Association Rules over Interval Data

R. J. Miller*
Dept of Computer & Info. Science
Ohio State University
rjmiller@cis.ohio-state.edu

Y. Yang
Dept of Computer & Info. Science
Ohio State University
yangy@cis.ohio-state.edu

Abstract

We consider the problem of mining association rules over interval data (that is, ordered data for which the separation between data points has meaning). We show that the measures of what rules are most important (also called rule *interest*) that are used for mining nominal and ordinal data do not capture the semantics of interval data. In the presence of interval data, support and confidence are no longer intuitive measures of the interest of a rule. We propose a new definition of interest for association rules that takes into account the semantics of interval data. We developed an algorithm for mining association rules under the new definition and overview our experience using the algorithm on large real-life datasets.

1 Background

Much work in data mining revolves around the discovery of rules within large quantities of data. Rules over relations are of the form $C_1 \Rightarrow C_2$ where C_1 and C_2 are conditions on tuples of the relation [PS91]. Such a rule may be exact, meaning that all tuples that satisfy C_1 also satisfy C_2, it may be strong, meaning that tuples satisfying C_1 almost always satisfy C_2 or it may be approximate meaning the some of the tuples satisfying C_1 also satisfy C_2 [PS91]. Most often, the conditions are restricted to be simple equality predicates restricting attribute values (e.g., $Quantity = 10$) or conjunctions of such predicates on different attributes.

Rules are typically ranked by some measure of interest. Common measures are based on the frequency with which a rule appears in the relation or, for approximate rules, the strength of the rule implication. Less commonly this measure may be based on the complexity of the rule or other parameters [PS91]. When these measures are numerical, some researchers have referred to the resulting rules as quantitative rules, using the adjective quantitative to refer to the interest measure not to the consideration of quantitative data [HCC93]. Such work should not be confused with the definition of rules over quantitative data which is the problem considered in this paper.

The term *association rule* has been used to describe a specific form of such rules [AIS93, AS94, CNFF96, HS95, MTV94, PCY95, SON95]. The rule frequency is a measure of how often a rule occurs in a data set. When defined as the fraction of all tuples in a relation that satisfy a rule (specifically, $|C_1 \wedge C_2|/|r|$ where r is the set of all tuples considered), frequency is referred to as *support* [AIS93]. The strength of a rule implication is a measure of how often a rule is likely to be true within the data set. When defined as the fraction of all tuples satisfying C_1 that also satisfy C_2 (that is, $|C_1 \wedge C_2|/|C_1|$), rule strength is referred to as *confidence*. The problem of "discovering" association rules is then traditionally defined as follows: given fixed thresholds for the permissible minimum support and confidence, find all association rules that hold with more than the given support and confidence. For clarity, we will refer to this definition of association rules as *classical association rules*. Numerous algorithms have been proposed to solve this problem [AIS93, SON95, AS94, HCC93, HS95, MTV94, PCY95, Toi96]. The main idea behind many of these is the use of the minimum support threshold to limit the space of rules that needs to be searched [AS94]. The algorithms require that the minimum confidence and support be specific *a priori* by a user. The user is given no guidance on selecting the confidence or support thresholds and will not know if a given pair of thresholds will yield no rules or thousands of rules on a given data set.

Algorithms for discovering classical association rules make different assumptions about the type of data sets to be mined. In general, the data set is a relational table. The domains of the attributes may be restricted to boolean domains [AIS93, AS94]. Under this formulation, an association rule $(X_1 = 1) \wedge (X_2 = 1) \wedge ...(X_n = 1) \Rightarrow (Y_1 = 1) \wedge ...(Y_m = 1)$ is often abbreviated as $X \Rightarrow Y$ where $X = \{X_1, ..., X_n\}$ and $Y = \{Y_1, ..., Y_m\}$ are sets of attributes.[1] This definition has been generalized to include relational tables over arbitrary domains including qualitative and quantitative domains [SA96]. It is clear from the formulation of the association rule problem that the complexity of the search depends not only on the number of attributes, but also on the number of values an attribute may have. To manage the added complexity of considering large domains, data values

*This research has been support by the OSU Research Foundation and the Ohio Supercomputer Center.

