

Mining Hyperclique Patterns with Confidence Pruning

Hui Xiong*
Computer Science
Department
University of Minnesota
200 Union Street SE
Minneapolis, MN-55455,USA
huix@cs.umn.edu

Pang-Ning Tan
Computer Science
Department
University of Minnesota
200 Union Street SE
Minneapolis, MN-55455,USA
ptan@cs.umn.edu

Vipin Kumar
Computer Science
Department
University of Minnesota
200 Union Street SE
Minneapolis, MN-55455,USA
kumar@cs.umn.edu

ABSTRACT

Standard association-rule mining algorithms have relied on the support-based pruning strategy to discover interesting patterns. Although this strategy can ease the bottleneck of itemset generation, it can potentially miss many interesting patterns, particularly those with low support but high confidence. The problem becomes even more critical if items have widely differing support. For such data sets, setting up support threshold too low leads to generation of too many uninteresting associations involving items with substantially different levels of support, and setting support threshold too high leads to elimination of all patterns involving low-support items. To address these problems, we propose the concept of a hyperclique pattern, which uses an interestingness measure called h-confidence to find patterns containing items that are highly affiliated with each other. We show that h-confidence not only possesses the desirable downward closure property for identifying highly associated patterns at low support levels, it has the ability to remove spurious associations involving items from different support levels. In addition, we present an algorithm called hyperclique miner, which can effectively prune the cross-support patterns and efficiently discover hyperclique patterns at all levels of support. As demonstrated by our extensive experiments on both real and synthetic data sets, the performance of hyperclique miner is several orders of magnitude faster than frequent pattern generating algorithms, such as Apriori and CHARM, particularly at low levels of support. Finally, we show that hyperclique patterns are very promising for clustering items in a high dimensional space.

1. INTRODUCTION

In the past decade, association rule discovery has been

*Contact Author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WOODSTOCK '97 El Paso, Texas USA
Copyright 2003 ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

the subject of extensive research in data mining [1, 3, 4, 5, 10, 12, 21, 22]. Generating association rules is a computationally intensive problem due to the combinatorial explosion of possible rules. Standard algorithms such as Apriori [4] and FP-growth [15] rely on the support-based pruning strategy to constrain the exponential search space. Although this strategy can ease the computational bottleneck of association rule generation, setting the right minimum support threshold for a given data set is not a trivial problem.

- If the minimum support threshold is set too high, the algorithms can potentially miss many interesting patterns, particularly *high-confidence rules with low support levels* [16]. These rules are useful for characterizing interesting associations among less frequently bought items such as caviar and vodka, gold necklaces and earrings, or TVs and DVD players.
- If the minimum support threshold is set too low, the *computational and memory requirements* tend to increase significantly, and the number of rules generated also becomes prohibitively large, many of which may be uninteresting. In fact, many spurious associations involving *items with substantially different support levels* will be formed; e.g. $A \rightarrow B$, where A is an item with low support and B is an item with high support. The problem becomes even more critical for dense data sets, which contain a large number of long frequent patterns at low levels of support.

To illustrate the above problem, consider the `pumsb` census data set, which is a benchmark for evaluating the performance of association rule algorithms on dense transaction data sets¹. Figure 1 shows the support distribution of all items from this data set. Observe that the distribution is rather skewed, with 81.5% of the items having support less than 1% while 0.95% of them having support values greater than 90%. We can partition the items into five disjoint groups according to their support levels, as shown in Table 1. The group `S1` has the lowest support range (less than or equal to 1%) but contains the most number of items, i.e., 1739 items. In order to capture high confidence rules involving items from `S1`, we have to set the support threshold to be less than 1%. However, such a low support threshold degrades the performance of existing algorithms substantially

¹The data set is obtained from IBM Almaden at <http://www.almaden.ibm.com/cs/quest/demos.html>.

and produces too many rules that far exceed our ability to analyze them manually.

More importantly, if the support threshold is only 0.05%, many rules relating items from $S1$ with $S5$ are generated. Just to give an indication of the scale, out of 18847 frequent pairs involving items from $S1$ or $S5$ for the support threshold of 0.05%, about 93% contain mixtures of items from $S1$ and $S5$. The correlations within the mixed item pairs are extremely poor because the presence of the frequent item from $S5$ does not necessarily imply the presence of the rare item from $S1$. Thus, even though the support and confidence for some of the rules could be high, they are not interesting from a statistical perspective because they could happen simply by chance due to the high-support of items in $S5$. Also, from a market-basket analysis viewpoint, such patterns would correspond to uninteresting rules relating the infrequent items such as TV and DVD player to frequent items such as bread and milk.

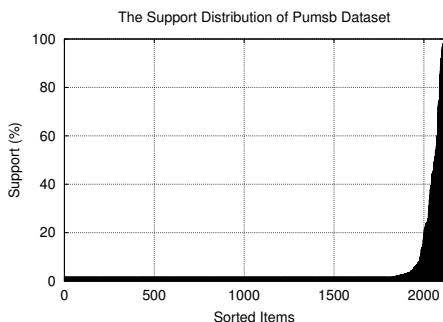


Figure 1: The support distribution of Pumsb.

Table 1: Groups of items for pumsb data set.

Group	$S1$	$S2$	$S3$	$S4$	$S5$
Support	0-1%	1%-5%	5%-40%	40%-90%	>90%
# Items	1735	206	101	51	20

The motivation for our current work is to establish a novel technique that can strike a balance between the ability to identify highly associated patterns at low support levels, and the ability to remove spurious associations involving items from different support levels. To satisfy the first requirement, the technique must be able to efficiently discover interesting patterns at very low support thresholds. To satisfy the second requirement, the technique must be able to prune away most of the uninteresting patterns using a reasonably intuitive interest measure. A naive approach would be to apply the existing association-rule mining algorithm with a very low minimum support threshold, and then use the interestingness measure to eliminate spurious patterns as a post-processing step. This approach may fail for one obvious reason: current algorithms break down at a very low support threshold. Although there has been recent work that attempts to find frequent itemsets without applying any support thresholds, such methods are either too expensive or are incomplete (using sampling [8] and other approximate methods [29]).

To address these issues, we introduce a new type of association pattern, called a hyperclique pattern, which contains only items that are strongly affiliated with each other.

In other words, given an itemset X that contains k items, say $\{x_1, x_2, \dots, x_k\}$, if any one of the items is present in a transaction, then there is a strong likelihood that the rest of the items in X would also be present in the same transaction. These patterns are valuable because they correspond to itemsets that contain tightly-coupled items. Discovering such itemsets would be extremely useful for a variety of applications such as clustering, copy detection [27], and collaborative filtering [11].

A key idea central to the discovery of hyperclique patterns is the use of an affinity measure called h-confidence, which is similar in nature to the concept of all-confidence recently proposed by Omiecinski [23]. The h-confidence measure, like the all-confidence measure, possesses an anti-monotonic property that allows us to incorporate the measure directly into the mining process, rather than using it during post-processing. However, in this paper, we show that h-confidence has an even more important property, called the *cross-support property*, which allows drastic pruning of the exponential search space by eliminating patterns involving items from different levels of support. For example, this property allows us to eliminate patterns relating a less frequently bought item such as caviar to a frequently-bought item such as milk even when the minimum support threshold is very low. Due to these properties, hyperclique patterns can be found even when the support threshold is set to zero.

1.1 Contribution

The main contributions of this paper are as follows:

1. We introduce a novel type of association pattern called a hyperclique pattern, which contains only items that are highly affiliated with each other.
2. We present an interestingness measure called h-confidence to evaluate the overall affinity among items within a hyperclique pattern. We show that this measure is not only anti-monotonic, but also possesses an additional desirable property that allows us to substantially prune the candidate space (even before applying the anti-monotonic property) and avoids generating patterns containing items with different support levels. In addition, we show that the h-confidence measure is highly correlated with the Jaccard measure [25] and the cosine measure for pairs of items[25].
3. We propose an algorithm called hyperclique miner for mining hyperclique patterns. This algorithm not only takes advantage of the anti-monotonic property of h-confidence, it also employs the cross-support property to dramatically reduce the exponential search space and efficiently generate hyperclique patterns at all levels of support. The proofs of correctness and completeness for this algorithm are also presented.
4. We perform extensive experiments to evaluate hyperclique miner and showed that the performance of this algorithm is several orders of magnitude faster than standard frequent pattern generating algorithms such as Apriori and Charm, particularly at low levels of support.
5. We describe two potential applications of mining hyperclique patterns including discovering interesting patterns involving low-support items and hyperclique-based

clustering. Both of them can significantly benefit from the discovery of hyperclique patterns.

1.2 Related Work

Early approaches to mining association patterns have focused on developing efficient algorithms for pruning the exponential search space. While support-based pruning [4] remains as the most widely-used approach, it is ineffective especially when the data set is dense or if analysts are interested in finding low support patterns. The concepts of maximal [5, 6] and closed itemsets [24, 30] have been proposed to address the computational challenges of mining association patterns in dense data sets. However, algorithms developed for mining these concepts may not be efficient at very low support thresholds. Liu et al. [20] has proposed another technique to address the low support problem by allowing users to specify different support thresholds for different items. Although this technique allows us to find frequent patterns at low support levels, it can potentially generate too many cross-support patterns, e.g., the aforementioned “caviar-milk problem”.

As a result, there has been considerable interest in developing techniques for mining association patterns without support constraints. These techniques are aimed at utilizing other interestingness measure to efficiently extract interesting patterns from data. Although various measures have been proposed in the statistics, machine learning, and data mining literature [28], most of them do not possess the desired anti-monotonic property for pruning the search space or have other limitations. For example, Cohen et al. [8] have proposed using the following similarity measure, $sim(x, y) = \frac{P(x \cap y)}{P(x \cup y)}$, to capture interesting pairs of items (x, y) without using a minimum support threshold. However, unlike hyperclique patterns, their analysis is restricted only to pairs of items.

Omicinski [23] proposed three measures for evaluating the interestingness of an association pattern as alternatives to the support measure. One of their proposed measures, called all-confidence, is similar in nature to the h-confidence measure described in this paper, although both measures are developed from different perspectives. Specifically, the all-confidence measure is introduced to find the minimum confidence of all possible rules that can be extracted from a given pattern. On the other hand, the h-confidence measure is proposed to define a hyperclique pattern P , in which the presence of an item $x \in P$ in a transaction strongly implies the presence of every other item $x' \in P$ in the same transaction. Omicinski proved that the all-confidence measure is anti-monotonic, and thus, can be used to effectively prune the exponential search space. In this paper, we showed that the h-confidence measure has an important cross-support property, which allows us to prune patterns containing items with different levels of support.

Wang et al. proposed the use of universal-existential upward closure property of confidence to extract association rules without specifying the support threshold. This property states that if an association rule $A \rightarrow c$ satisfies a minimum confidence threshold, then there must exist a specialized rule $A \cap x \rightarrow c$ such that its confidence is also greater than the threshold. However, the technique is only applicable if the data set contains items that belong to a set of mutually exclusive and exhaustive attribute values, e.g., $x \in \{Gender = Male, Gender = Female\}$. Thus,

if the rule $Married = Yes \rightarrow OwnCar = Yes$ passes the minimum confidence threshold, then either $Married = Yes, Gender = Male \rightarrow OwnCar = Yes$ or $Married = Yes, Gender = Female \rightarrow OwnCar = Yes$ must also pass the minimum confidence threshold. Nevertheless, this technique is still quite expensive for large data sets and is not applicable to market-basket data for which the items do not form any mutually exclusive nor exhaustive sets.

