A short version of this paper appears in Workshop on Research Issues on Data Mining and Knowledge Discovery 1997

Clustering Based On Association Rule Hypergraphs *

Eui-Hong (Sam) Han George Karypis Vipin Kumar

Bamshad Mobasher

Department of Computer Science
University of Minnesota
4-192 EECS Bldg., 200 Union St. SE
Minneapolis, MN 55455, USA

{han,karypis,kumar,mobasher}@cs.umn.edu

Technical Report 97-019 (June 18, 1997)

Abstract

Traditional clustering algorithms, used in data mining for transactional databases, are mainly concerned with grouping transactions, but they do not generally provide an adequate mechanism for grouping items found within these transactions. *Item clustering*, on the other hand, can be useful in many data mining applications. We propose a new method for clustering related items in transactional databases that is based on partitioning an association rule hypergraph, where each association rule defines a hyperedge. We also discuss some of the applications of item clustering, such as the discovery of meta-rules among item clusters, and clustering of transactions. We evaluated our scheme experimentally on data from a number of domains, and, wherever applicable, compared it with *AutoClass*. In our experiment with stock-market data, our clustering scheme is able to successfully group stocks that belong to the same industry group. In the experiment with congressional voting data, this method is quite effective in finding clusters of transactions that correspond to either democrat or republican voting patterns. We found clusters of segments of protein-coding sequences from protein coding database that share the same functionality and thus are very valuable to biologist for determining functionality of new proteins. We also found clusters of related words in documents retrieved from the World Wide Web (a common and important application in information retrieval). These experiments demonstrate that our approach holds promise in a wide range of domains, and is much faster than traditional clustering algorithms such as *AutoClass*.

Keywords: Data mining, clustering, association rules, hypergraph partitioning.

*This work was supported by NSF grant ASC-9634719, Army Research Office contract DA/DAAH04-95-1-0538, Cray Research Inc. Fellowship, and IBM Partnership Award, the content of which does not necessarily reflect the policy of the government, and no official endorsement should be inferred. Access to computing facilities was provided by AHPCRC, Minnesota Supercomputer Institute, Cray Research Inc., and NSF grant CDA-9414015. See http://www.cs.umn.edu/~han for other related papers.

1 Introduction

Clustering in data mining [SAD+93, CHY96, GHK+96] is a discovery process that groups a set of data such that the intracluster similarity is maximized and the intercluster similarity is minimized [CHY96]. These discovered clusters are used to explain the characteristics of the data distribution. For example, in many business applications, clustering can be used to characterize different customer groups and allow businesses to offer customized solutions, or to predict customer buying patterns based on the profiles of the cluster to which they belong.

In transaction databases, attributes generally represent items that could potentially occur in one transaction. If I is the number of different items, then each transaction can be represented by a point in an I-dimensional space. Traditional clustering techniques [NH94, CS96, SD90, Fis95, DJ80, Lee81] focus mainly on grouping together such transactions based on some measure of similarity or distance. These techniques, however, do not generally provide an adequate mechanism for grouping related items that appear within transactions. In this paper we are mainly concerned with clustering related items and the application of *item clustering* to various data mining domains.

Given a database of transactions, there are two areas where clustering related items may be useful. First, item clusters themselves could lead to the discovery of important hidden knowledge within the database. Such clusters could be used in some domains to classify data items, to make predictions about similar data items, or to reduce the size of rule sets by eliminating those that are not interesting. Consider for instance the items that are sold in a grocery store. If we can cluster these items into item-groups that are often sold together, we can then use this knowledge to perform effective shelf-space organization as well as target sales promotions.

Furthermore, once the item clusters have been found, an analyst can use each item cluster as a unique item and discover relationships (such as association rules) among the item clusters. For example, given a database of stock market data, where each row would represent one day of activity for the available stocks, item clustering can be used to discover industry groups containing related stocks. Then, by finding association rules among these item clusters, one can discover various relationships between the industry groups in terms of their price or volume movements (see Section 5).

The second area where item clustering can play an important role is transaction clustering. As noted above, transaction clustering has been the purview of traditional clustering algorithms, and has been used for knowledge discovery in a variety of domains. Most of these clustering algorithms are able to effectively cluster transactions only when the dimensionality of the space (*i.e.*, the number of different items) is relatively small and most of the items are present in each transaction [DJ80, SD90, NH94]. However, these schemes fail to produce meaningful clusters, if the number of items is large and/or the fraction of the items present in each transaction is small. This type of data-sets are quite common in many data mining domains (*e.g.*, market basket analysis). For example, a typical grocery store sells thousands of different items but each customer buys only a small number of them (usually less than thirty).

Item clustering can be used to improve the performance of traditional clustering algorithms by reducing the dimensionality of the clustering problem. Once item clusters are found, they can be used as (a much smaller set of) attributes in a new *meta-transaction* database, thus making the application of clustering algorithms more efficient. There are also situations where item clusters naturally induce cluster of transactions.

Another way that item clustering can be used to cluster transactions is by inverting that transaction database so that each transaction ID is treated as an attribute, and then performing item clustering on these new attributes, thus effectively partitioning the transaction space.

In this paper, we propose a new methodology for clustering related items in transactions using association rules and hypergraph partitioning. Association rule discover in data mining has been used to discover relationships in very large data repositories [AMS$^+$96, HS95]. Here we explore the feasibility and advantages of using the discovered association rules to cluster closely related items into groups. Our algorithm for item clustering uses as its basis the frequent item sets, derived as part of association rule discovery, that meet a minimum support criterion [AMS$^+$96]. These frequent item sets are then used to group items into hypergraph edges, and a hypergraph partitioning algorithm [KAKS97] is used to find the item clusters. We also explore the use of discovered item clusters to cluster related transactions containing the data items.

We evaluated our item clustering schemes on five different data sets, namely S&P500 stock data, US congressional voting data, protein coding data, Web document data and US census data. Wherever applicable, we compared our results with the those of *AutoClass* [CS96]. These experiments demonstrate that our approach is applicable in a wide range of domains and provide better clusters and is much faster than *AutoClass*. We chose *AutoClass* for comparison because *AutoClass* can handle data with the mixture of continuous and discrete attributes, where distance or similarity measures are difficult to define, and is known for producing quality clusters.

The rest of this paper is organized as follows. Section 2 contains related work. Section 3 presents clustering of items and Section 4 discusses applications of the item clusters . Section 5 presents the experimental results. Section 6 contains conclusion and future works.

2 Related Work

Clustering of transactions can be done using methods that have been studied in several areas including statistics [DJ80, Lee81, CS96], machine learning [SD90, Fis95], and data mining [NH94, CS96]. Most of the these approaches are based on either probability, distance or similarity measure. Distance-based methods such as k-means method [JD88], nearest-neighbor method [LF78] and CLARANS algorithm [NH94] are effective when the distance between two data points can be easily defined. Transaction data, however, often contains a mixture of attributes, and so the distance between data points in the multidimensional space of attributes may be very hard to define, thus making distance-based methods inapplicable. If the data contains only discrete attributes or continuous attributes that can be discretized, the similarity measure can be obtained and translated into a distance matrix with multidimensional scaling methods. However, these multidimensional scaling methods have high computational complexity of $O(n^2)$ where n is the number of transactions [JD88]. Probability based methods do not require distance measure and are more applicable in data mining domain where attribute space has mixture of discrete and continuous attributes. *AutoClass* [CS96] is based on the probabilistic mixture modeling [TSM85] and handles the mixture of attributes. However, the underlying expectation-maximization (EM) algorithm [TSM85] has the computational complexity of $O(kd^2nI)$ where k is the number of clusters, d is the number of attributes, n is the number of transactions and I is the average number of iterations of the EM algorithm. In data mining domain, where the n and d is large, *AutoClass*' run time can be unacceptably high.