SIGMOD '97 AZ,USA

[1] Such boolean tables are often represented in an unnormalized form as a list of tuple identifiers paired with a set of values [AIS93]. Logically, each value represents an attribute on which the tuple has the value true.

may be grouped together and considered only collectively. Specifically, a hierarchy may be defined over the values of a domain (for example, a hierarchy of continent-country--region-city may be used to group geographic values). This hierarchy may then be used to reduce the space of rules considered [SA95, HF95]. Additionally, if an attribute is linearly ordered then values may be grouped into ranges. Instead of considering all values of a Salary attribute, values may be partitioned into ranges (for example, ranges of $10,000 increments) [SA96].

These solutions work well for nominal datasets in which the data values represent names with no relative meaning and ordinal data where data values have meaning only in relation to each other. However, we show that these definitions and algorithms may yield very unintuitive results when applied to interval data where the separation between data values has meaning [JD88]. In Section 2, we explore these difficulties. In Section 3, we introduce the notion of an adaptive algorithm for which the quality of the results may be varied dynamically depending on the available memory resources. In Sections 4 and 5, we develop new definitions of association rules that are appropriate for interval data. We present an algorithm for discovering association rules over interval data in Section 6. In Section 7, we briefly overview our experience using the algorithm on large real-life datasets.

The important contributions of our work include:

- The formulation of the association rule problem for interval data in a way that respects the quantitative properties and semantics of the data. We refer to this formulation as *distance-based association rules.*
- Application of adaptive techniques to develop mining algorithms for both classical and distance-based association rules that find rules of the highest quality and interest within given memory constraints.

2 Association Rules on Interval Data

We begin by examining a few examples to illustrate how classical association rules would apply to relations containing interval data. From these examples, we extract three goals that will motivate our algorithms.

As noted in [SA96], when data domains become large, the expense of finding association rules involving all possible values may be prohibitive. Using a higher support and confidence thresholds can reduce this cost but may also give very few rules. Semantically interesting rules may not be found. To address this problem, Srikant and Agrawal have defined what they call *quantitative association rules (QAR)* [SA96]. A QAR is simply a classical association rule in which the predicates may be equality predicates ($Attr = val$) or range predicates ($val1 \leq Attr \leq val2$). To limit the number of ranges (intervals) that must be considered, quality metrics are imposed on what constitutes a "good" interval. These metrics are based on the following observations: if intervals are too large, they may hide rules that exist between portions of the interval; if intervals are too small, they may not have sufficient support so few rules will be found.

Based on this, a measure of *K-partial completeness* was defined to ensure that intervals are neither too big, nor too small with respect to the set of rules they can generate [SA96]. Although designed for quantitative data, this measure uses only the ordinal properties of the data (a qualitative feature) to group adjacent values into intervals [JD88]. The initial partitioning is an equi-depth partitioning where the depth (support) of each partition is determined by the partial completeness level. Hence, the intervals are determined by their relative ordering (and their support). For a depth d, the first d values (in order) are placed in one interval, the next d in a second interval, etc. Quantitative properties of the intervals, including such measures as the relative distance between values, the density of an interval or the distance between intervals are not considered.

	Equi-depth		Distance-based	
Salary	No.	Interval	No.	Interval
18K	1		1	[18K, 18K]
30K	1	[18K, 30K]	2	
31K	2		2	[30K, 31K]
80K	2	[31K, 80K]	3	
81K	3		3	[80K, 82K]
82K	3	[81K, 82K]	3	

Figure 1: Equi-depth vs. distance-based partitioning

This quality measure is very appropriate for ordinal data where there is a linear order defined over the data but the separation between values has no meaning (for example, a domain where (1, 2, 3) is semantically equivalent to (1, 20, 300) [JD88]). However, the same cannot be said when this quality measure is applied to interval data. In Figure 1, we have given an example of an equi-depth partition of a Salary attribute. Intuitively, intervals that include close data values (for example, [81K, 82K]) are more meaningful than intervals involving distant values (for example, [31K, 80K]). In this example, a more natural partition (using our intuitive understanding of the data) would take the distance between items into account when creating intervals as shown in the second column. This is based on our belief that it is less likely that a rule involving the interval [31K, 80K] will be of interest, especially considering that no tuples occupy or support the interior portion of the interval. From this example, we draw the following principle.