1.3 Overview

The remainder of this paper is organized as follows. Section 2 defines the concept of hyperclique patterns and formulates the problem of mining hyperclique patterns. In section 3, we describe the properties of hyperclique patterns. Section 4 proposes the hyperclique miner algorithm and provides an analysis of correctness and completeness of the proposed algorithm. Hyperclique-based clustering is introduced in Section 5. Section 6 presents our experimental results. Finally, in section 7, we draw conclusions and suggest future work.

2. PROBLEM FORMULATION

In this section, we define the hyperclique pattern mining problem. To facilitate our discussion, we first present some notations and terminology.

2.1 Hyperclique Pattern Concepts

A hypergraph $H = \{V, E\}$ consists of a set of vertices $\{V\}$ and a set of hyperedges $\{E\}$. The concept of a hypergraph is an extension of the concept of a graph in the sense that each hyperedge can connect more than two vertices. In our hypergraph model, we treat every itemset as a hypergraph in which each item is represented as a vertex and every vertex has a hyperedge to all other vertices. We call such a hypergraph as a hyperclique.

Definition 1. The **h-confidence** $hconf(P)$ of an itemset $P = \{i_1, i_2, \dots, i_m\}$ is a measure that reflects the overall affinity among items within the itemset. This measure is defined as $\min\{conf\{i_1 \rightarrow i_2, \dots, i_m\}, conf\{i_2 \rightarrow i_1, i_3, \dots, i_m\}, \dots, conf\{i_m \rightarrow i_1, \dots, i_{m-1}\}\}$, where $conf$ follows the confidence definition of an association rule [3].

Example 1. Consider an itemset $P = \{A, B, C\}$, assume that we have $\text{supp}(\{A\}) = 0.1$, $\text{supp}(\{B\}) = 0.1$, $\text{supp}(\{C\}) = 0.06$, $\text{supp}(\{A, B, C\}) = 0.06$, where supp follows the support definition of the association rule [3]. Then

$$\begin{aligned} conf\{A \rightarrow B, C\} &= \text{supp}(\{A, B, C\}) / \text{supp}(\{A\}) = 0.6 \\ conf\{B \rightarrow A, C\} &= \text{supp}(\{A, B, C\}) / \text{supp}(\{B\}) = 0.6 \\ conf\{C \rightarrow A, B\} &= \text{supp}(\{A, B, C\}) / \text{supp}(\{C\}) = 1 \end{aligned}$$

Hence, $hconf(P) = \min\{conf\{B \rightarrow A, C\}, conf\{A \rightarrow B, C\}, conf\{C \rightarrow A, B\}\} = 0.6$.

Definition 2. Given a transaction database and the set of all items $I = \{I_1, I_2, \dots, I_n\}$, a **hyperclique pattern** is a subset of items that has the h-confidence value greater than a preset h-confidence threshold.

In the hypergraph model, for a hyperclique pattern $P = \{i_1, i_2, \dots, i_m\}$, every item within this hyperclique pattern can be represented as a vertex and has a hyperedge to all other vertices (items). The weight of the hyperedge from one vertex i_k to all other vertices is defined as $conf\{i_k \rightarrow i_1, i_2, \dots, i_{k-1}, i_{k+1}, i_m\}$.

2.2 Problem Statement

We can formalize the hyperclique pattern mining problem as the following:

Given:

- 1) a set I of K Boolean items, $I = \{I_1, I_2, \dots, I_K\}$.
- 2) a set T of N transactions $T = \{t_1, \dots, t_N\}$, where each $t_i \in T$ is a subset of the set I .
- 3) a h-confidence threshold.
- 4) a support threshold.

Find:

Hyperclique patterns satisfying the user-specified support and h-confidence thresholds

Objectives:

1) **Completeness:** An algorithm is complete if it finds all hyperclique patterns which have support and h-confidence levels greater than user specified thresholds.

2) **Correctness:** An algorithm is correct if any hyperclique pattern it finds has support and h-confidence levels greater than user specified thresholds.

3) **Computational efficiency:** We use the standard frequent pattern mining algorithms such as Apriori and Charm as our baseline. The IO and CPU cost of generating hyperclique patterns should be less than those of the baseline algorithm. We assume that pruning using the h-confidence thresholds can be applied as a post-processing step for these algorithms.

3. HYPERCLIQUE PATTERN PROPERTIES

We illustrate the properties of hyperclique patterns in this section. These properties are useful for the efficient computation and interpretation of hyperclique patterns.

3.1 Anti-monotonic Property

As previously noted, the h-confidence measure is equivalent to the all-confidence measure proposed by Omiecinski. Given an itemset $P = \{i_1, i_2, \dots, i_m\}$, all-confidence is defined as the minimum confidence of all possible rules extracted from P . On the other hand, h-confidence is defined as the minimum confidence of all rules of the form $r : i_j \rightarrow i_1 i_2 \dots i_{j-1} i_{j+1} \dots i_m$. This measure can be computed efficiently in the following way.

LEMMA 1. *Given an itemset $C = \{i_1, i_2, \dots, i_m\}$, the h-confidence $hconf(C)$ is equivalent to*

$$\frac{\text{supp}(\{i_1, i_2, \dots, i_m\})}{\max_{1 \leq k \leq m} \{\text{supp}(\{i_k\})\}}. \quad (1)$$

PROOF. By definition 1, we have the following:

$$\begin{aligned} Hconf(C) &= \min\{\text{conf}\{i_1 \rightarrow i_2, \dots, i_m\}, \text{conf}\{i_2 \rightarrow i_1, i_3, \dots, i_m\}, \\ &\quad \dots, \text{conf}\{i_m \rightarrow i_1, \dots, i_{m-1}\}\} \\ &= \min\left\{\frac{\text{supp}(\{i_1, i_2, \dots, i_m\})}{\text{supp}(\{i_1\})}, \frac{\text{supp}(\{i_1, i_2, \dots, i_m\})}{\text{supp}(\{i_2\})}, \right. \\ &\quad \left. \dots, \frac{\text{supp}(\{i_1, i_2, \dots, i_m\})}{\text{supp}(\{i_m\})}\right\} \\ &= \text{supp}(\{i_1, \dots, i_m\}) \times \min\left\{\frac{1}{\text{supp}(\{i_1\})}, \frac{1}{\text{supp}(\{i_2\})}, \right. \\ &\quad \left. \dots, \frac{1}{\text{supp}(\{i_m\})}\right\} \\ &= \frac{\text{supp}(\{i_1, i_2, \dots, i_m\})}{\max_{1 \leq k \leq m} \{\text{supp}(\{i_k\})\}}. \end{aligned}$$

□

However, one can show that the association rule with minimum confidence must have exactly the same form as r , which is why h-confidence is equivalent to the all-confidence measure.

Omiecinski have shown that all-confidence is an anti-monotonic measure, i.e., if an itemset $\{i_1, \dots, i_m\}$ is above the all-confidence threshold, so is every subset of size $m-1$.

LEMMA 2. *The h-confidence is monotonically non-increasing as the size of the hyperclique pattern increases.*

PROOF. Let $Hconf(P)$ denote the h-confidence for a hyperclique pattern $P = \{i_1, i_2, \dots, i_m\}$. Then for any hyperclique $P' = P \cup i_{m+1}$, the h-confidence $Hconf(P') = Hconf(P \cup i_{m+1})$. According to Definition 1, we obtain:

$$\begin{aligned} Hconf(P \cup i_{m+1}) &= \min\{\text{conf}\{i_1 \rightarrow i_2, \dots, i_{m+1}\}, \text{conf}\{i_2 \rightarrow i_1, i_3, \dots, i_{m+1}\}, \\ &\quad \dots, \text{conf}\{i_m \rightarrow i_1, i_2, \dots, i_{m-1}, i_{m+1}\}, \text{conf}\{i_{m+1} \rightarrow i_1, \\ &\quad \dots, i_m\}\} \\ &\leq \min\{\text{conf}\{i_1 \rightarrow i_2, \dots, i_{m+1}\}, \text{conf}\{i_2 \rightarrow i_1, i_3, \dots, i_{m+1}\}, \\ &\quad \dots, \text{conf}\{i_m \rightarrow i_1, i_2, \dots, i_{m-1}, i_{m+1}\}\} \\ &= \min\left\{\frac{\text{supp}(\{i_1, \dots, i_{m+1}\})}{\text{supp}(\{i_1\})}, \frac{\text{supp}(\{i_1, \dots, i_{m+1}\})}{\text{supp}(\{i_2\})}, \right. \\ &\quad \left. \dots, \frac{\text{supp}(\{i_1, \dots, i_{m+1}\})}{\text{supp}(\{i_m\})}\right\} \\ &\leq \min\left\{\frac{\text{supp}(\{i_1, \dots, i_m\})}{\text{supp}(\{i_1\})}, \frac{\text{supp}(\{i_1, \dots, i_m\})}{\text{supp}(\{i_2\})}, \right. \\ &\quad \left. \dots, \frac{\text{supp}(\{i_1, \dots, i_m\})}{\text{supp}(\{i_m\})}\right\} \\ &= \min\{\text{conf}\{i_1 \rightarrow i_2, \dots, i_m\}, \text{conf}\{i_2 \rightarrow i_1, i_3, \dots, i_m\}, \\ &\quad \dots, \text{conf}\{i_m \rightarrow i_1, \dots, i_{m-1}\}\}. \\ &= Hconf(P) \end{aligned}$$

Therefore, Lemma 2 holds. Note that we have used the anti-monotonic property of the conventional support measure in the above proof. □

This property allows us to push the h-confidence constraint into the search algorithm. Thus, when searching for hyperclique patterns, a candidate pattern $\{i_1, \dots, i_m\}$ needs to be checked only if all its subsets of size $m-1$ are hyperclique patterns. This pruning strategy is similar to the support-based pruning strategy used by existing frequent pattern mining algorithms. The anti-monotonic property also allows us to develop a flexible pruning scheme that applies either support-based or h-confidence-based pruning, depending on the choice of h-confidence and support thresholds.

Below, we provide another useful bound for the efficient computation of h-confidence.

LEMMA 3. *Given an itemset $C = \{i_1, \dots, i_m\}$, the h-confidence $hconf(C)$ has the following upper bound:*

$$\frac{\min_{1 \leq l \leq m} \{\text{supp}(\{i_l\})\}}{\max_{1 \leq k \leq m} \{\text{supp}(\{i_k\})\}}$$

PROOF. By Lemma 1, we know:

$$\begin{aligned} Hconf(C) &= \frac{\text{supp}(\{i_1, i_2, \dots, i_m\})}{\max_{1 \leq k \leq m} \{\text{supp}(\{i_k\})\}} \\ &\leq \frac{\min_{1 \leq l \leq m} \{\text{supp}(\{i_l\})\}}{\max_{1 \leq k \leq m} \{\text{supp}(\{i_k\})\}} \end{aligned}$$

□

When searching for hyperclique patterns, Lemma 3 can be used to further prune the candidate patterns. Because the support of each individual item is stored in memory, we can easily compute the upper bound of the h-confidence for each candidate itemset using Lemma 3. If the upper bound of the h-confidence is below the h-confidence threshold, we can prune the candidate itemset immediately. Furthermore, we can obtain an even tighter upper bound on the h-confidence of a size- k candidate itemset by replacing the numerator of the expression given in Lemma 3 with the minimum support of its subsets of size $k - 1$. This may help us prune the candidate space more aggressively.