The use of hypergraphs in data mining has been studied recently. For example [GKMT97] shows that the problem of finding maximal elements in a lattice of patterns is closely related to the hypergraph transversal problem [EG95] and explore the use of known algorithms for finding maximal elements to solve the hypergraph transversal problem.

The clustering or grouping of association rules have been proposed in [TKR+95], [LSW97] and [KA96]. [TKR+95] proposed clustering of association rules that have the same right-hand side to structure association rules for better understanding. The clustering is accomplished by defining a distance between association rules (the number of rows where the two rules differ). [LSW97] proposed a heuristic approach to clustering two attribute association rules, based on the geometric properties of the two-dimensional grid. Association rules of the form $A \wedge B \Rightarrow C$, where A and B are continuous attributes and C is categorical, are clustered in this approach. Segmentation of attributes values of A and B with a specific value of C is obtained. The algorithm does not handle the discrete attributes on the left-hand side of the rules. In both of these works, the focus is on finding clusters of association rules that have the same right hand side rather than on finding item clusters. [KA96] also proposed to cluster database attributes based on binary associations. In this approach, the association graph is constructed by taking items in a database as vertex set and considering binary associations among the items as edges in the graph. The minimum spanning tree is constructed and the remaining edges in the minimum spanning tree are proposed as the most interesting associations. It is suggested that successive removal of edges from the minimum spanning tree would produce clusters of attributes, but it is noted that a good criterion for continuing the clustering process is hard to define [KA96].

3 Clustering of Items

Our algorithm for clustering related items consists of the following two steps. During the first step, it constructs a weighted hypergraph H to represent the relations among different items, and during the second step, it uses a hypergraph partitioning algorithm to partition this hypergraph into k partitions such that the items in each partition are highly related. Each one of these partitions will become an item cluster. We first present a brief overview of association rules that are used to model the information in a transaction database as a hypergraph, and then describe the hypergraph modeling and the clustering algorithm.

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Table 1: Transactions from supermarket.

3.1 Association Rules

Association rules capture the relationship of items that are present in a transaction [AMS+96]. Let T be the set of transactions where each transaction is a subset of the item-set I. Let C be a subset of I, then we define the *support count* of C with respect to T to be:

$$\sigma(C) = |\{t|t \in T, C \subseteq t\}|.$$

Thus $\sigma(C)$ is the number of transactions that contain C. For example, consider a set of transactions from supermarket as shown in Table 1. The items set I for these transactions is $\{$Bread, Beer, Coke, Diaper, Milk$\}$. The support count of $\{$Diaper, Milk$\}$ is $\sigma(Diaper, Milk) = 3$, whereas $\sigma(Diaper, Milk, Beer) = 2$.

An *association rule* is an expression of the form $X \overset{s,\alpha}{\Longrightarrow} Y$, where $X \subseteq I$ and $Y \subseteq I$. The *support* s of the rule $X \overset{s,\alpha}{\Longrightarrow} Y$ is defined as $\sigma(X \cup Y)/|T|$, and the confidence α is defined as $\sigma(X \cup Y)/\sigma(X)$. For example, consider a rule $\{$Diaper, Milk$\} \Longrightarrow \{$Beer$\}$, i.e. presence of diaper and milk in a transaction tends to indicate the presence of beer in the transaction. The support of this rule is $\frac{\sigma(Diaper, Milk, Beer)}{5} = 0.40$. The confidence of this rule is $\frac{\sigma(Diaper, Milk, Beer)}{\sigma(Diaper, Milk)} = 0.66$. Rules with high confidence (i.e., close to 1.0) are important, because they denote a strong correlation of the items in the rule. Rules with high support are important, since they are supported by a non-trivial fraction of transactions in the database.

The task of discovering an association rule is to find all rules $X \overset{s,\alpha}{\Longrightarrow} Y$, such that s is greater than a given minimum support threshold and α is greater than a given minimum confidence threshold. The association rule discovery is composed of two steps. The first step is to discover all the frequent item-sets (candidate sets that have more support than the minimum support threshold specified). The second step is to generate association rules from these frequent item-sets. The computation of finding the frequent item-sets is usually much more expensive than finding the rules from these frequent item-sets.

A number of algorithms have been developed for discovering frequent item-sets [AIS93, AS94, HS95]. *Apriori* algorithm presented in [AS94] is one of the most efficient algorithms available. This algorithm has been experimentally shown to be linearly scalable with respect to the size of the database [AS94], and can also be implemented on parallel computers [HKK97] to use their large memory and processing power effectively.

3.2 Hypergraph Modeling

A hypergraph [Ber76] $H = (V, E)$ consists of a set of vertices (V) and a set of hyperedges (E). A hypergraph is an extension of a graph in the sense that each hyperedge can connect more than two vertices. In our model, the set of vertices V corresponds to the item set I in the transaction database, and each hyperedge $e \in E$ corresponds to a set of related items. A key problem in modeling of transaction data as hypergraph is the determination of related items that can be grouped as hyperedges and determining weights of each such hyperedge.

The frequent item sets computed by an association rule algorithm such as Apriori are excellent candidate to find such related items. Note that these algorithm only find frequent item sets that have support greater than a specified threshold. The value of this threshold may have to be determined in a domain specific manner.

Assignment of weight to the resulting hyperedges is more tricky. One obvious possibility is to use the support of each frequent item set as the weight of the corresponding hyperedge. This has the disadvantage that the weight of larger hyperedges will generally be much more smaller than the weight of smaller hyperedges. Another, more natural, possibility is to define a weight as a function of the support and confidence of the rules that are made of a group of items in a frequent item set. In our current implementation of the model, each frequent item-set is represented by a hyperedge $e \in E$ whose weight is equal to the average confidence of all the association rules involving the items in the item-set. For example, if {A,B,C} is a frequent item-set, then the hypergraph contains a hyperedge that connects A, B, and C. If $\{A\} \xRightarrow{0.8} \{B,C\}$, $\{A,B\} \xRightarrow{0.4} \{C\}$, $\{A,C\} \xRightarrow{0.6} \{B\}$, $\{B\} \xRightarrow{0.4} \{A,C\}$, $\{B,C\} \xRightarrow{0.8} \{A\}$, and $\{C\} \xRightarrow{0.6} \{A,B\}$ are all the possible association rules with confidence noted and the weighting function is the average of the confidences, then the weight of the hyperedge connecting A,B, and C is 0.6. We will refer to this hypergraph as the ***association-rule hypergraph***.

