  and the only thing preventing it from appearing is the upper bound support threshold which was set to be  $S_{max}$ . This means that the rule will be discovered  $R$  will be discovered across at least one cluster under traditional (frequent) association rule mining.

In a real world setting we envisage that rare association rule mining will be done in conjunction with frequent rule mining and thus any rule discovered on the unclustered data set will be picked up by AICluster under association rule mining as a whole.

We now consider the reverse situation. Will all rules discovered by AICluster be also discovered by Apriori Inverse? This will not be the case as we have certain rules  $R'$  that apply on clusters that will not apply on the unclustered data set as rules such as  $R'$  will fail to meet the lower bound confidence threshold  $C_{min}$ . There are two cases to consider:

**Case 1:**The antecedent occurs in other clusters without the occurrence of the consequent thus lowering the confidence of the rule  $R'$  on the unclustered data set. This is the effect of the contamination issue that we referred to earlier in the paper.

**Case 2:**The antecedent does not occur in any other clusters. In this case it is possible that the Fisher test fails as the number of co-occurrences of the antecedent and consequent is a very small proportion of the total number of instances across the unclustered data set (which is much larger than the number of instances in the cluster where  $R'$  is true). Thus the Fisher test would flag these co-occurrences as chance collisions.

The cases 1 and 2 represent cases where rules on clusters do not manifest on the unclustered data set. We thus make the claim that AICluster outperforms Apriori Inverse with respect to rule coverage.  $\square$

In the next section, we present the results from our technique and compare our approach with the Apriori Inverse algorithm.

## 5 Experimental Results

In this section, we compare the performances of the standard Apriori Inverse algorithm with our proposed Apriori Inverse with Clustering (AICluster) algorithm. Testing of the algorithms was carried out on seven different datasets from the UCI Machine Learning Repository (Newman et al. 1998).

Table 1 represents the rules found using the AICluster and Apriori Inverse algorithms. For AICluster we set the maximum support threshold

Table 1: Results Based On AICluster and Apriori Inverse algorithm

Dataset	AICluster		Apriori Inverse					
	maxsup (0.30)		maxsup (0.10)		maxsup (0.20)		maxsup (0.30)	
	Rules	Avg Rule Support	Rules	Avg Rule Support	Rules	Avg Rule Support	Rules	Avg Rule Support
Zoo	2	4	1	8	10	11	72	13
Hepatitis	1	3	0	0	11	6	11	6
Flag	20	4	3	4	27	6	135	9
Heart	2	6	0	0	0	0	0	0
Soybean-Large	1862	6	166	7	6446	7	6975	8
Congressional Votes	3	3	0	0	0	0	0	0
Dataset	maxsup (0.01)		maxsup (0.05)		maxsup (0.10)		maxsup (0.15)	
	Rules	Avg Rule Support	Rules	Avg Rule Support	Rules	Avg Rule Support	Rules	Avg Rule Support
Mushroom	76179	4	10772	17	27505	17	39543	17

(maxsup) to 0.30 for all datasets except for the mushroom dataset. For six of the datasets, we ran Apriori Inverse at three different maxsup values of 0.10, 0.20, and 0.30. This was done in order to obtain a benchmark for comparison with AICluster on the all important rule support measure. In all of the experiments, we set the minimum threshold values for confidence and lift to 0.90 and 1.0 respectively.

We now compare the number of rules generated using AICluster with maxsup at 0.30 and Apriori Inverse with maxsup at 0.10. From Table 1 above we can clearly see that comparable levels of actual rule support occur at a maxsup of 0.3 for AICluster and 0.1 for Apriori Inverse. As we expected, the maxsup threshold for AICluster had to be set higher due to the fact that each cluster is smaller in size than the unclustered dataset. At these support thresholds we can see from Table 1 that the rule coverage for AICluster is consistently greater than that of Apriori Inverse for the first six datasets in Table 1 (with the exception of mushroom).

In the case of the mushroom dataset we lowered the maxsup thresholds for both algorithms in view of its relatively large size. We set maxsup to 0.01 for AICluster and then tested Apriori Inverse at lower values to compensate for the larger dataset size that it operates on, in keeping with our experimentation on the other six datasets that we experimented with. However, these settings caused Apriori Inverse to perform very poorly with respect to rule coverage and we thus decided to test it with maxsup values of 0.05, 0.1 and 0.15. Despite these favourable settings for Apriori Inverse, Table 1 shows that AICluster still significantly outperforms Apriori Inverse with respect to rule coverage.