3.2 Cross-Support Property

The hyperclique pattern has another important property as shown below in Lemma 4, which can help us *avoid identifying spurious patterns across different support levels*. We call such spurious patterns as cross-support patterns.

LEMMA 4. *Given two itemsets $S_i = \{x_1, x_2, \dots, x_m\}$ and $S_j = \{y_1, y_2, \dots, y_n\}$ with the condition that $\max_{1 \leq p \leq m} \{supp(\{x_p\})\} < \min_{1 \leq q \leq n} \{supp(\{y_q\})\}$, the h-confidence value of any cross-support pattern $P = \{x_{i_1}, x_{i_2}, \dots, x_{i_k}, y_{j_1}, y_{j_2}, \dots, y_{j_l}\}$ from these two itemsets has an upper bound:*

$$\frac{\max_{1 \leq p \leq m} \{supp(\{x_p\})\}}{\min_{1 \leq q \leq n} \{supp(\{y_q\})\}} \quad (2)$$

PROOF. Let $hconf(P)$ indicate the h-confidence for the cross-support pattern P . According to the anti-monotonic property of the support, Lemma 1, and the given condition $\max_{1 \leq p \leq m} \{supp(\{x_p\})\} < \min_{1 \leq q \leq n} \{supp(\{y_q\})\}$, we can prove this lemma as follows:

$$\begin{aligned} hconf(P) &= \frac{supp(\{x_{i_1}, x_{i_2}, \dots, x_{i_k}, y_{j_1}, y_{j_2}, \dots, y_{j_l}\})}{\max\{supp(x_{i_1}), \dots, supp(x_{i_k}), supp(y_{j_1}), \dots, supp(y_{j_l})\}} \\ &= \frac{supp(\{x_{i_1}, x_{i_2}, \dots, x_{i_k}, y_{j_1}, y_{j_2}, \dots, y_{j_l}\})}{\max\{supp(y_{j_1}), \dots, supp(y_{j_l})\}} \\ &\leq \frac{supp(\{x_{i_1}, x_{i_2}, \dots, x_{i_k}, y_{j_1}, y_{j_2}, \dots, y_{j_l}\})}{\min\{supp(y_{j_1}), \dots, supp(y_{j_l})\}} \\ &\leq \frac{supp(\{x_{i_1}, x_{i_2}, \dots, x_{i_k}, y_{j_1}, y_{j_2}, \dots, y_{j_l}\})}{\min_{1 \leq q \leq n} \{supp(\{y_q\})\}} \\ &\leq \frac{supp(\{x_{i_1}, x_{i_2}, \dots, x_{i_k}\})}{\min_{1 \leq q \leq n} \{supp(\{y_q\})\}} \\ &\leq \frac{\max_{1 \leq p \leq m} \{supp(\{x_p\})\}}{\min_{1 \leq q \leq n} \{supp(\{y_q\})\}} \quad \square \end{aligned}$$

Lemma 4 provides an upper bound for the h-confidence values of all possible cross-support patterns from two itemsets with different levels of support. In other words, if we set the h-confidence threshold higher than this upper bound, we do not need to generate any cross-support pattern as the candidate pattern during the pattern searching process.

COROLLARY 1. *Given a set of items, we can sort them in increasing order of support as shown in Figure 2. (For illustrative purposes, we assume that the support increases linearly.) For any given item x with support s_x , we can form a set of items I_x consisting of all items having support greater than or equal to s_x . Given an h-confidence threshold,*

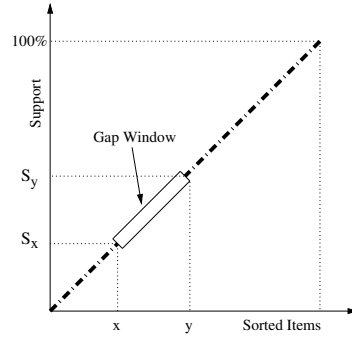


Figure 2: Cross-Support Technique Illustration.

h_c , we can find a corresponding item y with support s_y such that: (1) There exists a set I_y consisting of all items having support less than or equal to s_y , and (2) No cross-support patterns involving items from both I_x and I_y can have h-confidence greater than h_c .

PROOF. First, we add all items with support value less than $s_x \times h_c$ into the set I_y . The item with the highest support in I_y is chosen as y . Furthermore, $s_y/s_x < h_c$ due to the way I_y is constructed. Using Lemma 4 by treating I_x as S_j and I_y as S_i , the upper bound of h-confidence for any cross-support patterns involving I_x and I_y is s_y/s_x , which must be less than h_c . \square

For a specified h-confidence threshold, Corollary 1 provides one way to construct itemsets with the different levels of support and guarantees all cross-support patterns from these itemsets are not hyperclique patterns.

3.3 High Affinity Property

In this subsection, we present the high affinity property of hyperclique patterns. As shown by Lemma 5 and Lemma 6 below, we prove that the h-confidence measure is highly correlated with the Jaccard measure and the cosine measure for pairs of items.

LEMMA 5. *After pruning with the h-confidence threshold h_c , the Jaccard value of every size-two hyperclique pattern is greater than or equal to $h_c/2$, where the Jaccard value follows the classic definition of the Jaccard measure. Given a pair itemset $\{A, B\}$, the Jaccard measure [25] can be computed as $\frac{supp(\{A, B\})}{supp(\{A\}) + supp(\{B\}) - supp(\{A, B\})}$.*

PROOF. Let $P = \{i_1, i_2\}$ be any hyperclique pattern which satisfies the minimum h-confidence threshold h_c . By Lemma 1, $hconf(P) = \frac{supp(\{i_1, i_2\})}{\max\{supp(\{i_1\}), supp(\{i_2\})\}}$. Without loss of generality, let $supp(\{i_1\}) \geq supp(\{i_2\})$. Then $hconf(P) = \frac{supp(\{i_1, i_2\})}{supp(\{i_1\})}$. Since $supp(\{i_1\}) \geq supp(\{i_2\})$ and $Jaccard(P) = \frac{supp(\{i_1, i_2\})}{supp(\{i_1\}) + supp(\{i_2\}) - supp(\{i_1, i_2\})}$, we know $Jaccard(P) \geq \frac{supp(\{i_1, i_2\})}{2supp(\{i_1\})} = h_c/2$. \square

LEMMA 6. *After pruning with the h-confidence threshold h_c , the cosine value of every size-two hyperclique pattern is greater than or equal to h_c , where the cosine value follows the classic definition of the cosine measure. Given an item-pair $\{A, B\}$, the cosine measure [25] can be computed as $\frac{supp(\{A, B\})}{\sqrt{supp(A) * supp(B)}}$.*

PROOF. Let $P = \{i_1, i_2\}$ be any pair hyperclique pattern which satisfies the h-confidence threshold h_c . By Lemma 1, $hconf(P) = \frac{supp(\{i_1, i_2\})}{\max\{supp(\{i_1\}), supp(\{i_2\})\}}$. Without loss of generality, let $supp(\{i_1\}) \geq supp(\{i_2\})$. Then $hconf(P) = \frac{supp(\{i_1, i_2\})}{supp(\{i_1\})}$. Since $Cosine(P) = \frac{supp(\{i_1, i_2\})}{\sqrt{supp(i_1) * supp(i_2)}}$ and $supp(\{i_1\}) \geq supp(\{i_2\})$, we get $Cosine(P) \geq \frac{supp(\{i_1, i_2\})}{supp(\{i_1\})} \geq h_c$. \square

For longer itemsets, we can compute the average Jaccard or cosine measure for every pair of items within a given itemset. Since every pair of items within a hyperclique pattern has h-confidence greater than h_c , the average pairwise Jaccard or cosine measure of an itemset must also satisfy the above lemma.

4. HYPERCLIQUE MINER ALGORITHM

In this section, we present a level-wise algorithm, called hyperclique miner, for discovering hyperclique patterns.

TID	Items
1	1, 2
2	1, 2
3	1, 3, 4
4	1, 2
5	1, 2
6	1, 2
7	1, 2, 3, 4, 5
8	1, 2
9	1, 2
10	2, 3, 5

Item	Support
1	0.9
2	0.9
3	0.3
4	0.2
5	0.2

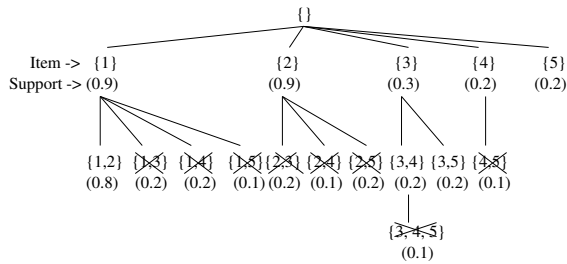


Figure 3: A running example.

Since hyperclique pattern mining is a combinatorial search problem, we can use Rymon’s generic set-enumeration tree search framework [26] to demonstrate how sets of items are to be completely enumerated.

Example 2. We illustrate how hyperclique miner works using the running example as shown in Figure 3. As can be seen, the process of searching hyperclique patterns is illustrated by the generation of branches of a set-enumeration tree. In this running example, we set the minimum support threshold to zero and the minimum h-confidence threshold to 55%. There are two kinds of pruning techniques which we can enforce in our algorithm.

1. We can prune itemsets by the anti-monotonic property of the h-confidence measure (Lemma 2). For instance, by Lemma 1, the h-confidence of the candidate pattern $\{4, 5\}$ is $supp(\{4, 5\})/\max\{supp(\{4\}), supp(\{5\})\} = 0.1/0.2 = 0.5$, which is less than 55%. Hence, the itemset $\{4, 5\}$ is not a hyperclique pattern and is pruned.

Then, we can prune the candidate pattern $\{3, 4, 5\}$ by the anti-monotonic property of the h-confidence measure, since this pattern has one subset $\{4, 5\}$, which is not a hyperclique pattern.

2. We can prune cross-support patterns by the cross-support property of the hyperclique pattern. For instance, for the sorted item list $\{1, 2, 3, 4, 5\}$, if we split this item list into two itemsets $S_1 = \{1, 2\}$ and $S_2 = \{3, 4, 5\}$, We can compute the h-confidence upper bound for cross-support patterns from these two itemsets by Lemma 4, which is equal to $\max\{supp(\{3\}), supp(\{4\}), supp(\{5\})\}/\min\{supp(\{1\}), supp(\{2\})\} = 3/9 = 0.33$. Hence, every cross-support pattern must have the h-confidence value less than 0.33. As a result, for the h-confidence threshold 55%, we can prune all these cross-support pattern before generating them as candidate patterns. In contrast, without applying cross-support pruning, we have to generate six cross-support patterns including $\{1, 3\}$, $\{1, 4\}$, $\{1, 5\}$, $\{2, 3\}$, $\{2, 4\}$, and $\{2, 5\}$ as candidate patterns and prune them later by computing the h-confidence values. Note that, the anti-monotonic property of the h-confidence measure cannot help prune those six candidate patterns, since every subset of those patterns are hyperclique patterns.