3.3 Finding Clusters of Items

Note that the frequent items sets already represent relationship among the items of a transaction. But these relationships are "fine grain". For example, consider the following three frequent item sets found from a database of stock transactions:

{Texas Inst↑, Intel↑, Micron Tech↑}

{National Semiconduct↑, Intel↑}

{National Semiconduct↑,Micron Tech↑}

These item sets indicate that on many different days, stocks of Texas Instrument, Intel and Micron Technology moved up together, and on many days, stocks of Intel and National Semiconductor moved up together, etc. From these, it appears that Texas Instrument, Intel, Micron Technology and National Semiconduct or are somehow related. But a frequent item set of these four stocks may have small support and my not be captured by the association rule computation algorithm.

However the hypergraph representation can be used to cluster together relatively large groups of related items by partitioning it into highly connected partitions. One way of achieving this is to use a hypergraph partitioning algorithm that partitions the hypergraph into two parts such that the weight of the hyperedges that are cut by the partitioning is minimized. Note that by minimizing the hyperedge-cut we essentially minimize the relations that are violated by splitting the items into two groups. Now each of these two parts can be further bisected recursively, until each partition is highly connected. One way to define a fitness function is to use the connectivity of the vertices in each partition to determine whether a partition is highly connected or it needs to be subdivided further. If each vertex belongs to a large fraction of the hyperedges in its partition, then we can stop the recursive subdivision.

The method as described, works for transactions with binary attributes (either an item is present or is absent from the transactions). For discrete attributes with multiple values, each value of the attribute is considered to be a separate item. Continuous attributes can be handled once they have been discretized using techniques proposed in [SA96].

3.4 Computational Complexity

There are two distinct phases in our item-clustering algorithm, namely finding the association rules and partitioning the association-rule hypergraph.

The problem of finding association rules has been shown to be linearly scalable with respect to the number of transactions [AMS$^+$96]. Very fast and highly efficient algorithms such as Apriori are able to quickly find association rules in very large databases. For example, our experiments with Apriori have shown that the rules in more than 1 million transactions can be found in less than hundreds of seconds. Thus computing these association rules is not a bottleneck. Furthermore, many times these rules will be already available as a result of an earlier analysis.

Hypergraph partitioning is a well studied problem and highly efficient algorithms such as HMETIS have been developed [KAKS97]. In particular, the complexity of HMETIS for a k-way partitioning is $O((V + E)\log k)$ where V is the number of vertices and E is the number of edges. The number of vertices in an association-rules hypergraph is the same as the number of distinct items in the database which is fixed for a given database. The number of hyperedges is the same as the number of frequent item-sets with support greater than the minimum support. Note that the number of frequent item sets (i.e., hyperedges) does not increase as the number of transactions increases. Furthermore, we can control the number of hyperedges by changing the minimum support and/or lower limit on the weights of the hyperedges. In particular, we can raise the minimum support to decrease the number of frequent item-sets and we can include hyperedges in the hypergraph only if they have at least a certain minimum weight. Hence, our clustering method is linearly scalable with respect to the number of transactions.

4 Applications of Item Clusters

Once item clusters have been discovered, a new *meta-transaction* database can be induced using the original transaction database and the item clusters. The new database has item clusters as attributes, and attribute

Item Cluster	{A, B, C}
Support	A: 0.2, B:0.5, C:0.1
Utility	A: 5 , B:2, C:10
SU	17

	item matching	item matching with utility
$T_1 = \{A,C\}$	$\frac{2}{3} = 0.67$	$\frac{5+10}{17} = 0.88$
$T_2 = \{A,B\}$	$\frac{2}{3} = 0.67$	$\frac{5+2}{17} = 0.41$

Table 2: Examples of matching transactions with an item cluster.

values are *matching* scores reflecting the affinity between transactions and item clusters. Table 3 depicts the general form of the meta-transaction database.

One simple scoring function based on item matching is defined as the fraction of the items in the cluster that are present in the transactions; i.e., the matching score of a transaction T_i on an item cluster IC_j is:

$$m_{i,j} = \frac{|T_i \bigcap IC_j|}{|IC_j|}$$

For example, transactions T_1 and T_2 of Table 2 match 2 out of 3 items from the item cluster and thus have 0.67 as matching value. One variation of this matching scoring function uses support information of each item in the cluster. In this matching scheme, each item has a utility value that is reciprocal to the support of the item. Let $SU(I) = \sum_{i \in I} utility(i)$, i.e., $SU(I)$ is the sum of all the utility values of items in the set I. Then the matching score of a transaction T_i on an item cluster IC_j is:

$$m_{i,j} = \frac{SU(T_i \bigcap IC_j)}{SU(IC_j)}$$

For example, transaction T_1 of Table 2 has matching value 0.88 and transaction T_2 has 0.41. With this matching scheme, two transactions have different matching values whereas they have the same matching value in the simple item matching scheme. This matching function favors clustering transactions that match attribute values that occur less frequently.

However, the matching scheme based on item matching is biased with respect to the size of the item clusters. Transactions tend to match smaller item clusters more. Furthermore, this scoring function does not take into consideration the association rules of the item clusters. We are currently considering other matching schemes based on the association rules within the item clusters.

Clustering of transactions After meta transactions have been defined (Table 3), based on the matching scoring functions, there are several options for clustering original transactions. The first option is to assign

	IC_1	IC_2	IC_3	IC_4	$\cdots$	IC_k
T_1	$m_{1,1}$	$m_{1,2}$	$m_{1,3}$	$m_{1,4}$	$\cdots$	$m_{1,k}$
T_2	$m_{2,1}$	$m_{2,2}$	$m_{2,3}$	$m_{2,4}$	$\cdots$	$m_{2,k}$
T_3	$m_{3,1}$	$m_{3,2}$	$m_{3,3}$	$m_{3,4}$	$\cdots$	$m_{3,k}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
T_n	$m_{n,1}$	$m_{n,2}$	$m_{n,3}$	$m_{n,4}$	$\cdots$	$m_{n,k}$

Table 3: Meta transactions with k item clusters.

each transaction to the item cluster with the best matching value to define a cluster:

$$C_j = \{T_i | j = \max_{1 \le l \le k} m_{i,l}\}$$

The second option is fuzzy clustering where each transaction is not assigned to a single cluster but assigned to multiple clusters with the degree of confidence as matching values. For instance, for a transaction T_i, we say that it belongs to C_j with confidence of $m_{i,j}$.

Dimensionality Reduction Yet another option is to use Table 3 as a new data and use any distance or probability based clustering algorithms. There are a couple of advantages of this approach. As noted earlier, distance-based clustering methods can not handle the mixture of continuous and discrete attribute values well. However, the meta transactions consist of continuous attributes only and thus the attribute characteristics are uniform and normalization is straightforward. Another advantage of the meta transactions is that the attribute space has been reduced dramatically. In meta transactions the number of attributes is the number of item clusters which is usually substantially smaller than the original number of attributes. The reduced dimensionality provides two advantages. First is that it makes the problem space more dense. Consider a supermarket transaction database where the number of attributes correspond to the number of items sold in a grocery store. When the transactions are put into the attribute space defined by this database, the space is very sparse and none of the existing clustering algorithms can find the clusters very well. The meta transactions has much smaller number of attributes and the same number of transactions. Thus, the attribute space is more densely populated with the transactions. This gives a better hope of finding clusters with existing cluster algorithms. Second advantage of reduced dimensionality is the computational efficiency. All the existing clustering program will benefit because most of them have $O(dn)$-like complexity. Especially for *AutoClass* which has $O(d^2n)$ complexity, the smaller d will improve the computational efficiency immensely.