Table 2 gives a more in-depth view of the rule bases covered by the two algorithms. The most striking feature is the very low degree of overlap between the two algorithms. The degree of overlap ranged from 0.73% (for mushroom) to 10% (for the much smaller Flag dataset) across the range of datasets tested. Coupled with the fact that AICluster on its own covers a very large percentage (93% and 88% for the two larger datasets, soybean and mushroom respectively) of the total rule base in the rare mining mode, this shows once again its superiority over Apriori Inverse in terms of rule coverage. As shown in section 4 above this percentage will rise to 100% when frequent association rule mining is done in conjunction with rare mining via AICluster.

In the next section we analyse the information content of the rules produced by these techniques.

## 5.1 Rule Analysis on Congressional Votes

The AICluster algorithm identified 4 rare rules from the set of clustered transactions in Congressional Votes dataset using a maximum support threshold of 0.3. Clustering on this dataset produced a total of 4 clusters. Two of the rules were from cluster 0 and the other 2 rules came from cluster 3.

### Cluster 0:

physician-fee-freeze:y → Class:republican, (Conf:1.00, Lift:31.0)

Class:republican → physician-fee-freeze:y, (Conf:1.00, Lift:31.0)

### Cluster 3:

anti-satellite-test-ban:y → export-administration-act-south-africa: y, (Conf:0.94, Lift:1.54)

physician-fee-freeze:n → Class:democrat, (Conf:1.00, Lift:16.83)

However the Apriori Inverse algorithm failed to produce any rare rules at the maximum support threshold that we set. This is due to the groupings contaminating each other and preventing candidate rules from meeting the minimum confidence threshold.

## 5.2 Rule Analysis on Zoo

Using the AICluster algorithm we were able to find 2 rules from the set of clustered transactions in the Zoo Dataset.

### Cluster 0:

fins:1 → aquatic:1 (Conf =1.00, Lift =6.33)

legs:0 → fins:1 (Conf =1.00, Lift =9.50)

These two rules are particularly interesting, as the class of animals in the cluster was Type 1 which was mammal. In this instance, we only had three transactions (seal, dolphins, porpoise) which have fins, are aquatic and are mammals. Finding these rules within this cluster is indeed interesting. In the case of Apriori Inverse we were able to detect only 1 rule with maxsup at 0.10.

### Apriori Inverse(maxsup =0.10)

type:6 → legs:6 (Conf:1.00, Lift:10.10)

Setting the maximum support at 0.30, we were able to find 2 rules using AICluster which were not belonging to “Type 6”. This was because the dataset was clustered in such a manner which allowed all homogeneous transactions to be clustered together. In each of the homogeneous clusters we were not able to find any rare rules. “Type 6” appeared 8 times within

Table 2: Summary of Rules from AICluster and Apriori Inverse

Dataset	No. of Rules Apriori Inverse	No. of Rules AICluster	No. of Rules Complete Set	No. of Rules Overlap
Zoo	1	2	3	0
Hepatitis	0	1	1	0
Flag	3	20	21	2
Heart	0	2	2	0
Soybean-large	166	1862	1993	35
Congressional Votes	0	3	3	0
Mushroom	10772	76179	86314	637

the original dataset which had 101 transactions. After transaction clustering we found that type 6 was clustered together in a particular cluster. The cluster consists of “Type 6” and 8 transactions. The support of “Type 6” in the cluster was 1.00 and in effect was no longer considered rare.

### 5.3 Rule Analysis on Heart Dataset

Using AICluster algorithm we were able to find 2 rules from the set of clustered transactions in the Heart Cleveland Dataset. The attribute *num* represents the diagnosis of heart disease and *ca* represents the number of major vessels (0-3) coloured by fluoroscopy. The attribute *thal* has three values, 3 for normal, 6 for fixed defect, and 7 for reversible defect.

#### Cluster 0:

num:3 → ca:2 (Conf =1.00, Lift =11.22)

#### Cluster 1:

num:0 → thal:normal (Conf =0.92, Lift =2.08)

We were able to find one rule with AICluster which is the rule found in Cluster 0 above. However we were not able to detect any rules with Apriori Inverse on the unpartitioned dataset due to contamination from other sub groupings.