4.1 Algorithm Description

The Hyperclique Miner is an algorithm which can generate all hyperclique patterns with support and h-confidence above user-specified support and h-confidence thresholds.

Hyperclique Miner

Input:

- 1) a set F of K Boolean feature types $F = \{f_1, f_2, \dots, f_K\}$
- 2) a record T of N transactions $T = \{t_1 \dots t_N\}$, each $t_i \in T$ is a record with K attributes $\{i_1, i_2, \dots, i_K\}$ taking values in $\{0, 1\}$, where the i_p is the boolean value for the feature type f_p .
- 3) A user specified h-confidence threshold (min_conf)
- 4) A user specified support threshold (min_supp)

Output:

hyperclique patterns with h-confidence $> min_conf$ and support $> min_supp$

Method:

- 1) Get size-1 prevalent items
- 2) Construct itemsets at different levels of support
- 3) **for** size of itemsets in $(2, 3, \dots, K - 1)$ **do**
- 4) Generate candidate hyperclique patterns using the *generalized apriori-gen* algorithm
- 5) Prune based on the support of hyperclique patterns
- 6) Prune based on the h-confidence of candidate hyperclique patterns
- 7) Generate hyperclique patterns

Explanation of the detailed steps of the algorithm

Step 1 scans the database and gets the support for every item. Items with support above min_supp form size-1 candidate set C_1 . The h-confidence values for size-1 itemsets are 1. All items in the set C_1 are sorted by the support values and relabeled in alphabetic order.

Step 2 We construct itemsets in the way that items at the same support levels are grouped into the same itemset

and no cross-support patterns are satisfied with the specified h-confidence threshold.

Step 3 to Step 7 loops through 2 to K-1 to generate qualified hyperclique patterns of size 2 or more. It stops whenever an empty candidate hyperclique set of some size is generated.

Step 4 uses generalized *apriori_gen* to generate candidate hyperclique patterns of size k from prevalent hyperclique patterns of size k-1. The generalized *apriori_gen* function is an adaptation of the *apriori_gen* function of the *Apriori* algorithm [4]. Let C_{k-1} indicate the set of all prevalent hyperclique patterns of size k-1. The function works as follows. First, in the join step, we join C_{k-1} with C_{k-1} and get candidate set C_k . Next in the prune step, we delete all candidate hyperclique patterns $c \in C_k$ based on the following three conditions: 1) some k-1 subset of c is not in C_{k-1} . (Recall that this prune step is also done in *apriori_gen* [4] because of the anti-monotonicity property of support). Since our h-confidence measure also has the anti-monotonicity property, the same scheme-level confidence-pruning can apply here. 2) By Lemma 3, for $c = \{i_1, \dots, i_k\}$, if $\frac{\min_{1 \leq i \leq k} \{supp(\{i_i\})\}}{\max_{1 \leq p \leq k} \{supp(\{i_p\})\}} < min_conf$, the h-confidence of c cannot be greater than min_conf . 3) cross-support candidate hyperclique patterns.

Step 5 calculates the support for all candidate hyperclique patterns in C_k and prunes this candidate set using support threshold min_supp .

Step 6 calculates the h-confidence for all candidate hyperclique patterns in C_k and prunes this candidate set using h-confidence threshold min_conf .

Step 7 generates all prevalent hyperclique patterns.

4.2 Proofs of Completeness and Correctness

THEOREM 1. *The hyperclique miner algorithm is complete.*

PROOF. The completeness of the hyperclique miner algorithm can be shown by the following two facts. The first is that a set-enumeration tree search [26] is complete. The second is that branches of the set-enumeration tree are pruned if and only if they lead to only infrequent hyperclique patterns or low-confident frequent hyperclique patterns. \square

THEOREM 2. *The hyperclique Miner algorithm is correct.*

PROOF. The hyperclique miner algorithm identifies only frequent and confident hyperclique patterns. This is due to the fact that both the support-based pruning function and the hyperclique-confident-based pruning function within the body of the hyperclique miner continually remove any infrequent or low-confident itemsets. \square

5. HYPERCLIQUE-BASED CLUSTERING

In this section, we present how to use hyperclique patterns for clustering high dimensional data.

Clustering of large dimensional datasets is one of the biggest challenges in data mining. Traditional clustering schemes such as *Autoclass* [7] and K-means [17] tend to produce extremely poor results when they are directly applied to large high-dimensional datasets [2, 14]. One promising approach, proposed by Han et al. [13], is to cluster the data in a high dimensional space based on a hypergraph model. In this

approach, the authors used frequent itemsets to capture relationships of items of size greater than or equal to 2. For example, if $\{A, B, C\}$ is a frequent itemset, then the hypergraph contains a hyperedge that connects A, B, and C. The authors considered the rules for this itemset to be $\{A, B\} \rightarrow \{C\}$ (0.4), $\{A, C\} \rightarrow \{B\}$ (0.6), and $\{B, C\} \rightarrow \{A\}$ (0.8), where the value in the parentheses is the confidence for the rule. Then they set the average confidence as the weight for the hyperedge connecting A, B, and C, which is 0.6,

However, their approach has two difficulties. First, the frequent pattern is not a good representative to capture the overall affinity among items. For example, the frequent patterns crossing different levels of support may not have strong affinity among items, even though a high average confidence (weight) for the hyperedge can be obtained. Second, since many items have very small support, the number of frequent patterns (hyperedges) has to be very large if we want to cover more items. This will result in poor clustering results, since the hypergraph has too many hyperedges and it will be hard to partition a dense hypergraph.

In this paper, we propose a hyperclique-based clustering approach. Based on the descriptions from previous sections, we have already seen that hyperclique patterns contain items that are highly affiliated with each other and that the h-confidence is a good interest measure to evaluate the overall affinity among items within a hyperclique pattern. Therefore, the hyperclique pattern is a good representative to capture the relationships of items of size greater than or equal to 2. Also, we can easily identify the hyperclique pattern at all levels of support. As a result, in our hypergraph model, each hyperclique pattern is represented by a hyperedge whose weight is equal to the h-confidence of this hyperclique pattern. For example, if $\{A, B, C\}$ is a hyperclique pattern and the h-confidence for this pattern is 0.8, then the hypergraph contains a hyperedge that connects A, B, and C. The weight for this hyperedge is 0.8.

After constructing the hypergraph, we can use the hypergraph representation to cluster relatively large groups of related items by partitioning it into highly connected partitions. We 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. In our experiment, we use HMETIS. HMETIS [19, 18] is a multi-level hypergraph partitioning algorithm that has been shown to produce high quality bi-sections on a wide range of problems arising in scientific and VLSI applications. HMETIS minimizes the weighted hyperedge cut and thus tends to create partitions in which the connectivity among the vertices in each partition is high, resulting in good clusters.

Our hyperclique-based clustering approach has three advantages. First, items have varied levels of support; therefore, clusters of items can also occur at varied levels of support. Since hyperclique patterns can be identified from all levels of support and the affinity of items within a hyperclique pattern can be controlled by the minimum h-confidence threshold, these patterns should serve as good representatives of hyperedges to connect items. As a result, the effect of clustering based on hyperclique patterns can be significantly better than clustering based on frequent patterns. Second, a hyperclique-based clustering scheme does not require dimensionality reduction [9], because it uses the hypergraph model to represent relations among items. In con-

trast, the traditional clustering schemes, such as k-means and Autoclass, usually cannot avoid doing dimension reduction in order to achieve better clustering results. Finally, our hypergraph model allows us to correctly determine the quality of the clusters by looking at the internal connectivity of the nodes in each cluster.

6. EXPERIMENTAL EVALUATION

We present extensive experiments to evaluate the performance of the hyperclique miner algorithm and the quality of hyperclique patterns. Specifically, we demonstrate: (1) the effectiveness of h-confidence pruning, (2) the scalability of the hyperclique miner algorithm, and (3) the overall quality of hyperclique patterns.

6.1 The Experimental Setup

Our experiments were performed on both real and synthetic data sets. Synthetic data sets were generated by using the IBM Quest synthetic data generator [4], which gives us the flexibility of controlling the size and dimensionality of the database. A summary of the parameter settings used to create the synthetic data sets is presented in Table 2, where $|T|$ is the average size of a transaction, N is the number of items, and $|L|$ is the maximum number of potential frequent itemsets. Each data set contains 100000 transactions, with an average frequent pattern length equal to 4.

Table 2: Parameter settings for synthetic data sets.

Data set name	$ T $	$ L $	N	Size(MBytes)
T10.L1000.N10000	10	1000	10000	4.96
T20.L2000.N20000	20	2000	20000	10.73
T30.L3000.N30000	30	3000	30000	16.43
T40.L4000.N40000	40	4000	40000	22.13

The real data sets are obtained from several application domains. The characteristics of each data set ² is given in Table 3.

Table 3: Real Dataset Characteristics.

Data set	#Item	#Record	Avg. Length	Source
Pumsb	2113	49046	74	IBM Almaden
Pumsb*	2089	49046	50	IBM Almaden
Chess	75	3196	37	UCI Repository
Mushroom	119	8124	23	UCI Repository
LA1	29704	3204	145	TREC-5
Retail	14462	57671	129	Retail Store

The first four data sets in the table, i.e., **pumsb**, **pumsb***, **chess**, and **mushroom**, are often used as benchmark for evaluating the performance of association rule algorithms on dense data sets. The **pumsb** and **pumsb*** data sets correspond to binarized versions of a census data set. The difference between them is that **pumsb*** does not contain items with support greater than 80%. The LA1 data set is part of the TREC-5 collection ³ and contains news articles from

²Note that the number of items shown in Table 3 for **pumsb**, **pumsb***, **chess**, and **mushroom** are quite different from the numbers reported in [30], because we only consider item IDs for which the count is at least one. For example, although the minimum item ID in **pumsb** is 0 and the maximum item ID is 7116, there are only 2113 distinct item IDs that appear in the data set.

³The data set is available at <http://trec.nist.gov>.

the Los Angeles Times. Finally, **retail** is a masked data set obtained from a large mail-order company.

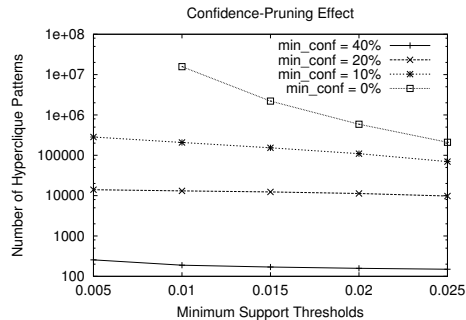


Figure 4: Number of patterns generated by the hyperclique miner on LA1 data set.

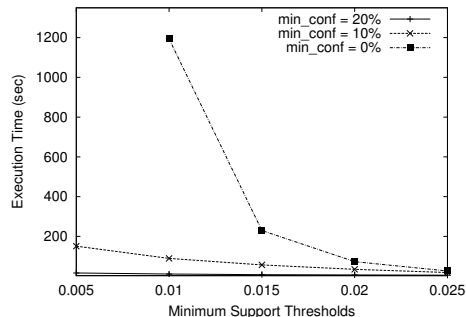


Figure 5: The execution time of the hyperclique miner on LA1 data set.