Goal 1 *In selecting intervals or groups of data to consider, we want a measure of interval quality that reflects the distance between data points.*

As Srikant and Agrawal point out, their measure of interval quality does not work well on skewed data since it may separate close values that have the same behavior (e.g., participate in the same rules) [SA96]. They suggest the need for developing alternative measures that are based on the range of an interval. In Section 4, we develop an alternative quality measure that is based on a more comprehensive consideration of the quantitative properties of an interval or region.

In addition to influencing the choice of data groupings, the semantics of interval data may influence both the interpretation of a rule (that is, the conditions under which a rule is considered to be true). We illustrate this using the two relations of Figure 2. In both relations, the following classical association rule holds.

$$Job = DBA \wedge Age = 30 \Rightarrow Salary = 40,000 \qquad (1)$$

In both relations, the support of this rule is 50% (three of the six tuples in the relation satisfy the rule) and the confidence is 60% (three of the five 30 year-old DBAs earn

Relation R1				Relation R2			
Id	Job	Age	Salary	Id	Job	Age	Salary
t1	Mgr	30	40,000	s1	Mgr	30	40,000
t2	DBA	30	40,000	s2	DBA	30	40,000
t3	DBA	30	40,000	s3	DBA	30	40,000
t4	DBA	30	40,000	s4	DBA	30	40,000
t5	DBA	30	100,000	s5	DBA	30	41,000
t6	DBA	30	90,000	s6	DBA	30	42,000

Figure 2: Rule definition should reflect distance measure.

40,000). Yet, if we put aside the definition of association rules for a moment and take an intuitive look at the data sets, Relation R2 seems to be a better fit for this rule. Our intuition comes from the fact that within R2, it is more likely that a 30 year-old DBA earns about 40,000. From this intuition, we make two observations that will motivate our work.

First, the formulation of the classical association rule is based on exact set membership. Indeed, it was initially motivated by data sets in which tuples contained simple sets of nominal values from a single unordered domain (also referred to as the boolean association rule problem [SA96]). Although, the confidence measure provides some notion of approximation ("60% of tuples satisfy Rule (1)") it does not provide any room for approximation in data values. In particular, it cannot be used to express a rule such as "30 year-old DBAs earn about 40,000".

Goal 2 *For interval data, a definition of an association rule $C_1 \Rightarrow C_2$ is required that models the following semantics: items in C_1 will be close to satisfying C_2.*

Interpreting Rule (1) in this way, we would like a measure of rule interest that assigns this rule a higher rating in R2 than in R1. In terms of support or more generally rule frequency, the tuples s5 and s6 provide support for the rule (albeit not as much as tuples s2–s4). So the occurrence of the rule is more frequent in R2 compared with R1. Also, the likelihood that a 30 year-old DBA will earn close to 40,000 is higher in R2 than in R1, so we are more confident of the rule in R2. To capture these points, Rule (1) should be assigned both higher support and higher confidence in R2 than in R1. Our final goal (which is closely related to Goal 2) can be stated as follows.

Goal 3 *For interval data, the measures of rule interest, including the measures of rule frequency and strength of rule implication, should reflect the distance between data points.*

We address these three goals in turn in the following sections by developing a series of definitions for the association rule problem culminating in a definition of *distance-based association rules* that meets all stated goals.

3 Adaptive Solution

While numerous algorithms for discovering association rules have been proposed, we give an outline that is common to many of these algorithms [AIS93, AS94].