## 6 Conclusion

Our approach first clustered transactions into homogeneous clusters and then generated rare rules from each of the clusters formed. These rules expressed their own associations without interference or contamination from other sub groupings that have different patterns of relationships. We were able to demonstrate that the clustering process added value to the rare association rule mining process, generating interesting rules that could not be found otherwise.

One possible direction for future work would be to weigh itemsets according to their support, with lower frequency itemsets given a higher weight over their higher frequency counterparts. This would lead to the generation of rare rules that have lower frequency in each of their individual terms than would be possible with the current version of AICluster.

## References

Agrawal, R., Imielinski, T. & Swami, A. (1993), Mining association rules between sets of items in large databases, in P. Buneman & S. Jajodia, eds, ‘Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data’, pp. 207 – 216.

Cutting, D. R., Pedersen, J. O., Karger, D. & Tukey, J. W. (1992), Scatter/gather: A cluster-based approach to browsing large document collections, in ‘Proceedings of the Fifteenth Annual International

ACM SIGIR Conference on Research and Development in Information Retrieval’, pp. 318–329.

Ganti, V., Gehrke, J. & Ramakrishnan, R. (1999), CACTUS: Clustering categorical data using summaries, in ‘KDD ’99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining’, ACM Press, New York, NY, USA, pp. 73–83.

Guha, S., Rastogi, R. & Shim, K. (2000), ‘ROCK: A robust clustering algorithm for categorical attributes’, *Information Systems* **25**(5), 345–366.

Jain, A. K., Murty, M. N. & Flynn, P. J. (1999), ‘Data clustering: a review’, *ACM Comput. Surv.* **31**(3), 264–323.

Koh, Y. S. & Pears, R. (2008), Transaction clustering using a seeds based approach, in T. Washio, E. Suzuki, K. M. Ting & A. Inokuchi, eds, ‘PAKDD’, Vol. 5012 of *Lecture Notes in Computer Science*, Springer, pp. 916–922.

Koh, Y. S. & Rountree, N. (2005), Finding sporadic rules using Apriori Inverse, in T. B. Ho, D. W.-L. Cheung & H. Liu, eds, ‘PAKDD’, Vol. 3518 of *Lecture Notes in Computer Science*, Springer, pp. 97–106.

Koh, Y. S., Rountree, N. & O’Keefe, R. A. (2008), ‘Mining interesting imperfectly sporadic rules’, *Knowl. Inf. Syst.* **14**(2), 179–196.

Liu, B., Hsu, W. & Ma, Y. (1999), Mining association rules with multiple minimum supports, in ‘Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining’, pp. 337 – 341.

Newman, D., Hettich, S., Blake, C. & Merz, C. (1998), ‘UCI repository of machine learning databases’, <http://www.ics.uci.edu/~mllearn/MLRepository.html>.

Plasse, M., Niang, N., Saporta, G., Villeminot, A. & Leblond, L. (2007), ‘Combined use of association rules mining and clustering methods to find relevant links between binary rare attributes in a large data set’, *Computational Statistics & Data Analysis* **52**(1), 596–613.

Wang, K., Xu, C. & Liu, B. (1999), Clustering transactions using large items, in ‘CIKM ’99: Proceedings of the Eighth International Conference on Information and Knowledge Management’, ACM Press, New York, NY, USA, pp. 483–490.

Xu, J., Xiong, H., Sung, S. Y. & Kumar, V. (2003), A new clustering algorithm for transaction data via caucus, in ‘Advances in Knowledge Discovery and Data Mining: 7th Pacific-Asia Conference, PAKDD 2003, Seoul, Korea, April 30 - May 2, 2003. Proceedings’, pp. 551–562.

Yun, C.-H., Chuang, K.-T. & Chen, M.-S. (2001), An efficient clustering algorithm for market basket data based on small large ratios, *in* 'COMPSAC '01: Proceedings of the 25th International Computer Software and Applications Conference on Invigorating Software Development', IEEE Computer Society, Washington, DC, USA, pp. 505–510.

Yun, H., Ha, D., Hwang, B. & Ryu, K. H. (2003), 'Mining association rules on significant rare data using relative support', *The Journal of Systems and Software* **67**(3), 181 – 191.