Our experiments were performed on a Sun Ultra 10 workstation with a 440 MHz CPU and 128 Mbytes of memory running the SunOS 5.7 operating system. Note that we have implemented hyperclique miner on top of the publicly available Apriori implementation by Borgelt (<http://fuzzy.cs.uni-magdeburg.de/borgelt>). As a result, the performance of hyperclique miner is almost equivalent to Apriori when the h-confidence threshold is set to zero.

6.2 The Pruning Effect of Hyperclique Miner

The purpose of this experiment is to demonstrate the effectiveness of the h-confidence pruning on hyperclique pattern generation. First, we illustrate how the performance of the algorithm changes as the h-confidence threshold is increased.

Figure 4 shows the number of patterns generated from the LA1 dataset at different h-confidence thresholds. As can be seen, at a fixed support threshold, the number of generated patterns increases quite dramatically with decreasing h-confidence thresholds. For example, when the support threshold is 0.005, the number of patterns generated is greater than 10^8 when the h-confidence threshold is equal to zero. In contrast, there are only several hundred hyperclique patterns when the h-confidence threshold is set to 40%. Moreover, when the h-confidence threshold is zero, we can see a rapid increase of the number of patterns generated as the support threshold is decreased. However, for a h-confidence threshold greater than or equal to 10%, the

number of patterns generated increases much less rapidly with decreasing support threshold.

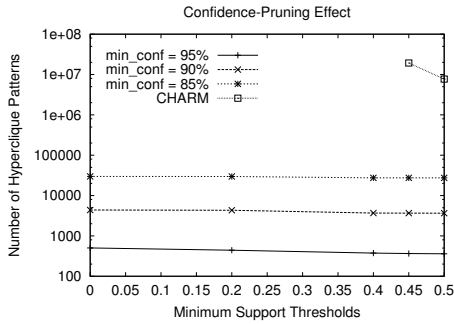


Figure 6: Number of patterns generated by the hyperclique miner and CHARM on pumsb data set.

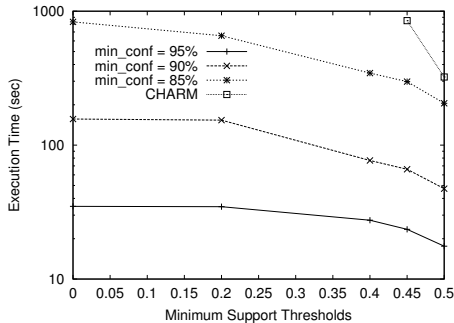


Figure 7: The execution time of the hyperclique miner and CHARM on pumsb data set.

Figure 5 shows the execution time of hyperclique miner when applied to the LA1 data set. Note that the execution time reduces significantly with increasing h-confidence thresholds. For instance, we can discover hyperclique patterns in just a few seconds at 20% h-confidence threshold and 0.005 support threshold. Without h-confidence pruning, the traditional algorithms such as Apriori would break down at 0.005 support threshold.

The above results suggest a trade-off between execution time and the number of hyperclique patterns generated at different h-confidence thresholds. In practice, analysts may start with a high h-confidence threshold first at support threshold close to zero, to rapidly extract the highly affiliated patterns, and then gradually reduce the h-confidence threshold to obtain more patterns that are less tightly-coupled.

Next, we compare the performance of hyperclique miner against state-of-the-art frequent pattern mining algorithms on dense data sets. Recently, the CHARM algorithm was proposed by Zaki et al. [30] to efficiently discover frequent closed itemsets. The results of their paper suggest that CHARM outperforms many existing algorithms, such as Apriori [4], CLOSET [24], and MAFIA [6], on dense data sets particularly at low levels of support. Thus, we choose CHARM as the baseline for comparing the performance of our algorithm on dense data sets.

Figure 6 shows the number of patterns generated by hyperclique miner and CHARM for the pumsb data set. As

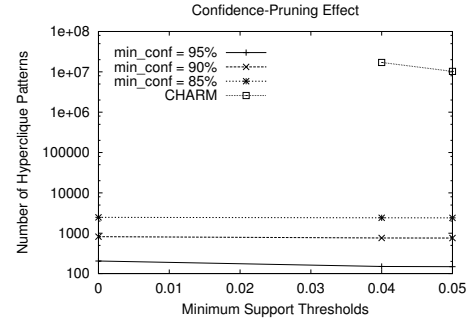


Figure 8: Number of patterns generated by the hyperclique miner and CHARM on pumsb* data set.

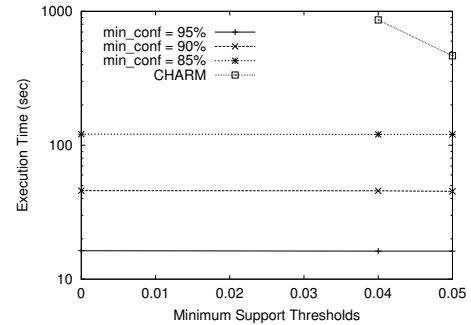


Figure 9: The execution time of the hyperclique miner and CHARM on pumsb* data set.

shown in this figure, the number of patterns discovered by our algorithm is several orders of magnitude smaller than the number of patterns found by CHARM provided that the h-confidence threshold is sufficiently high. In addition, CHARM has difficulties in identifying patterns when the support threshold is less than or equal to 0.4. However, our technique identifies many patterns with very low support. For instance, we obtain a long pattern containing 9 items with the support 0.23 and h-confidence 94.2%. Recall from Table 1 that nearly 96.6% of the items have support less than 0.4. With a support threshold greater than 0.4, CHARM can only identify associations among a very small fraction of the items. Moreover, the execution time for CHARM is considerably higher than hyperclique miner, as shown in Figure 7. With h-confidence pruning, we can use hyperclique miner to discover hyperclique patterns even at support threshold equal to zero. Finally, when we continue to reduce the h-confidence threshold, the runtime of hyperclique miner goes up. However, it is possible to incorporate h-confidence pruning strategy into the CHARM implementation to further improve the performance of hyperclique miner on dense data sets.

Similar results are also obtained for the pumsb* data set, as shown in figures 8 and 9. For the pumsb* data set, CHARM becomes computationally infeasible at 0.04 support threshold, rather than 0.4 for the pumsb data set. This can be explained by the fact that pumsb* eliminates all items having support greater than 0.8, thus removing a large number of cross-support patterns between the highly frequent items and the less frequent items. Hyperclique miner does not

encounter this problem because it implicitly removes the cross-support patterns.

Finally, we have also conducted our experiments on the UCI dense data sets, including `chess` and `mushroom`. We obtain very similar results as the experiments for `pumsb` and `pumsb*`. Those experimental results are shown in Appendix.

6.3 Scalability of the Hyperclique Miner

We tested the scalability of the hyperclique miner algorithm on the synthetic data sets listed in Table 2. For our scale-up experiments, we set the support threshold to 0.01% and increase the number of items from 10000 to 40000. Figure 10 shows the scalability of hyperclique miner in terms of the number of patterns generated by the algorithm. Notice that without h-confidence pruning, the number of patterns generated grows exponentially and the algorithm breaks down for the data set with 40000 items. With h-confidence pruning, the number of hyperclique patterns generated is more manageable and does not grow quite as fast. Figure 11 shows the execution time for our scale-up experiments. The results of this figure indicate that, without h-confidence pruning, there is a rapid growth in the total execution time as the number of items increases.

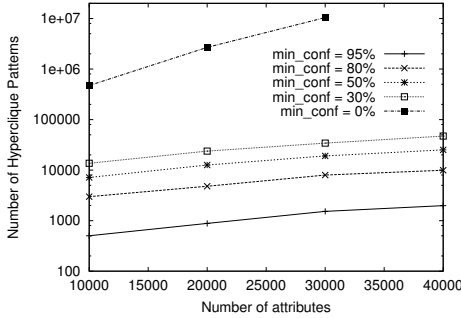


Figure 10: Number of patterns generated by the hyperclique miner with increasing number of items.

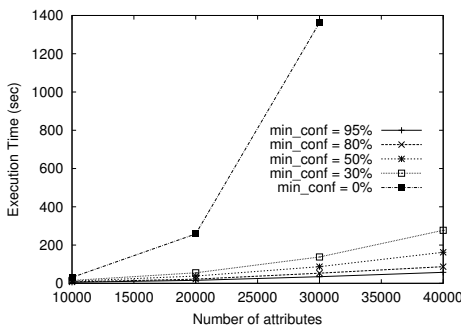


Figure 11: The execution time of the hyperclique miner with increasing number of items.

6.4 Quality of Hyperclique Patterns

In the previous section, we show that hyperclique miner can extract patterns even when the support threshold is set to zero provided the h-confidence threshold is sufficiently large. In this section, we examine the quality of patterns extracted by our algorithm.

Table 4 shows several interesting hyperclique patterns identified at low levels of support from the LA1 data set. It can be immediately seen that the hyperclique patterns contain words that are closely related to each other. For example, the pattern {arafat, yasser, PLO, Palestine} includes words that are frequently found in news articles about Palestine. These patterns cannot be discovered using standard frequent pattern mining algorithms due to their low support values.

Table 4: Hyperclique Patterns from LA1.

Hyperclique patterns	support	h-confidence
{najibullah, kabul, afghan}	0.2%	54.5%
{steak, dessert, salad, sauce}	0.1%	40.0%
{arafat, yasser, PLO, Palestine}	0.4%	52.0%
{shamir, yitzhak, jerusalem, gaza}	0.2%	42.9%
{amal, militia, hezbollah, syrian, beirut}	0.1%	40.0%

Table 5 shows some of the interesting hyperclique patterns extracted from the `retail` data set. For example, we identified a hyperclique pattern involving closely related items such as Nokia battery, Nokia adapter, and Nokia wireless phone. We also found several interesting patterns containing very low support items such as {earrings, gold ring, bracelet}. These items are expensive, rarely bought by customers, and belong to the same product category. Another example is the pattern {jar cookie, canisters 3pc, box bread, soup tureen, goblets 8pc}, which corresponds to a group of china (tableware).

Table 5: Hyperclique Patterns from Retail.

Hyperclique patterns	support	h-confidence
{earrings, gold ring, bracelet}	0.019%	45.8%
{nokia battery, nokia adapter, nokia wireless phone}	0.049%	52.8%
{coffee maker, can opener, toaster}	0.014%	61.5%
{baby bumper pad, diaper stacker, baby crib sheet}	0.028%	72.7%
{skirt tub, 3pc bath set, shower curtain}	0.26%	74.4%
{jar cookie, canisters 3pc, box bread, soup tureen, goblets 8pc}	0.012%	77.8%

We have also evaluated the quality of hyperclique patterns using other objective interestingness measures. Specifically, we use two other measures, Jaccard and cosine, to determine the average affinity of items within a hyperclique pattern. We compute the average affinity in the following way: For each hyperclique pattern $X = \{x_1, x_2, \dots, x_k\}$, we calculate the correlation and Jaccard coefficient for each pair of items (x_i, x_j) within the pattern. The average affinity of a hyperclique pattern is defined as the average pairwise correlation or Jaccard measure of the pairs. Note that this experiment was conducted on Retail data set with the h-confidence threshold 0.8 and the support threshold 0.0005.