Meta Rules An additional advantage of meta transactions defined in the form of Table 3 is that they provide a way to find meta-level association rules among different item clusters. We discretize the attributes in Table 3 and find association rules from these meta transactions. The results are association rules among different item clusters. Generalized or multi-level association rule discovery algorithms [SA95, HF95] require the hierarchy to be provided explicitly from the user. However, in many real life domains, the hierarchy of items may be unavailable or hard to define. Item clusters provide a structure that is discovered from the transactions without any user provided information. The association rules among these item clusters provide useful high-level knowledge.

5 Experimental Results

We tested the ability of our item-clustering algorithm to correctly find groups of related items, on data-sets arising in many application areas including stock market, voting records, biology, and census. The results of these experiments, as well as the applications of these item-clusters on each data-set are described in the following subsections.

We have also compared our results with that of *AutoClass* whenever it is possible. *AutoClass* cannot be used directly to cluster the items; however, we can transpose the transactions in our data set, such that each item now becomes a transaction, and each transaction becomes an item. We chose *AutoClass* for comparison because *AutoClass* can handle data with the mixture of continuous and discrete attributes, where distance or similarity measures are difficult to define, and is known for producing good quality clusters.

In all of our experiments, we used a locally implemented version of *Apriori* algorithm [AMS$^+$96] to find the association rules and construct the association-rule hypergraph. We used the average of confidences of the association rules as a weight for the corresponding hyperedges. For partitioning the association-rule hypergraph, we used HMETIS [KAKS97]. HMETIS is a multilevel partitioning algorithm that has been shown to quickly produce partitions that minimize the sum of the weight of the hyperedges that straddle partitions (*i.e.*, minimize the hyperedge cut). HMETIS produces k-way partitions, where k (i.e., the number of partitions) is specified by the user. The partitions produced by HMETIS are subject to balance constraints that restrict the relative size of the various partitions. One of the limitations of HMETIS is that since the number of partitions is fixed and it is forced to find partitions that are subject to balanced constraints, it might bisect a highly connected partition. As a result, some of the partitions do not satisfy the fitness function described in Section 3. For this reason, after getting the partitions produced by HMETIS, we select as item-clusters only those partitions that meet the fitness criterion Section 3. We used the simple scoring function described in Section 4 for matching transactions against item clusters to construct meta transactions.

5.1 S&P 500 Stock Data

Our first data-set consists of the daily activity of the stocks that belong to the S&P500 index. It is well known in the financial markets, that during many days stocks belonging to the same industry group tend to trade similarly. For example, a group of financial stocks tend to move up or down together depending on the market's belief about the health of this particular industry group. For this reason, we used this data set to verify the ability of our item-clustering algorithm to correctly cluster the various stocks according to their industry group. In particular, one transaction in this experiment is one day's S&P500 stock price movements compared to the previous day's closing price. The items in a transaction are company names with either up or down indicator. Thus, there is a total of 1000 distinct items (two for each company) and each transaction can have up to 500 of these items. Note that if in a particular day the price of a stock did not change relative to the previous day, then there is no item associated with this stock in that day. Given this transaction definition, an item-cluster will be a group of stocks and their relative movement. For example, an item-cluster may contain some stocks going up and some stocks going down.

Our data set consisted of a total of 716 transactions dating from Jan. 1994 to Oct. 1996. We used these transactions to find the association rules and construct the association-rule hypergraph. In particular, we found the association rules with a support of at least 3% which lead to a hypergraph consisting of 440 vertices and 19602 hyperedges. Note that the number of vertices is considerably smaller than the number of distinct items in the data-set. This is because some of the stocks do not move very frequently, hence the corresponding items do not have sufficient support. This hypergraph was then partitioned into 40 partitions. Out of these 40 partitions, only 20 of them satisfy the fitness function. These item-clusters and the industry groups associated with the clusters are shown in Table 4. Looking at these 20 clusters we can see that our item-clustering algorithm was very successful in grouping together stocks that belong to the same industry group. Out of the 20 clusters shown in Table 4, the first 16 clusters contain companies belonging to a single industry group. For example, our algorithm was able to find technology-, bank-, financial-, and, oil-related stock-clusters. The remaining four clusters contain companies that belong to two or three different industry groups. Also, it is interesting to see that our clustering algorithm partitioned the technology companies into two groups, one consisting mostly of networking and semiconductor companies (item-cluster one and seven). Also note that item-cluster 17 contains stocks that are rail- and oil-related that move in the opposite direction. That is, it contains rail-related stocks that move up and oil-related stocks that move down. Even though this is not an industry group, this cluster corresponds to related moves of the underlying securities (when oil prices go down, oil-related stocks go down and rail-related stocks go up, since their profit margins improve).

We also used *AutoClass* to find item-clusters. Specifically, each transaction in the transposed data set corresponds to an S&P500 stock, and each trading day is an attribute. Each attribute has three possible values: stock moved up, down or remain unchanged on that day. We used *AutoClass* on this transposed data-set to cluster the items. *AutoClass* found only three clusters of stocks. Two of these clusters contain a lot of stocks. One has 299 and the other has 137, and none of them contains stocks predominantly from any industry group or even a small number of different industry groups. The only interesting item-cluster was the third one which contains 26 stocks, all of them corresponding to technology-related companies. From this experiment we see that *AutoClass* is quite ineffective in finding good item-clusters. This is because as we transpose the transactions we dramatically increase the dimensionality of the problem (now it is equal to the number of days). Besides producing poor item-clusters, *AutoClass* also requires much more time than our item-clustering algorithm. It took *AutoClass* 445 seconds to produce the three item-clusters, whereas our algorithm was able to find 20 item-clusters in 121 seconds, which includes the time required to find the association rules, as well as the time to construct and partition the association-rule hypergraph.