> **Scan 1** *Scan the data once and count the number of occurrences of each data value (that is, the number of tuples containing the value). Call this set* candidate 1-itemsets. *Set $i = 1$.*
>
> **REPEAT**
>
> > **Prune i** *Discard candidate i-itemsets that have a count less than a threshold value s_0. Call the remaining sets* frequent i-itemsets. *If this set is empty or i is equal to the cardinality of the relation, stop.*
> >
> > **Scan i** *Scan the data and count the number of tuples containing each set of values of size i : $(v_1, v_2, ..., v_i)$ such that all $i-1$ subsets are frequent.*

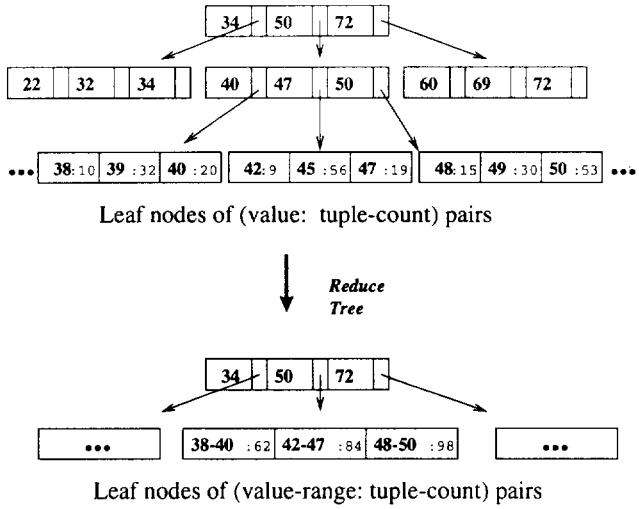

Figure 3: Adaptive summary trees of 1-itemset counts.

The complexity of this algorithm depends on the complexity of the data, but typically even for large data sets the number of passes over the data is not inordinately large. Of course, the hidden cost of this algorithm lies in the size of the memory required to store the counts. If the counts of 1-itemsets do not fit in memory (which may be the case for multi-dimensional data sets over large domains), there may be a significant cost associated with the IO required for the itemset counts.

To reduce the amount of storage, we can group values and only store counts of groups. A group may be a semantic generalization of a set of data values (we can store one count for all cars rather than a separate count for Hondas, Fords, etc.) or an interval of ordered values (ages 20-30) [SA95, HF95]. However, the use of groups reduces the precision of the result. Taking our cue from recent work in clustering of large datasets [ZRL96], we redefine the association rule problem to include the operating constraint that given a limited amount of memory, we would like to find association rules at the finest (most detailed) level possible without dramatically increasing IO cost or response time. This statement of the problem suggests the need for an adaptive algorithm, one that can adjust the level of precision (i.e., group larger sets of values together) as memory becomes scarce. A brief birds-eye view of this idea is depicted in Figure 3. For linearly ordered data, values (and

their counts) are organized in memory in a height-balance tree. A separate tree is maintained for each attribute that can be grouped. As memory gets scarce, the height of the tree is reduced. A simple way of viewing this reduction is to consider each leaf node of the tree as having been replaced by the appropriate summary count in the parent node.

4 Association Rules over Interval Data

To address Goal 1, we propose a more general form of association rule where the predicates considered may include predicates expressing that an attribute or set of attributes must fall within a given subset of values. This set may be defined as an interval on a single attribute as done in [SA96] or it may be an arbitrary region or neighborhood in a multi-dimensional space. Obviously, the cost of considering all possible subsets of data values is prohibitive. We will place restrictions on the properties of the subsets to be considered to ensure that they are interesting with respect to the semantics of the data.

4.1 Clusters

Our examples suggest the need to cluster data and consider association rules between these clusters. For interval data, there is a whole field devoted to the discovery and analysis of data groupings that reflect the relative distances between data points [Eve93]. Clustering techniques vary tremendously in how they use distances to determine groupings and there is no universal definition of what a cluster is or what properties it must have [Eve93].

We adopt a common form of clustering to identify data groups that are compact (the distance between points within a cluster is small) and isolated (relatively separable from other groups). This latter criteria can be formalized by defining the region around the cluster to be relatively sparse. The equi-depth partitioning of Figure 1 fails to meet both of these criteria. The problem of finding good clusters can be formalized as: find a set of K clusters that minimize a given distance metric (such as the average distance between pairs of points in each cluster) [KR90, EKX95, NH94, ZRL96].