Figure 12 compares the average correlation for hypercliques versus non-hyperclique patterns. We sorted the average correlation and displayed them in increasing order. Notice that the hyperclique patterns have extremely high average pairwise correlation compared to the non-hyperclique patterns. This result supports our previous assertion that hyperclique patterns can identify itemsets that contain only tightly-coupled items. A similar result was obtained when using Jaccard as the interestingness measure, as shown in Figure 13. In addition, this experimental result is also agree with the statement in Lemma 5.

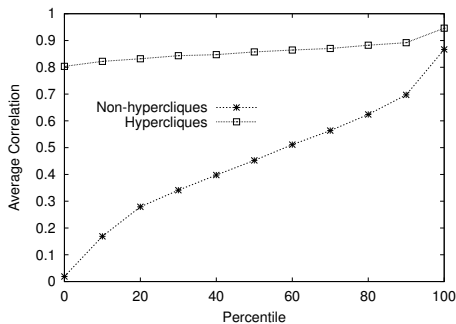


Figure 12: The average correlation of each pair of items for hyperclique and non-hyperclique patterns.

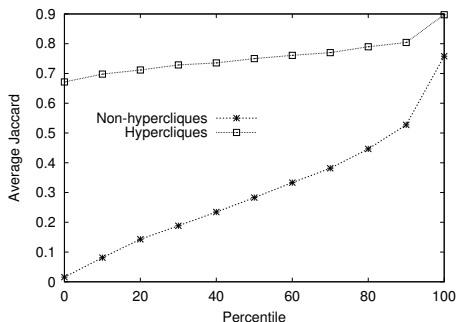


Figure 13: The average Jaccard value of each pair of items for hyperclique and non-hyperclique patterns.

6.5 Hyperclique-based Clustering

We used the S&P 500 index dataset for our clustering experiments. The dataset is similar to the one used by Han et al. [13] in their hypergraph-based clustering paper. For their experiments, they used a minimum support threshold equal to 3% and obtained 19602 frequent itemsets covering 440 of the items. Thus, their initial hypergraph contained 440 vertices and 19602 hyperedges. Our experiment results indicate that the number of frequent itemsets increases to 1387292 and the number of items covered is 914 when we reduce the support threshold to 1%. This creates an extremely dense hypergraph, making it impossible to apply the standard hypergraph clustering algorithm. In contrast, with hyperclique patterns, we do not have any trouble creating hypergraphs at 1% minimum support threshold. With a minimum h-confidence threshold set at 20%, we obtain 10455 hyperclique patterns covering 820 items.

In the hypergraph-based clustering experiments, Han et al. [13] built a hypergraph consisting of 440 vertices and 19602 hypergraphs. This hypergraph was then partitioned into 40 partitions. Out of these partitions, 16 clusters were clean clusters as they contained stocks primarily from one industry group. In our experiment, we built a hypergraph consisting of 820 vertices and 10455 hyperedges. For the purpose of comparison, we partitioned our hypergraph into 80 partitions. In this way, we could make sure that the average size of our clusters was almost the same as theirs. Our experimental results show that we obtained systematically better clustering results. First, 41 out of 80 clusters were clean clusters in our clustering results. Second, many items

Table 6: Clustering of S&P 500 Stock Data

No	Discovered Clusters	Industry Group
1	Baltimore Gas↓, CINergy Corp↓, Amer Electric Power↓, Duke Power↓, Consolidated Edi↓, Entergy Corp↓, Genl Public Util↓, Houston Indus↓, PECO Energy↓, Texas Utilities↓	Power
2	Becton Dickinson↓, Emerson Electric↓, Amer Home Product↓, Johnson & Johnson↓, Merck Co↓, Pfizer Inc↓, Schering-Plough↓, Warner-Lambert↓	health product
3	Bank of New York↓, Bank of Boston↓, CoreStates Financial↓, CIGNA Corp↓, Comerica Inc↓, Aetna Life & Cas↓, Amer General↓, Fleet financial↓, Morgan (J.P.)↓, KeyCorp↓, Mellon Bank Corp↓, NationsBank Corp↓, Natl City Corp↓, Wells Fargo↓, BankAmerica Corp↓	Financial
4	Bell Atlantic Co↑, BellSouth Corp↑, CPC Intl↑, GTE Corp↑, Ameritech Corp↑, NYNEX Corp↑, Pacific Telesis↑, SBC Communication↑, US West Communication↑	Comm.
5	duPont (EI) deNemo↑, Goodrich (B.F.)↑, Nalco Chemical↑, Rohm & Haas↑, Avon Products↑	chemical
6	Federated Dept↑, Gap Inc↑, Nordstrom Inc↑, Pep Boys-Man↑, Sears↑, TJX companies↑, Walmart↑	Retail

with low levels of support could not be included in their clustering scheme. As a result, only 440 items were covered in their clustering results. In contrast, 820 items were included in our clustering results, since our clustering approach can cover items at all levels of support. Third, all clean clusters identified in their approach were present in our clustering results. Finally, for the same clean cluster identified, more items were included in our clusters.

Some of the clean clusters which could not be identified by the hypergraph-based clustering approach from this dataset are shown in Table 6 and the complete list of clusters is given in Appendix. As the table shows, our clustering approach was able to find retail, chemical, health-product, power and communication clusters.

7. CONCLUSIONS

In this paper, we formalized the hyperclique pattern mining problem. The hyperclique pattern is a novel type of association pattern which contains items that are highly affiliated with each other. The hyperclique miner, a new algorithm for mining hyperclique patterns, was presented and analyzed for correctness and completeness. Our experimental results show that the hyperclique mining framework can effectively identify interesting patterns at varied levels of support and can avoid the generation of spurious patterns across different support levels.

There are several directions for future work on this topic. The hyperclique pattern miner algorithm presented in this paper is based upon the Apriori algorithm. It will be useful to explore implementation based upon other algorithm for mining hyperclique patterns, such as TreeProjection [1], CHARM, and FP-growth [15]. There is a potential for using the hyperclique concept in a variety of applications such as dimensionality reduction [9], copy detection [27], and collaborative filtering [11]. Finally, the extension of the hyperclique concept to continuous-valued domains is a challenging task.

8. ACKNOWLEDGMENTS

This work was partially supported by NSF grant # ACI-9982274, DOE contract # DOE/LLNL W-7045-ENG-48 and by Army High Performance Computing Research Center contract number DAAD19-01-2-0014. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by AHPARC and the Minnesota Supercomputing Institute. Finally, we would like to thank Professor Mohammed J. Zaki for providing us the source code for CHARM and Professor Shashi Shekhar for valuable comments.

9. REFERENCES

- [1] R. Agarwal, C. Aggarwal, and V. Prasad. A tree projection algorithm for generation of frequent itemsets. *Journal of Parallel and Distributed Computing*, 2000.
- [2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. of the ACM SIGMOD*, 1998.
- [3] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD*, pages 207–216, May 1993.
- [4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th VLDB*, 1994.
- [5] R. J. Bayardo. Efficiently mining long patterns from databases. In *Proc. of the ACM SIGMOD*, 1998.
- [6] D. Burdick, M. Calimlim, and J. Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Proc. of IEEE Conf. on Data Engineering (ICDE)*, 2001.
- [7] P. Cheeseman and J. Stutz. Bayesian classification (autoclass): theory and results. *Advances in Knowledge Discovery and Data Mining*, pages 153–180, 1996.
- [8] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. Ullman, and C. Yang. Finding interesting associations without support pruning. In *Proc. of IEEE Conf. on Data Engineering (ICDE)*, pages 489–499, 2000.
- [9] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, Jan 2001.
- [10] W. DuMouchel and D. Pregibon. Empirical bayes screening for multi-item associations. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2001.
- [11] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. In *Communications of the ACM*, 1991.
- [12] D. Gunopulos, R. Khardon, H. Mannila, and H. Toivonen. Data mining, hypergraph transversals, and machine learning. In *PODS*, 1997.
- [13] E. Han, G. Karypis, and V. Kumar. Hypergraph based clustering in high-dimensional data sets: A summary of results. *Bulletin of the Technical Committee on Data Engineering*, 21(1), March 1998.
- [14] E. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs. *Proc. of the Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 9–13, 1997.
- [15] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *ACM SIGMOD Intl. Conference on Management of Data*, 2000.
- [16] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.
- [17] A. Jain and R. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1998.
- [18] G. Karypis. hmetis1.5.3. In <http://www.cs.umn.edu/~karypis/hmetis/hmetis/index.html>, 1998.
- [19] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Application in vlsi domain. In *Proceedings ACM/IEEE Design Automation Conference*, 1997.
- [20] B. Liu, W. Hsu, and Y. Ma. Mining association rules with multiple minimum supports. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 1999.
- [21] S. Morishita and J. Sese. Traversing itemset lattices with statistical metric pruning. In *Proc. of ACM SIGACT-SIGMOD-SIGART Symp. on Database Systems (PODS)*, pages 226–236, May 2000.
- [22] R. Ng, L. Lakshmanan, J. Han, and A. Pang. Exploratory mining via constrained frequent set queries. In *Proc. 1999 ACM-SIGMOD*, pages 556–558, June 1999.
- [23] E. Omiecinski. Alternative interest measures for mining associations. In *IEEE TKDE*, Jan/Feb 2003.
- [24] J. Pei, J. Han, and R. Mao. Closet: An efficient algorithm for mining frequent closed itemsets. In *Proc. 2000 ACM-SIGMOD Int. Workshop on Data Mining and Knowledge Discovery (DMKD)*, May 2000.
- [25] C. J. V. Rijsbergen. *Information Retrieval (2nd Edition)*. Butterworths, London, 1979.
- [26] R. Rymon. Search through systematic set enumeration. In *Proc. of Third Int'l. Conf. on Principles of Knowledge Representation and Reasoning*, pages 539–550, 1992.
- [27] N. Shivakumar and H. Garcia-Molina. Building a scalable and accurate copy detection mechanism. In *Proc. of the 3rd International Conference on the Theory and Practice of Digital Library*, 1996.
- [28] P. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *Proc of the Eighth ACM SIGKDD*, 2002.
- [29] C. Yang, U. M. Fayyad, and P. S. Bradley. Efficient discovery of error-tolerant frequent itemsets in high dimensions. In *ACM SIGKDD*, 2001.
- [30] M. Zaki and C.-J. Hsiao. Charm: An efficient algorithm for closed itemset mining. In *2nd SIAM International Conference on Data Mining*, 2002.

APPENDIX

A. THE EXPERIMENTAL RESULTS FOR MUSHROOM AND CHESS DATASETS

B. THE COMPLETE LIST OF CLUSTERS

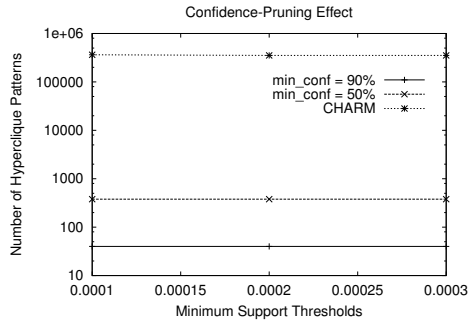


Figure 14: Number of patterns generated by the hyperclique miner and CHARM on mushroom data set.