	Discovered Item Clusters	Industry Group
1	APPLIED MATL↓, BAY NETWORK↓, 3 COM↓, CABLETRON SYS↓, CISCO↓, DSC COMM↓, HP↓, INTEL↓, LSI LOGIC↓, MICRON TECH↓, NATL SEMICONDUCT↓, ORACLE↓, SGI↓, SUN↓, TELLABS INC↓, TEXAS INST↓	Technology 1↓
2	APPLE COMP↓, AUTODESK↓, ADV MICRO DEVICE↓, ANDREW CORP↓, COMPUTER ASSOC↓, CIRC CITY STORES↓, COMPAQ↓, DEC↓, EMC CORP↓, GEN INSTRUMENT↓, MOTOROLA↓, MICROSOFT↓, SCIENTIFIC ATL↓	Technology 2↓
3	FANNIE MAE↓, FED HOME LOAN↓, MBNA CORP↓, MORGAN STANLEY↓	Financial↓
4	BAKER HUGHES↑, DRESSER INDS↑, HALLIBURTON HLD↑, LOUISIANA LAND↑, PHILLIPS PETRO↑, SCHLUMBERGER↑, UNOCAL↑	Oil↑
5	BARRICK GOLD↑, ECHO BAY MINES↑, HOMESTAKE MINING↑, NEWMONT MINING↑, PLACER DOME INC↑	Gold↑
6	ALCAN ALUMINUM↓, ASARCO INC↓, CYPRUS AMAX MIN↓, INLAND STEEL INC↓, INCO LTD↓, NUCOR CORP↓, PRAXAIR INC↓, REYNOLDS METALS↓, STONE CONTAINER↓, USX US STEEL↓	Metal↓
7	APPLIED MATL↑, BAY NETWORK↑, 3 COM↑, CABLETRON SYS↑, CISCO↑, COMPAQ↑, HP↑, INTEL↑, LSI LOGIC↑, MICRON TECH↑, NATL SEMICONDUCT↑, ORACLE↑, MOTOROLA↑, SUN↑, TELLABS INC↑, TEXAS INST↑	Technology↑
8	AUTODESK↑, DSC COMM↑, DEC↑, EMC CORP↑, COMPUTER ASSOC↑, GEN INSTRUMENT↑, MICROSOFT↑, SCIENTIFIC ATL↑, SGI↑, MERRILL LYNCH↑	Technology↑
9	BOISE CASCADE↑, CHAMPION INTL↑, GEORGIA-PACIFIC↑, INTL PAPER↑, JAMES RIVER CORP↑, LOUISIANA PACIFIC↑, STONE CONTAINER↑, TEMPLE INLAND↑, UNION CAMP CORP↑, WEYERHAEUSER↑	Paper/Lumber↑
10	AMERITECH CP↑, BELL ATLANTIC↑, NYNEX CORP↑	Regional Bell↑
11	BARRICK GOLD↓, ECHO BAY MINES↓, HOMESTAKE MINING↓, NEWMONT MINING↓, PLACER DOME INC↓	Gold↓
12	BANK BOSTON↑, CITICORP↑, CHASE MANHAT NEW↑, GREAT WEST FIN↑, BANC ONE CORP↑	Bank↑
13	ALCAN ALUMINUM↑, ASARCO INC↑, CYPRUS AMAX MIN↑, INLAND STEEL INC↑, INCO LTD↑, NUCOR CORP↑, PHELPS DODGE↑, REYNOLDS METALS↑, USX US STEEL↑, ALUM CO AMERICA↑, UNION CARBIDE↑	Metal↑
14	BRWNG FERRIS↑, CHRYSLER CORP↑, CATERPILLAR INC↑, CNF TRANS↑, DEERE & CO↑, FORD MOTOR CO↑, FOSTER WHEELER↑, GENERAL MOTORS↑, INGERSOLL-RAND↑	Motor/Machinery↑
15	CIRC CITY STORES↑, DILLARD↑, DAYTON HUDSON↑, FED DEPT STR↑, GAP INC↑, J C PENNEY CO↑, NORDSTROM ↑, SEARS ROEBUCK↑, TJX CO INC↑, WAL-MART STORES↑	Retail/Department Stores↑
16	APPLE COMP↑, ADV MICRO DEVICE↑, AMGEN↑, ANDREW CORP↑, BOSTON SCIEN CP↑, HARRAHS ENTER↑, HUMANA INC INC↑, SHARED MED SYS↑, TEKTRONIX↑, UNITED HEALTH CP↑, US SURGICAL↑	Technology/Electronics↑
17	BURL NTHN SANTA↑, CSX CORP↑, HALLIBURTON HLD↓, HELMERICH PAYNE↓	Rail↑/Oil↓
18	AMR CORP↑, COLUMBIA/HCA↑, COMPUTER SCIENCE↑, DELTA AIR LINES↑, GREEN TREE FIN↑, DISCOVER↑, HOME DEPOT INC↑, MBNA CORP↑, LOWES COMPANIES↑, SW AIRLINES↑, MORTON INTL↑, PEP BOYS↑	Air/Financial/Retail↑
19	CHRYSLER CORP↓, FLEETWOOD ENTR↓, GENERAL MOTORS↓, HUMANA INC↓, INGERSOLL-RAND↓, LOUISIANA PACIF↓, MERRILL LYNCH↓, NOVELL INC↓, PACCAR INC↓, TEKTRONIX↓	Auto/Technology↓
20	CNF TRANS↓, CENTEX CORP↓, GAP INC↓, HARRAHS ENTER↓, LOWES COMPANIES↓, SW AIRLINES↓, MCI COMMS CP↓, SEARS ROEBUCK↓, SHARED MED SYS↓, TELE COMM INC↓, UNITED HEALTH CP↓, US SURGICAL↓	Home Building/Retail/Technology↓

Table 4: Clustering of S&P 500 Stock Data

Rule No.	Confidence(%)	Meta Association Rules
1	75.0	{Technology 2↓} $\Longrightarrow$ {Technology 1↓}
2	100.0	{Technology 2↓, Metal↓} $\Longrightarrow$ {Technology 1↓}
3	80.0	{Gold↑, Technology 2↓} $\Longrightarrow$ {Financial↓}
4	81.0	{Rail↑/Oil↓} $\Longrightarrow$ {Motor/Machinery↑}

Table 5: Some Examples of the Meta Association Rules Discovered

We applied the method for computing meta association rules described in Section 4 on the item clusters found from the S&P 500 data. Some of the rules are shown in Table 5 with their confidence. Rule 1 shows that 75% of time when the Technology cluster 2 went down, the Technology cluster 1 went down. Rule 2, on the other hand, shows that every time when the Technology cluster 2 and the Metal cluster went down, the Technology cluster 1 went down. These two meta association rules demonstrate the usefulness of the meta rules in that two technology clusters tend to move together and when the Technology cluster 2 and the Metal cluster moves down together, there is higher chance that the Technology cluster 1 goes down. Rules 3 shows that the Gold cluster and the Financial cluster move in opposite directions, which is a widely understood market behavior.

5.2 Congressional Voting Records Database

The second data set that we experimented with is the Congressional Voting Records Database provided by [MM96]. This data set includes 435 transactions each corresponding to one Congressman's vote on 16 key issues. Thus, for each bill the data set has two different items, one corresponding to YES or NO vote. Our goal with this data set was not only to find item-clusters, but also to use these item-clusters to cluster the actual transactions.

We used our item-clustering algorithm to find the clusters of votes, by forming and partitioning the association-rule hypergraph. Our hypergraph partitioning algorithms found three item-clusters that are shown in Table 6. With these item-clusters we then clustered the actual transactions. As discussed in Section 4, this was done by assigning each transaction to the item-cluster that it matches the most. This lead to three transaction-clusters, one for each item-cluster. We evaluated the quality of these transaction-clusters by looking to what extent they represent voting records of congressmen belonging to the same party. In particular, for each transaction-cluster, we counted the number of democrats and republicans that belong to it. These results are shown in Table 7. From this table we see that the first cluster represents predominately republican congressmen, whereas the other two clusters represent predominately democrat congressmen.