Before we give a formal definition, we introduce some notation.

Let $R = \{A_1, A_2, ..., A_m\}$ be a relation schema and r be a relation over R where $|R| = m$ and $|r| = n$. We use the convention that symbols from the end of the alphabet $X, Y, ...$ refer to sets of attributes and symbols from the beginning $A, B, ...$ refer to single attributes.

A cluster is a set of tuples. For a specific set of attributes X, we will place restrictions on the properties of these tuples when projected on X. For this reason, we say that a cluster is "defined on" X and denote the cluster C_X.

A possible quality measure on a one dimensional cluster is the range or smallest interval containing all points [SA96] or on 2 dimensions, the area of the smallest bounding box. However, the area does not reflect the density or coverage of points within a cluster. We therefore have chosen to use a common measure from Statistics, the average pairwise (intra-cluster distance) or diameter of a cluster [KR90, ZRL96]. We use δ_X to denote a distance metric on values in the attribute set X, such as the Euclidean or Manhattan distance.

Dfn 4.1 *The* **diameter** d *on* X *of a set of tuples* $S = \{t_i : 1 \leq i \leq N\}$ *is the average pairwise distance between tuples projected on* X.

$$d(S[X]) = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} \delta_X(t_i[X], t_j[X])}{N(N-1)} \quad (2)$$

In order to use clusters in finding association rules, we will restrict the quality of a cluster using two thresholds on the cluster size and diameter.

Dfn 4.2 *A* **cluster** C_X *defined on a a set of attributes* X *is any subset of* r *that satisfies the following for some* density threshold d_0^X *and* frequency threshold s_0.

$$d(C_X[X]) \leq d_0^X$$
$$|C_X| \geq s_0$$

The number of tuples in the cluster is $|C_X|$ *and the dimension is* $|X|$. •

The first criteria ensures that the cluster is sufficiently dense. The second criteria ensures that the cluster is frequent, i.e., that it is supported by a sufficient number of tuples. To ensure that clusters are isolated from each other, we will rely on a clustering algorithm to discover clusters that are as isolated as possible.

4.2 Rules

We now turn to the issue of how clusters can be used in association rules. We would like a definition of association rules that permits rules such as $X_1 \in C_{X_1}[X_1], ..., X_x \in C_{X_x}[X_x] \Rightarrow Y_1 \in C_{Y_1}[Y_1], ..., Y_k \in C_{Y_y}[Y_y]$. We will always label clusters with the attribute or set of attributes over which they are formed, so without ambiguity we will abbreviate this rule as $C_{X_1}...C_{X_x} \Rightarrow C_{Y_1}...C_{Y_y}$.

If we restrict consideration to one dimensional clusters, we can directly use the definition of quantitative association rules [SA96].

Let A be an attribute of R with quantitative domain D_A. Let $I_R = \{(A, l, u) \mid A$ is an attribute of $R, l \in D_A, u \in D_A$ and $l \leq u\}$. $I_A \in I_R$ is called an interval, by convention we label an interval with the attribute over which it is defined. For $I_X \subseteq I_R$, $X = \{A \mid (A, l, u) \in I_X\}$. The extension of I_X is defined as $\{t \mid u \leq t[A] \leq l$ for all $(A, l, u) \in I_X\}$ and the size of I_X, $|I_X|$ is the size of its extension. Notice that I_X with non-empty extensions can be written as $I_{A_1} \cup ... \cup I_{A_l}$ for some set of distinct attributes A_i and intervals I_{A_i}.

Dfn 4.3 *[SA96] A* **quantitative association rule** *is an implication of the form* $I_X \Rightarrow I_Y$ *where* $I_X \subset I_R$ *and* $I_Y \subset I_R$ *where* $X \cap Y = \emptyset$. •

A rule $I_X \Rightarrow I_Y$ holds with confidence c if $|I_X \cup I_Y|/|I_X| \geq c$. A rule $I_X \Rightarrow I_Y$ holds with support s if $|I_X \cup I_Y| \geq s$.