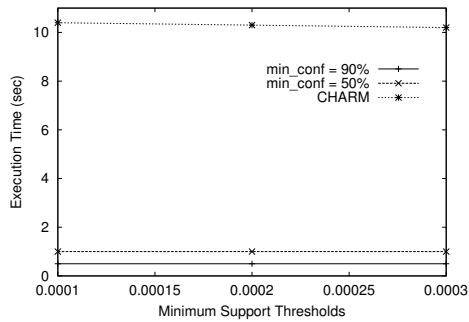


Figure 15: The execution time of the hyperclique miner and CHARM on mushroom data set.

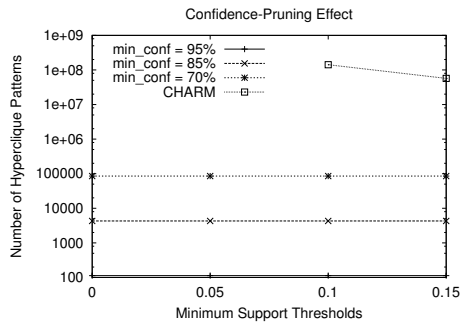


Figure 16: Number of patterns generated by the hyperclique miner and CHARM on chess data set.

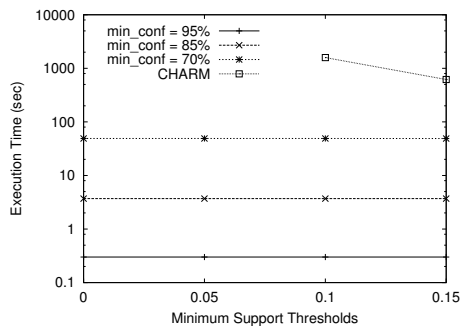


Figure 17: The execution time of the hyperclique miner and CHARM on chess data set.

Nc	Discovered Clusters
1	Baltimore Gas↓, CInergy Corp↓, Amer Electric Power↓, Duke Power↓, Consolidated Edi↓, Entergy Corp↓, Genl Public Util↓, Houston Indus↓, PECO Energy↓, Texas Utilities↓
2	Becton Dickinson↓, Emerson Electric↓, Amer Home Product↓, Johnson & Johnson↓, Merck Co↓, Pfizer Inc↓, Schering-Plough↓, Warner-Lambert↓
3	Bank of New York↓, Bank of Boston↓, CoreStates Financial↓, CIGNA Corp↓, Comerica Inc↓, Aetna Life & Cas↓, Amer General↓, Fleet financial↓, Morgan (J.P.)↓, KeyCorp↓, Mellon Bank Corp↓, NationsBank Corp↓, Natl City Corp↓, Wells Fargo↓, BankAmerica Corp↓
4	Bell Atlantic Co↑, BellSouth Corp↑, CPC Intl↑, GTE Corp↑, Ameritech Corp↑, NYNEX Corp↑, Pacific Telesis↑, SBC Communication↑, US West Communication↑
5	duPont (EI) deNemo↑, Goodrich (B.F.)↑, Nalco Chemical↑, Rohm & Haas↑, Avon Products↑
6	Federated Dept↑, Gap Inc↑, Nordstrom Inc↑, Pep Boys-Man↑, Sears↑, TJX companies↑, Walmart↑
7	Beverly Enterpri↓, ConAgra Inc↓, Community Psych↓, USF&G Corp↓, Genl Signal↓, Louisiana Land↓, Oryx Energy Co↓, Stride Rite↓, Textron Inc↓
8	CPC Intl↓, Gannett Co↑, McGraw-Hill Comp↑, Aon Corp↓, PPG Indus↓, Republic New York↓, Sherwin-Williams↓, Tribune Co.↓, Wrigley↓
9	Central & So. We↓, Loews Corp↓, Philip Morris Co↓, Ohio Edison↓, Public Svc Enter↓, Peoples Energy↓, Providian Corp↓, UST Inc↓
10	BellSouth Corp↓, GTE Corp↓, Ameritech Corp↓, McDonald's Corp↓, Meredith Corp↓, NYNEX Corp↓, Pacific Telesis↓, Rockwell Intl↓, Sysco Corp↓
11	Coastal Corp↓, ENSERCH Corp↓, Hasbro Inc↓, PanEnergy Corp↓, Whitman Corp↓, Williams Cos↓
12	Fleming Cos↓, gannett Co↓, Jostens Inc↓, Knight-Ridder Inc↓, McDonnell dougla↓, Pacific Gas & E1↓, SunTrust Banks↓, Albertson's Inc↓, Walgreen Co↓, Wendy's Intl↓
13	Bally Entertain↓, Boatmen's Bancsh↓, Chubb Corp↓, Intergraph Corp↓, Coca-Coca Co↓, Monsanto Co↓, USAir Group↓
14	Chevron Corp↓, First Bank Syste↓, Mobil Corp↓, Norfolk Southern↓, Potlatch Corp↓, Atlantic Richfie↓, Union Pacific↓
15	Dun & Bradstreet↓, Ecolab Inc↓, Intl Flavors/Fra↓, Harland↓, Minnesota Min'g↓, Quaker Oats↓, Polaroid Corp↓
16	Columbia Gas Sys↓, Echlin Inc↓, FMC Corp↓, Sprint Corp↓, Northrop Grumman↓, Thomas & Betts↓, US West Comm.↓
17	Briggs& Stratto↓, Dover Corp↓, Genl Mills↓, Heinz↓, Hershey Foods↓, Longs Drug Store↓, Sara Lee Corp↓, Snap-on Inc↓, Avery Dennison↓, Winn-Dixie Store↓
18	Consolidate Nat.↓, Eaton Corp↓, Occidental Petro↓, Shoney's Inc↓, Springs Industri↓, TRW Inc↓, Exxon corp↓, Barnett Banks In↓
19	Burlington Resource↓, Amerada Hess↓, Helmerich & Payn↓, Kerr-MCGee↓, Nalco Chemical↓, Phillips↓, SBC Communication↓, Schlumberger Ltd↓, Unocal Corp↓
20	duPont (EI) deNemo↓, Dow Jones& Co↓, Dow Chemical↓, Great Lakes Chem↓, Johnson Controls↓, Air Products & C↓, Sonat Inc↓, Ashland Inc↓, UNUM corp↓, United Technology↓, Seagram Co. Ltd↓, WMX Technologies↓
21	Campbell Soup↓, Carolina Pwr & L↓, Deluxe Corp↓, Gillette Co↓, Amer Intl Group↓, Amer Brands↓, USX-Marathon Grp↓, Avon Products↓
22	Armstrong World↓, First Union Corp↓, Genl Electronic↓, NIKE Inc↓, Procter & Gamble↓, Southern Co↓, St. Paul Cos↓, Transamerica Cor↓, Torchmark Corp↓, U.S. BanCorp↓
23	Bell Atlantic Co↓, Dresser Industry↓, NICOR Inc↓, Golden West Finl↓, Hilton Hotel↓, Hercules Inc↓, AlliedSignal Inc↓, SAFECO Corp↓, Tenneco Inc↓, AllTEL Corp↓, Wachovia Corp↓
24	Bemis Co↓, Columbia/HCA Hlt↓, Dayton Hudson↓, Eastman Kodak↓, Grace(W.R.)↓, Illinois Tool Wo↓, Penney (J.C.)↓, Mattel Inc↓, Millipore Corp↓
25	Crane Co↓, Disney Co↓, Enron Corp↓, Federal Express↓, Fluor Corp↓, Maytag Corp↓, Navistar Intl↓, Ogden Corp↓, Price/Costco Inc↓, Bard(C.R.)↓
26	CUC Intl↓, Engelhard Corp↓, Alexander Alex↓, Lilly↓, Medtronic Inc↓, Rubbermaid Inc↓, TJX Companies↓, WorldCom Inc↓, Woolworth Corp↓, Baxter Internati↓
27	Beneficial Corp↓, Alberto-Culver C↓, Ceridian Corp↓, Honeywell Inc↓, Lincoln Natl Cor↓, Mercantile Store↓, Norwest Corp↓, Northern Telecom↓, Reebok Intl↓, Alco Standard↓, Toys R US↓, Black & Decker C↓
28	Bankers Trust NY↓, Citicorp↓, Chase Manhattan↓, Dean Witter↓, Ahmanson (H F)↓, Federal Natl Mtg↓, Federal Home Loan↓, Great Westn Finl↓, Household Intl↓, Allstate Corp↓, Morgan Stanley G↓, Banc One Corp↓, PNC Bank Corp↓, Salomon Inc↓, Amer Express↓
29	Brown Group↓, Data General↓, AmerGreetings C↓, Fruit of The Loo↓, KMart↓, NACCO Indus Inc↓, Phelps Dodge↓
30	Ball Corp↓, Crown Cork & Sea↓, Echo Bay Mines↓, Homestake Mining↓, Newmont Mining↓, Placer Dome Inc↓, Placer Dome Inc↓, TRINOVA Corp↓, Barrick Gold↓