We also used *AutoClass* to cluster the voting transactions. As opposed to clustering items, *AutoClass* can

Cluster	Voting Items
1	adoption-of-the-budget-resolution-NO, physician-fee-freeze-YES, el-salvador-aid-YES religious-groups-in-schools-YES, anti-satellite-test-ban-NO, aid-to-nicaraguan-contras-NO, mx-missile-NO, education-spending-YES, crime-YES, duty-free-exports-NO
2	adoption-of-the-budget-resolution-YES, physician-fee-freeze-NO, el-salvador-aid-NO, anti-satellite-test-ban-YES, aid-to-nicaraguan-contras-YES, mx-missile-YES, education-spending-NO, crime-NO
3	handicapped-infants-NO, water-project-cost-sharing-YES, immigration-YES, synfuels-corporation-cutback-YES, superfund-right-to-sue-YES, export-administration-act-south-africa-NO

Table 6: Clustering of Congressional Voting Items

Class	Cluster 1	Cluster 2	Cluster 3
Republican	159	7	2
Democrat	38	216	13

Table 7: Clustering of Congressional Voting Data Set

Class	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Republican	0	130	28	2	8
Democrat	163	22	30	48	4

Table 8: Clustering of Congressional Voting Data Set by *AutoClass*

be used directly to cluster transactions. *AutoClass* found five clusters, whose quality is shown in Table 8. This table shows the number of democrats and republicans that belong to the clusters found by *AutoClass*. As we can see from this table, clusters 1, 2, and 4 are clean clusters, since they correspond to congressmen that belong predominately to the same party. However, clusters 3 and 5 are mixed. In particular, cluster 3 is evenly divided among democrats and republicans, whereas cluster 5 has eight republicans and four democrats. These comparisons show that our method produced clusters that are comparable (if not better) to those of *AutoClass* in quality.

5.3 Protein Coding Database

Our next data set arises in molecular biology. Molecular biologists study genes within the cells of organisms to determine the metabolic function of the proteins that those genes code for. Faced with a new protein, biologists have to perform a very laborious and painstaking experimental process to determine the functionality of the protein. To rapidly determine the function of many previously unknown genes, biologists quickly generate short segments of protein-coding sequences (called expressed sequence tags, or ESTs) and match each EST against the sequences of the known proteins, using similarity matching algorithms [AGM$^+$90, PL88]. If the EST clusters are available that are related to each other functionally, then biologists can match the new protein against the EST clusters to find those EST clusters that match the protein most closely. At this point, the biologists can focus on experimentally verifying the functionalities of the protein represented by the matching EST clusters. Hence finding clusters of related ESTs is an important problem.

Our data set consists of 11,986 transactions and 662 different items. Each transaction corresponds to a known protein and each item corresponds to an EST. The transactions for each known protein contains the ESTs that are matched with it. We used our method of clustering the ESTs (*i.e.*, the items of the data set) by forming the association-rule hypergraph and partitioning it. In finding the association rules we used a support of 0.02%, which essentially created rules that are supported by at least three transactions (*i.e.*, proteins). This led to a hypergraph with 407 vertices and 128,082 hyperedges. We used HMETIS to find 40 partitions, out of which 39 of them satisfied the fitness criteria. These 39 EST clusters were then given to biologist (the authors of [NRS$^+$95, SCC$^+$95]) to determine whether or not they are related. Their analysis showed that most of the EST clusters found by our algorithm correspond to ESTs that are related. In fact, 17 of the 39 clusters are very good, as each corresponds to a single protein family, whereas most of the remaining clusters correspond to three or four different protein families. Furthermore, by looking at the connectivity of each sub-hypergraph that corresponds to each one of this second class of clusters, we were able to see that they correspond to weakly connected sub-clusters, whose further subdivision will create single-protein clusters. This is particularly important, since it verified that the association-rule hypergraph is extremely accurate in modeling the relations between the various items, and we can easily verify the quality of the clusters by looking at their connectivity. Also, our clustering algorithm took under five minutes to find the various clusters which includes the time required to find the association rules and form the hypergraph.

We also used *AutoClass* to try to find clusters of ESTs by transposing the data set. This transposed data set consisted of 662 transactions, one for each EST, and 11,986 items, one for each known protein. On this transposed data set, *AutoClass* found 25 clusters of which only two of them were good clusters. The remaining clusters were extremely poor, as they included a very large number of ESTs (in general the number of related ESTs is very small, under ten ESTs). *AutoClass* found clusters of size 128, 80, 72, 55, and many other clusters of size 20 or greater, all of which were bad mixed clusters. We believe that as in the case of the S&P 500 data set, *AutoClass* was unable to successfully cluster the ESTs due to the high dimensionality, and the sparsity of the data set. Moreover, the limitations of *AutoClass* to handle data sets that have high dimensionality was also manifested in the amount of time that it required. It took more than 5 hours to find these clusters on 662 ESTs. Comparing this to our sub-five minute runtime, we see that our item-clustering algorithm is able not only to find much better clusters, but it is also faster by roughly two orders of magnitude.

5.4 Web Document Data

Our next data set consisted of Web documents and the words that frequently appear in them. Specifically, we are interested in finding clusters of related words that tend to appear together in a document. These word clusters can then be used to find similar documents from the Web, or potentially serve as a description or label for classifying documents [WP97].

We collected 87 documents from the Network for Excellence in Manufacturing (NEM Online)[1] site. From these documents we used a stemming algorithm [Fra92] to find the distinct stems that appear in them. There were a total of 5772 distinct word stems. We then constructed 87 transactions (one for each document) such that a distinct word stem is in the transaction only if it is contained in the document. These transactions formed the data set that we used to cluster the related words together.

We used our item-clustering algorithm to find word-clusters by constructing the association-rule hypergraph and partitioning it. In particular, we found all the association rules with a support of at least 5% (*i.e.*, the words must appear in at least five documents), and formed the corresponding hypergraph. This association-rule hypergraph has 304 vertices and 73793 hyperedges. Note that out of the 5772 distinct word stems, only 304 survived the association-rule discovery process. The rest of words did not survive because all association rules containing these words had support less than the preset threshold. Pruning infrequent word stems is actually very important as these words do not appear often enough to characterize documents. Thus, by using the association-rule hypergraph we are able to automatically prune non-essential information. This hypergraph was then partitioned into 20 partitions all of which passed the fitness criteria. Five of these word-clusters are shown in Table 9 and the remaining can be found at *http://www.cs.umn.edu/∼han/word-cluster.html*. Looking at this table we see that the word clusters found by our algorithm are indeed related. For example, cluster 1 includes word stems such as *http*, *internet*, *site*, and *web*, that clearly correspond to web-related information. Similarly, cluster 2 corresponds to computing/electronic related words, cluster 3

[1] http://web.miep.org:80/miep/index.html

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
http	access	act	data	action
internet	approach	busi	engineer	administrate
mov	comput	check	includes	agenci
please	electron	enforc	manag	complianc
site	goal	feder	network	establish
web	manufactur	follow	services	health
ww	power	govern	softwar	law
	step	informate	support	laws
		page	systems	nation
		public	technologi	offic
			wide	regulations

Table 9: Word Clusters using Item Clustering (Note that words are stemmed.)

corresponds to government-related words, cluster 4 corresponds to software/networking related words, and cluster 5 corresponds to legal related words.