For this definition, we redefine the problem of finding quantitative association rules as follows. For a given amount of available memory, a given minimum support s_0 and density thresholds d_0^X, find the largest number of intervals that minimize the given distance metric while keeping the running time proportional to the data size. For this set of intervals, find all rules with minimum confidence.

We can generalize this definition to multidimensional clusters by replacing the use of intervals with (the more general) clusters.

Dfn 4.4 *A generalized quantitative association rule is an implication of the form* $C_{X_1}...C_{X_x} \Rightarrow C_{Y_1}...C_{Y_y}$ *where all the* X_i *and* Y_j *are disjoint.* •

A rule $C_{X_1}...C_{X_x} \Rightarrow C_{Y_1}...C_{Y_y}$ holds with confidence c if $|(\cup_i C_{X_i}) \cup (\cup_j C_{Y_j})|/|\cup_i C_{X_i}| \geq c$. The rule holds with support s if $|(\cup_i C_{X_i}) \cup (\cup_j C_{Y_j})| \geq s$.

4.3 Algorithm

We describe an adaptive algorithm for finding generalize quantitative association rules (classical association rules over interval data) that has linear IO cost. The idea is to use a standard clustering algorithm to identify the intervals of interest followed by a standard association rule algorithm to identify rules over those intervals. The algorithm use a single partitioning of the attributes into disjoint sets (X_i) over which there is a meaningful distance metric.[2] Most often each X_i an individual attribute or a small set of closely related attributes over similar domains. As noted in Section 3, we have drawn the initial idea for developing an adaptive association rule algorithm from Birch, an adaptive clustering algorithm [ZRL96, ZRL97]. In addition to being adaptive, Birch has linear IO cost and provides a mechanism for handling outliers that can be modify to help prune clusters with low frequency. We use Birch to find clusters over each attribute. The clusters are created incrementally and represented by a compact summary. The summaries produced in the first phase are then used to form association rules.

4.3.1 Phase I – Identifying Clusters

The basic idea behind Birch is that clusters can be incrementally identified and refined in a single pass over the data. Each cluster is represented by a Clustering Feature (CF) that is a succinct summary of the properties of the cluster. From the CFs of two clusters, the CF of their union and a number of distance metrics (including the diameter of Equation 2) can be derived [ZRL96]. Hence, clusters can be combined and new points added to clusters using only the CFs.

A clustering feature (CF) contains the number of tuples, the linear sum of the tuples and the square sum of the tuples of a cluster. For $C_X = \{t_1, ..., t_N\}$, the CF is defined as:

$$CF(C_X) = (N, \sum_{i=1}^{N} t_i[X], \sum_{i=1}^{N} t_i[X]^2) \quad (3)$$

The clustering process is guided by a height-balanced tree (similar to a B^+-tree) of CF vectors. An internal node contains a list of (CF, pointer) pairs where each CF summarizes all data points in the descendents. Leaf nodes contain lists of CFs. The CF-tree is built incrementally by inserting new data points individually. Each data point is inserted by locating the closest CF at each level of the tree and following the corresponding pointer recursively. At each level, the closest CF is updated to reflect the insertion of the new point. At the leaf, the point is added to the closest cluster, if the diameter of the augmented cluster does not exceed a threshold diameter. Otherwise, a new cluster is created. When leaf nodes are full, they are split and the CF-tree that guides insertion is adjusted, much in the way a B^+-tree is adjusted in response to the insertion of a new tuple.

For each X_i, we select an initial diameter threshold distance metric and $d_0^{X_i}$. As data values are inserted into the CF-tree, if a cluster's diameter exceeds the threshold, it is split. The split may increase the size of the tree. If the memory is full, the tree is reduced by increasing the diameter threshold and rebuilding the tree. The rebuilding is done by re-inserting leaf CF nodes into the tree. Hence, the data or the portion of the data that has already been scanned does not need to be rescanned [ZRL96]. With the higher threshold, some clusters are likely to be merged reducing the space required for the tree.