Nc	Discovered Clusters
31	Harcourt General↓, Interpublic Grp↓, AMP Inc↓, Newell Co↓, Raychem Corp↓, Sun Co↓, At&T Corp↓, Boeing↓
32	Burlington Nther↓, Conrail Inc↓, CSX Corp↓, First Data↓, Harris Corp↓, Pacific Enterpri↓, Parker-Hannifin↓, Timken Co↓, Whirlpool Corp↓
33	Cooper Indus↓, Donnelley (RR) & S↓, EG&G Inc↓, ITT Corp (New)↓, King World Prod'↓, Manor Care↓, Pitney Bowes↓, Perkin-Elmer↓, Ryder System↓, Russell Corp↓, Times Mirror↓, Time Warner Inc↓, Xerox Corp↓
34	ComcastCI'A'Spl↓, Computer Science↓, Cyprus Amax Mine↓, Fleetwood Enterp↓, Inland Steel Ind↓, Nucor Corp↓, Pep Boys-Man↓, PACCAR Inc↓, ASARCO Inc↓, Sigma-Aldrich↓, Stone Container↓, Worthington Indu↓. USX-U.S. Steel G↓
35	Centex Corp↓, Green Tree Finl↓, Gap Inc, Block (H & R)↓, MBNA Corp↓, Lowe's Cos↓, MCI communication↓, Amgen Inc↓, Nordstrom Inc↓, Shared Medical S↓, U.S. Surgical↓
36	Consolidated Fre↓, Harrah's Enterta↓, Human Inc↓, Louisiana Pacifi↓, Novell Inc↓, Sears Roebuck↓, St. Jude Medical↓, Tandy Corp↓, Tele-communic'A'↓, United Healthcar↓, Willamette Indus↓, boise Cascade↓
37	Circuitry city↓, Digital Equipmen↓, EMC Corp↓, Genl Instrument↓, IBM↓, Advanced Micro D↓↓, Southwest↓, Merrill Lynch↓, Apple Computer↓, Motorola↓, Andrew Corp↓, Scientific-Atlan↓, Tektronix Inc↓
38	Computer Assoc Inc↓, Autodes↓, 3Com↓, Compaq Computer↓, cabletron System↓, Cisco Systems↓, DSC Communication↓, HP↓, Intel Corp↓
39	Applied material↓, LSI Logic↓, Micron Technology↓, National Semiconduct↓, Oracle Corp↓, Silicon Graphics↓, Sun Microsystems↓, Tellabs Inc↓, Texas Instrument↓, Bay Networks↓
40	Baker Hughes Inc↓, Boston Scientific↓, Dillard Dept Str↓, Allergan Inc↓, Federated Dept S↓, giddings & Lewis↓, Liz Claiborne↓, outboard Marine↓, WalMart↓, Alza Corp↓
41	Champion Intl↓, Georgia-Pacific↓, Intl Paper↓, Kroger Co↓, Mead Corp↓, Temple-Inland↓, Union Camp↓, Westvaco Corp↓, Weyerhaeuser Co↓
42	Browning-Ferris↓, Cincinnati Milac↓, Cooper Tire& Pac↓, Halliburton Co↓, Harnischfeger In↓, James River Corp↓, McDermott Intl↓, Pulte Corp↓, Pall corp↓
43	Chrysler Corp↓, Caterpillar Inc↓, Cummins Engine↓, Deere & Co↓, Ford Motor↓, Foster Wheeler↓, Genl Motors↓, Home Depot↓, Ingersoll-Rand↓, Kaufman & Broad↓, Limited Inc↓, Marriott Interna↓, Morton Internati↓, AMR Corp↓, Brunswick Corp↓
44	Aluminum Co↓, Bethlehem Steel↓, Delta Airline↓, Dana Corp↓, Goodyear Tire↓, Alcan Aluminium↓, Masco Corp↓, Inco Ltd↓, Praxair Inc↓, Reynolds Metals↓, Tandem↓, Union Carbide↓
45	ComcastCI'A'Spl↑, Cincinnati Milac↑, Centex Corp↑, Fluor Corp↑, Foster Wheeler↑, Masco Corp↑, Outboard Marine↑, PACCAR INC↑, Tele-Communic'A'↑, Baxter internati↑, Brunswick Corp↑
46	Dillard Dept Str↑, Dayton Hudson↑, Penney (j.C.)↑, Limited Inc↑, MCI Communication↑, Price/Costco Inc↑, Praxair Inc↑, WorldCom Inc↑
47	Cooper tire & Ru↑, Eaton Corp↑, Georgia-Pacific↑, harnishchfeger Inc↑, Intl Paper↑, James River corp↑, Mead Corp↑, temple-Inland↑, Union camp↑, Westvaco corp↑
48	Bethlehem Steel↑, Chrysler Corp↑, Caterpillar Inc↑, Dana Corp↑, Deere & Co↑, Ford Motor↑, Genl Motor↑, Inland Steel Ind↑, Ingersoll-rand↑, Nucor Corp↑, Parker-Hannifin↑, USX-US Steel↑
49	Champion Intl↑, Aluminum Co↑, Cyprus Amax Mine↑, Alcan Aluminum↑, Louisiana pacifi↑, Morton Internati↑, Inco Ltd↑, Phelps Dodge↑, Reynolds Metals↑, ASARCO Inc↑, Stone Container↑, Union Carbide↑, Willamette Indust↑, Weyerhaeuser Co↑, boise cascade↑
50	Browning-Ferris↑, Columbia/HCA Hlt↑, CUC Intl↑, Delta Airline↑, Dean Witter↑, HomeDepot↑, Houshold Intl↑, Block (H& R)↑, Allstate Corp↑, MBNA Corp↑, Southwest Airlin↑, Marriott↑, Interna↑, Morgan Stanley↑, AMR Corp↑
51	Boston Scientific↑, Green Tree Finl↑, Harrah's Enterta↑, Humana Inc↑, IBM↑, Lowe's cos↑, medtronic Inc↑, Novell Inc↑, Shared medical AS↑, St. Jude Medical↑, United Healthcare↑, US Surgical↑
52	Computer Assoc I↑, 3Com↑, Compaq↑, Cabletron System↑, Cisco↑, Digital Equipmen↑, DSC communication↑, HP↑, Intel↑, Applied Matreial↑, LSI Logic↑, Motorola↑, Micron Technology↑, Natl Semiconduct↑, Oracle↑, Sun↑, Tellabs↑, Texas Instrument↑, Bay Networks↑
53	CircuitryCity↑, Autodes↑, Computer Science↑, EMC Corp↑, Genl Instrument↑, Advanced Micro D↑, Merrill Lynch↑, Amgen Inc↑, Microsoft↑, Andrew Corp↑, Scientific-Atlan↑, Silicon Graphics↑, Tektronix Inc↑
54	CIGNA Corp↑, Aetna Life↑, Eastman↑, Great Atl & Pac↑, Corning Inc↑, Kroger Co↑, Tandy Corp↑, Amer Express↑, Xerox Corp↑
55	Burlington Nther↑, Conrail Inc↑, CSX Corp↑, Dow Jones↑, Donnelley(RR)↑, Goodyear Tire↑, Allied Signal↑, Alco Standard↑, Tyco Internation↑, Boeing Co.↑
56	Boatmen's Bancsh↑, Comerica Inc↑, First Bank Syste↑, Natl City Corp↑, Peoples Energy↑, Sysco Corp↑, Toys R US↑, Time Warner Inc↑, US Bancorp↑, Wells Fargo↑
57	Bank of Boston↑, Bankers Trust NY↑, Citicorp↑, Chase Manhattan↑, Federal Natl Mtg↑, Ahmanson (H F)↑, Federal Home Loan↑, Great Westn Finl↑, Mellon Bank Corp↑, NationsBank Corp↑, Norwest Corp↑, Bank One Corp↑, PNC Bank Corp↑, Salomon Inc↑, BankAmerica Corp↑

58	Discovered Clusters
58	Bank of New York↑, Beneficial Corp↑, CoreStates Financial↑, Fleet Financial↑, First Union Corp↑, Golden West Finl↑, Amer Intl Group↑, Morgan (J.P.)↑, Lincoln Natl Cor↑, SunTrust Banks↑, Wachovia Corp↑, Barnett Banks In↑
59	Alberto-Culver C↑, Ceridian Corp↑, Echlin Inc↑, Amer Greetings C↑, Federal Express↑, Lilly (Eli)↑, Merck & Co↑, Pfizer Inc↑, ALZA Corp↑
60	Bemis Co↑, Illinois Tool Wo↑, Mercantile Store↑, Pall Corp↑, Tenneco Inc↑, Worthington Indu↑
61	Ball Corp↑, Cooper Indus↑, Crown Cork & Sea↑, Liz Claiborne↑, Polaroid Corp↑, USAir Group↑, Seagram Co. Ltd↑, Woolworth Corp↑
62	Crane Co↑, Dover Corp↑, Grace (W.R.)↑, Harcourt General↑, ITT Corp (New)↑, Mattel Inc↑, Manor Care↑, NIKE↑, Northern↑, Rite Aid↑, Raychem Corp↑, TRINOVA Corp↑, Walgreen Co↑
63	Black & Decker↑, Fruit of the Loo↑, Honeywell Inc↑, Kaufman & Broad↑, King World Prod↑, millipore Corp↑, Pulte Corp↑, Reebok Intl↑, Rockwell Intl↑
64	Baltimore Gas↑, Amer Electric Pw↑, Carolina Pwr & L↑, Central & So↑, DTE Energy↑, Entergy Corp↑, Ohio Edison↑, Pacific Gas↑, PECO Energy↑, Public Svc Enter↑, PP&L Resources↑, Southern Co↑
65	CINergy Corp↑, Consolidate Edi↑, FPL Group↑, Pacific Enterpri↑, PacifiCorp↑, SAFECO Corp↑, AT&T Corp↑, Texas Utilities↑
66	Colgate-palmoliv↑, Houston Indus↑, Knight-Ridder Inc↑, Loews Corp↑, Pitney Bowes↑, Providian Corp↑, Sherwin-Williams↑, Service Corp Int↑, Westinghouse Ele↑
67	Data general↑, Ecolab Inc↑, EG&G Inc↑, Genl Mills↑, Genl Signal↑, Heinz(H.J.)↑, Ralston-Purina G↑, Sara Lee Corp↑, ALLTEL corp↑, Avery Dennison C↑
68	Becton Dickinson↑, Briggs & Stratto↑, Bally Entertain↑, Hilton Hotels↑, Ogden Corp↑, Timken Co↑, Whirlpool Corp↑
69	Bristol-Myers Sq↑, Amer Home Produc↑, Johnson & Johnson↑, Amdahl Corp↑, Schering-Plough↑, Unisys Corp↑, Warner-Lambert↑
70	Beverly Enterpri↑, Cummins Engine↑, Fifth Third Banc↑, Hasbro Inc↑, KeyCorp↑, Minnesota Min'g↑, Navistar Intl↑, NACCO Indus Inc↑
71	First Data↑, Hercules Inc↑, Harris Corp↑, Jostens Inc↑, Meredith Corp↑, Northrop Grumman↑, Rubbermaid Inc↑, Snap-on Inc↑, Thomas & Betts↑, Yellow Corp↑
72	Biomet Inc↑, Disney↑, Allergan Inc↑, Potlatch↑, Perkin-Elmer↑, Ryder System↑, Tandem Computers↑, Whitman Corp↑
73	Alexander & Alex↑, Intergraph Corp↑, Amer Brands↑, Philip Morris Co↑, Air Products & C↑, Republic New York↑, UST Inc↑, V.F. Corp↑
74	Chubb Corp↑, Genl Re Corp↑, Jefferson-Pilot↑, Aon Corp↑, PPG Indus↑, TRW Inc↑, UNUM Corp↑, Union Pacific↑
75	Deluxe Corp↑, Genuine Parts↑, Intl Flavors/Fra↑, Coca-cola↑, Monsanto Co↑, Norfolk Southern↑, Newell Co↑, Stanley Works↑, Textron Inc↑, Automatic Data P↑, Winn-Dixie Store↑, Wrigley (Wm) Jr↑
76	Clorox Co↑, Amer General↑, Emerson Electric↑, Genl Electric↑, Hershey Foods↑, Johnson Controls↑, Luby's Cafeteria↑, St. Paul Cos↑, USLIFE Corp↑
77	Coors (Adolph)C1↑, Armstrong World↑, ONEOK Inc↑, Amer Stores↑, Texaco Inc↑, Wendy's Intl↑, Exxon Corp↑
78	Barrick gold↑, Echo Bay Mines↑, Homestake Mining, Interpublic Grp↑, Newmont Mining↑, Placer Dome Inc↑, PanEnergy Corp↑
79	ENSERCH Corp↑, Mobil Corp↑, Amoco Corp↑, Pennzoil Co↑, Sun Co↑, Ashland Inc↑, Williams Cos.↑, Albertson's Inc.↑
80	Burlington Resource↑, Coastal Corp↑, Chevron↑, Enron↑, Amerada Hess↑, Kerr-McGee↑, USX-Marathon Grp↑, Occidental Petro↑, Phillips Petrole↑, Atlantic Richfie↑, Sonat Inc↑, Unocal Corp↑