We also used *AutoClass* to find word-clusters by first transposing the data set. In this transposed data set, there are a total of 5772 transactions and 87 distinct items. *AutoClass* found 35 word clusters that include several very large clusters (each containing over 200 words). The five smallest (and also better) clusters are shown in Table 10 and the remaining clusters can be found at *http://www.cs.umn.edu/∼han/word-cluster.html*. Looking at this table, we see that the word-clusters found by *AutoClass* are quite poor. In fact, we cannot see any obvious relation among the most of the words of each cluster. In contrast, the word-clusters found by our algorithm are considerably better and we can easily identify the type of documents that they describe. It is also interesting to note that the first cluster found by our algorithm (the web-related words) are equally dispersed among the word-clusters found by *AutoClass*. The poor performance of *AutoClass* is particularly surprising since the dimensionality of the data set is relatively small (only 87 distinct items) and there are a relatively small number of transactions.

5.5 US Census Database

Finally, to test the scalability of our item-clustering algorithm on a large data set we used the Adult Database [MM96], which is a subset of US Census data. This database has 48,842 transactions, 6 continuous and 8 discrete attributes. One transaction corresponds to one person's census data information. The continuous attributes were discretized based on the schemes proposed in [SA96] in the preprocessing step. This discretization of the continues attributes, increased the total number of items from 14 to 108.

We constructed the association-rule hypergraph by finding all association rules that have a support of at least 0.1%, which lead to a hypergraph with 108 vertices and 541 hyperedges. After partitioning this hypergraph into 10 clusters, and eliminating the ones that did not satisfy the fitness criteria, we obtained the 9 item-clusters shown in Table 11. Looking at this table we see that our algorithm was successful in group-

Cluster 1	Cluster 2		Cluster 3		Cluster 4		Cluster 5	
copyright	adopt	comprehens	concern	court	cornell	effort	congress	affirm
design	efficientli	held	agent	doc	amend	amendments	employ	engineer
found	hr	html	documents	appeals	formerli	appear	equal	home
internate	http	hyper	juli	list	meet	news	homepag	house
object	librari	mov	apr	notificate	onlin	organize	ii	iii
	offices	please	nov	pac	own	pages	implementate	legisl
	procedur	programm	patents	recent	people	portions	legislate	mail
	automaticalli	reserv	register	sites	publications	sections	major	name
	resist	bas	bear	tac	select	server	nbsp	page
	basic	ww	timeout	topics	servers	structur	rerpresentatives	section
	bookmarks	changes	trademark	user	technic	uscod	senat	send
	com		uspto	word	version	visit	thomas	bills
					web	welcom	track	trade
					center	central	webmast	action

Table 10: Word Clusters using *AutoClass*

ing together items (*i.e.*, census attributes as well as sub-intervals of the continues attributes) that are highly related. For example, cluster 2 represents a class of people with low education and/or immigrant from Mexico, cluster 3 represents a class of people who are very young with high school degree (possibly attending colleges) and working less than 40 hours per week (possibly working only part time). Cluster 4 represents a class of people who are old (possibly retired) with low education. Cluster 9 represents a class of people who are young and have low-paying jobs. On the other hand, Cluster 6 represents a class of people who are in their mid 30's to 50's with bachelors or masters degree, working as managers or professionals. Cluster 8 represents a class of people who are in their mid 40's to 60's with professional degree and with capital gains of 15K.

This data set has the largest number of transactions compared to the previous data sets. Nevertheless, our algorithm was able to produce the above clusters in only 110 seconds. In particular, it took Apriori 56 seconds to find the association rules, and we were able to construct and partition the hypergraph in only 54 seconds. Thus, our algorithm scales very nicely with increasing data set size. The main reason for this is that we operate on the actual data set only for finding the association rules. Once these rules have been found, we then operate only on the association-rule hypergraph, whose size is independent of the number of transactions in the data set.

6 Conclusion and Future Works

In this paper, we have presented clustering of items based on the association rules and the applications of the discovered item clusters in different areas. The modeling of association rules using hypergraph and partitioning the association rule hypergraph leads to an efficient method of finding item clusters. The item clusters in turn can be used to cluster transactions, used to reduce the dimensionality of the clustering problem, or used to find meta association rules among different item clusters.

Our experiments indicate that clustering of items and transactions based on the association rules holds

Cluster 1	native-country:	China, India, Japan, Philippines,South,Taiwan,Vietnam	occupation:	Tech-support
	race:	Asian-Pac-Islander		
Cluster 2	native-country:	Mexico	education:	1st-4th,5th-6th,7th-8th,9th
	education-num:	2–5		
Cluster 3	age:	17.0–23.5	hours-per-week:	1.0–35.5
	education:	HS-grad,Some-college	education-num:	9–10
	marital-status:	Divorced	occupation:	Adm-clerical,Other-service,Sales
	relationship:	Own-child,Unmarried	race:	Black
Cluster 4	age:	62.5–91.0	relationship:	Other-relative
	education:	10th,11th	education-num:	6–7
	marital-status:	Separated,Widowed	occupation:	Craft-repair
	fnlwgt:	145414.0–158686.5, 206429.0–219711.5, 237109.5–289433.0		
Cluster 5	age:	50.5–57.5	workclass:	State-gov
	hours-per-week:	60.5–100.0	fnlwgt:	39446.0–65711.0, 117830.0–130880.5
	education:	Assoc-voc, Doctorate	education-num:	11, 16
Cluster 6	age:	37.5–39.5, 41.5–45.5, 47.5–50.5	workclass:	Local-gov,Self-emp-not-inc
	hours-per-week:	40.5–45.5, 49.5–50.5	fnlwgt:	91710.0–106659.0
	education:	Bachelors,Masters	education-num:	13–14
	occupation:	Exec-managerial,Prof-specialty	relationship:	Wife
Cluster 7	age:	31.5–37.5, 39.5–41.5	workclass:	Federal-gov
	hours-per-week:	35.5–39.5, 50.5–59.5	occupation:	Protective-serv
	education:	Assoc-acdm	education-num:	12
	fnlwgt:	169538.0–178384.0, 187726.0-206429.0, 329073.5–379773.0		
Cluster 8	age:	45.5–47.5, 57.5–62.5	workclass:	Self-emp-inc
	capital-gain:	15022.0–15427.5	hours-per-week:	59.5–60.5
	fnlwgt:	65711.0–91710.0, 178384.0–187726.0, 289433.0–329073.5	education:	Prof-school
	education-num:	15		
Cluster 9	age:	25.5–27.5	hours-per-week:	45.5–49.5
	fnlwgt:	158686.5–169538.0, 219711.5–237109.5	education:	12th
	occupation:	Handlers-cleaners,Machine-op-inspct,Transport-moving	education-num:	8

Table 11: Item Clusters of Adult Database

great promise. Our experiments with stock-market data and congressional voting data show that this clustering scheme is able to successfully group items that belong to the same group. The clustering of the congressional voting data using discovered item clusters show that this method is quite effective in finding clusters of transactions that correspond to either democrat or republican voting patterns. We have also found clusters of segments of protein-coding sequences from protein coding database using our item clustering method. The discovered clusters of protein-coding sequences share the same functionality and thus are very valuable to biologist for determining functionality of new proteins. Our experiments with Web documents show that clustering of related words produced by our item clustering method is much more effective than that produced by *AutoClass*, and our method is much faster.