As the CF-tree is being built, small clusters (outliers) may be paged out to disk. We define outliers to be the clusters that are significantly smaller than the frequency threshold. Since this is done before all data has been scanned, clusters may be wrongly categorized as outliers. Hence, outliers need to be re-inserted into the complete tree to ensure that they are indeed outliers [ZRL96].

4.3.2 Phase II – Combining Clusters to Form Rules

All clusters found in Phase I will satisfy the density threshold. In Phase II, all clusters that also satisfy the frequency threshold (that is, clusters with sufficient support) are used. These clusters form the set of frequent itemsets that are used as input to the *a priori* algorithm [AS94]. If for some X_i there are no frequent clusters, we omit X_i from consideration in Phase II.

Clusters are combined based on the frequency with which they appear together in the dataset. Frequent i-itemsets (for $i > 1$) are identified as follows.

> **Scan i** *Scan the data and count the number of tuples containing each set of values of size i : $(v_1, v_2, ..., v_i)$ where each $v_j \in C_{X_j}$ and all $i-1$ subsets are frequent. Call this set candidate i-itemsets.*
>
> **Prune i** *Discard candidate i-itemsets that have a count less than a threshold value s_0. Call the remaining sets* frequent i-itemsets. *If this set is empty or i is equal to the cardinality of the relation, stop; else increment i and repeat.*

We have defined a cluster to be a set of tuples, but in Birch summaries (rather than tuple sets) are discovered. We therefore require a way to determine the cluster to which a point belongs. For each point, we can find the centroid closest to the point (by using the CF-tree as a search tree) and define the tuple to be in the cluster represented by this centroid.[3] Due to the local and incremental nature of Birch, this cluster may not be the same cluster to which the tuple was assigned when it was originally inserted into the CF-tree. Although we have describe Phase II using the *a priori* algorithm, other classical association rule algorithms may be used.

5 Distance-Based Association Rules

Using clustering, we have proposed a definition of association rules that meets Goal 1. However, this definition, does not meet Goals 2 or 3. Clustering is used to determine sets of dense values in a single attribute or over a set of attributes that are to be treated as a whole. These clusters are then combined with clusters in other dimensions to form rules based on the standard measure of rule support and confidence.

Consider two attributes X and Y that have been clustered individually into two sets of clusters. We have depicted two clusters graphically in Figure 4. When the data set is projected on the X-axis, the tuples of C_X represent a dense cluster of points. Similarly, C_Y represents a dense cluster when all tuples are projected on the Y-axis.

[2] This partitioning is defined by the user given the data semantics. The algorithm is applied to a single partitioning of the attributes.

[3] See Equation (4) in Section 5.

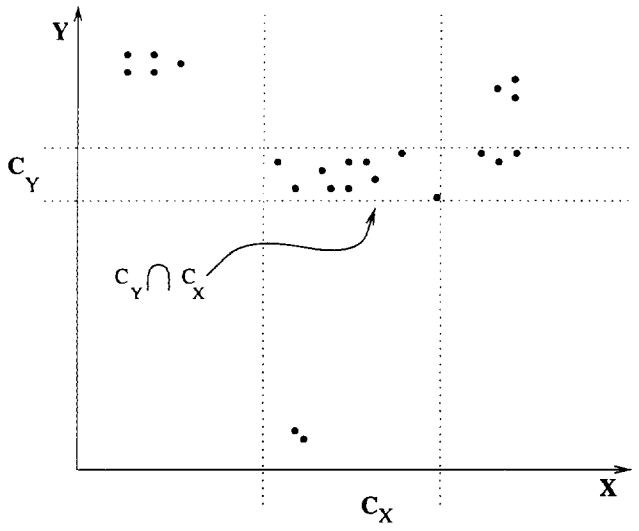

Figure 4: C_X is a cluster on X and C_Y a cluster on Y.