In our ongoing research, we are investigating ways to improve the modeling such that the relationship among items are more accurately captured in the hypergraph. More specifically, we are evaluating different weight functions for the hypergraph edges. We are also working on to improve the hypergraph partitioner HMETIS such that the number of partitions can be determined automatically by evaluating fitness of the partitions. In the construction of meta transactions using discovered item clusters, we are evaluating different matching schemes that can be used in matching transactions against item clusters.

Our algorithm for constructing clusters of items can be used in many ways other than discussed in this paper. For example, discovered clusters can be used for improving accuracy, efficiency and robustness of classification algorithm, and for detecting deviations. It also appears possible to find interesting high confidence association rules that have very low support using discovered item clusters.

Acknowledgments

We would like to thank Elizabeth Shoop and John Carlis for providing the protein coding data and verifying our results. We would also like to thank Jerome Moore for providing the Web document data.

References

[AGM$^+$90] Stephen Altschul, Warren Gish, Webb Miller, Eugene Myers, and David Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[AIS93] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of 1993 ACM-SIGMOD Int. Conf. on Management of Data*, Washington, D.C., 1993.

[AMS$^+$96] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast discovery of association rules. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI/MIT Press, 1996.

[AS94] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th VLDB Conference*, pages 487–499, Santiago, Chile, 1994.

[Ber76] C. Berge. *Graphs and Hypergraphs*. American Elsevier, 1976.

[CHY96] M.S. Chen, J. Han, and P.S. Yu. Data mining: An overview from database perspective. *IEEE Transactions on Knowledge and Data Eng.*, 8(6):866–883, December 1996.

[CS96] P. Cheeseman and J. Stutz. Baysian classification (autoclass): Theory and results. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI/MIT Press, 1996.

[DJ80] R. Dubes and A.K. Jain. Clustering methodologies in exploratory data analysis. In M.C. Yovits, editor, *Advances in Computers*. Academic Press Inc., New York, 1980.

[EG95] T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM Journal on Computing*, 24(6):1278–1304, 1995.

[Fis95] D. Fisher. Optimization and simplification of hierarchical clusterings. In *Proc. of the First Int'l Conference on Knowledge Discovery and Data Mining*, pages 118–123, Montreal, Quebec, 1995.

[Fra92] W.B. Frakes. *Stemming algorithms*. Prentice Hall, 1992.

[GHK+96] M. Ganesh, E.H. Han, V. Kumar, S. Shekhar, and J. Srivastava. Visual data mining: Framework and algorithm development. Technical Report TR-96-021, Department of Computer Science, University of Minnesota, Minneapolis, 1996.

[GKMT97] D. Gunopulos, R. Khardon, H. Mannila, and H. Toivonen. Data mining, hypergraph transversals, and machine learning. In *Proc. of Symposium on Principles of Database Systems*, Tucson, Arizona, 1997.

[HF95] J. Han and Y. Fu. Discovery of multiple–level association rules from large databases. In *Proc. of the 21st VLDB Conference*, Zurich, Switzerland, 1995.

[HKK97] E.H. Han, G. Karypis, and V. Kumar. Scalable parallel data mining for association rules. In *Proc. of 1997 ACM-SIGMOD Int. Conf. on Management of Data*, Tucson, Arizona, 1997.

[HS95] M. A. W. Houtsma and A. N. Swami. Set-oriented mining for association rules in relational databases. In *Proc. of the 11th Int'l Conf. on Data Eng.*, pages 25–33, Taipei, Taiwan, 1995.

[JD88] A.K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[KA96] A.J. Knobbe and P.W. Adriaans. Analysing binary associations. In *Proc. of the Second Int'l Conference on Knowledge Discovery and Data Mining*, pages 311–314, Portland, OR, 1996.

[KAKS97] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Application in VLSI domain. In *Proceedings ACM/IEEE Design Automation Conference*, 1997.

[Lee81] R.C.T. Lee. Clustering analysis and its applications. In J.T. Toum, editor, *Advances in Information Systems Science*. Plenum Press, New York, 1981.

[LF78] S.Y. Lu and K.S. Fu. A sentence-to-sentence clustering procedure for pattern analysis. *IEEE Transactions on Systems, Man and Cybernetics*, 8:381–389, 1978.

[LSW97] B. Lent, A. Swami, and J. Widom. Clustering association rules. In *Proc. of the 13th Int'l Conf. on Data Eng.*, Birmingham, U.K., 1997.

[MJHS96] B. Mobasher, N. Jain, E.H. Han, and J. Srivastava. Web mining: Pattern discovery from world wide web transactions. Technical Report TR-96-050, Department of Computer Science, University of Minnesota, M inneapolis, 1996.

[MM96] C.J. Merz and P.M. Murphy. UCI repository of machine learning databases. In *http://www.ics.uci.edu/ mlearn/MLRepository.html*, 1996.

[NH94] R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. In *Proc. of the 20th VLDB Conference*, pages 144–155, Santiago, Chile, 1994.

[NRS+95] T. Newman, E.F. Retzel, E. Shoop, E. Chi, and C. Somerville. Arabidopsis thaliana expressed sequence tags: Generation, analysis and dissemination. In *Plant Genome III: International Conference on the Status of Plant Genome Research*, San Diego, CA, 1995.

[PL88] William R. Pearson and David J. Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences*, 85:2444–2448, 1988.

[SA95] R. Srikant and R. Agrawal. Mining generalized association rules. In *Proc. of the 21st VLDB Conference*, pages 407–419, Zurich, Switzerland, 1995.

[SA96] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proc. of 1996 ACM-SIGMOD Int. Conf. on Management of Data*, Montreal, Quebec, 1996.

[SAD+93] M. Stonebraker, R. Agrawal, U. Dayal, E. J. Neuhold, and A. Reuter. DBMS research at a crossroads: The vienna update. In *Proc. of the 19th VLDB Conference*, pages 688–692, Dublin, Ireland, 1993.

[SCC+95] E. Shoop, E. Chi, J. Carlis, P. Bieganski, J. Riedl, N. Dalton, T. Newman, and E. Retzel. Implementation and testing of an automated EST processing and analysis system. In Lawrence Hunter and Bruce Shriver, editors, *Proceedings of the 28th Annual Hawaii International Conference on System Sciences*, volume 5, pages 52–61. IEEE Computer Society Press, 1995.

[SD90] J.W. Shavlik and T.G. Dietterich. *Readings in Machine Learning*. Morgan-Kaufman, 1990.

[Sho96] Elizabeth Shoop. *The Design and Implementation of an Extended Database System to Support Biological Sequence Similarity Analysis*. PhD thesis, University of Minnesota, September 1996.

[TKR+95] H. Toivonen, M. Klemettinen, P. Ronkainen, K. Hatonen, and H. Mannila. Pruning and grouping discovered association rules. In *ECML-95 Workshop on Statistics, Machine Learning, and Knowledge Discovery in Databases,*, Heraklion, Greece, 1995.

[TSM85] D.M. Titterington, A.F.M. Smith, and U.E. Makov. *Statistical Analysis of Finite Mixture Distributions*. John Wiley & Sons, 1985.

[WP97] Marilyn Wulfekuhler and Bill Punch. Finding salient features for personal web page categories. In *6th WWW Conference*, Santa Clara, CA, 1997.