In Figure 4, the confidence of $C_X \Rightarrow C_Y$ interpreted as a classical association rule is 10/12 which is more than the confidence of $C_Y \Rightarrow C_X$ (10/13). Under a distance-based measure, we would want each object in $C_Y - C_X$ to decrease the confidence not by the same amount but by an amount that is proportional to its distance from C_Y. So, while there are more points in $C_Y - C_X$ than $C_X - C_Y$, the former are comparatively closer to the intersection. Hence, the former set $(C_Y - C_X)$ should decrease the confidence less. Also, note that the larger the intersection is, the larger (and more distant) the set $C_Y - C_X$ can be for a given implication strength.

To quantify this, we will use a measure of distance between sets. To ensure that the implication $C_X \Rightarrow C_Y$ is strong, we want to ensure that for all tuples in C_X, their Y values are in C_Y or close to it. Hence, we want the distance from $C_Y[Y]$ to $C_X[Y]$ to be small. This distance measures the *degree of association* between C_X and C_Y.

The distance between two clusters can be defined using any of a number of standard statistical measures, including the average inter-cluster distance or the centroid Manhattan distance [ZRL96]. We will apply these distance measures to clusters projected on specific attributes.

Let $Ci = \{t_j^i : 1 \leq j \leq N_i\}$. The *image of a cluster* Ci on a set of attributes X, denoted $Ci[X]$, is defined as $\{t_j^i[X] : 1 \leq j \leq N_i\}$. (Note that the cluster Ci may be defined on X or on another set of attributes.)

The *centroid* of $Ci[X]$ is defined as:

$$\vec{X0}_i = \frac{\sum_{j=1}^{N_i} t_j^i[X]}{N_i} \tag{4}$$

The *Manhattan distance* $D1$ between the two images $C1[X]$ and $C2[X]$ is defined as:

$$D1(C1[X], C2[X]) = |\vec{X0}_1 - \vec{X0}_2| \tag{5}$$

The *average inter-cluster distance* $D2$ is defined as:

$$D2(C1[X], C2[X]) = \frac{\sum_{i=1}^{N_1} \sum_{j=1}^{N_2} \delta_X(t_i^1[X], t_j^2[X])}{N_1 N_2} \tag{6}$$

These measures and a number of additional distance metrics between clusters are defined in [ZRL96]. We will use D to refer to a distance metric between clusters when we are not making a distinction between specific measures.

For a rule $C_X \Rightarrow C_Y$, the larger the distance between $C_Y[Y]$and $C_X[Y]$ the weaker the implication. We now define a new form of strong association rules over interval data. We begin with the definition of 1:1 rules containing a single cluster in the antecedent and in the consequent.

Dfn 5.1 1:1 Distance-based Association Rule *Let X and Y be disjoint sets of attributes. A* **1:1 distance-based association rule (DAR)** *is a rule of the form R: $C_X \Rightarrow C_Y$ where C_X is a cluster on X and C_Y a cluster on Y. R holds with degree of association D_0 if:*

$$D(C_Y[Y], C_X[Y]) \leq D_0 \quad \bullet$$

A DAR models the semantics that tuples with X values in C_X will have Y values close to C_Y and satisfies Goal 2 of Section 2.

5.1 Connection to Classical Association Rules

Before presenting the general definition of DARs, we consider the connection between our definition and the classical definition which provides further motivation for our work. Consider nominal data over which the following distance metric is defined.

$$\delta(x, y) = \begin{array}{ll} 0, if & x = y \\ 1, if & x \neq y \end{array}$$

Only singleton sets are considered under classical association rules. If we set the diameter threshold to 0, then the only clusters that satisfy this definition must be identical on the value of a given attribute (that is, they are singleton sets).

Theorem 5.1 *A non-empty cluster C_A has diameter 0 iff for some value a, $C_A \subseteq \{t \in r : t[A] = a\}$.* •

Further, if we set s_0 to be the support threshold, then the set of all one dimensional clusters that fall within these thresholds is identical to frequent 1-itemsets under classical association rules. There is also a close relationship between the degree of an association and confidence.

Theorem 5.2 *The classical association rule $A = a \Rightarrow B = b$ holds with confidence c_0 iff the distance-based association rule $C_A \Rightarrow C_B$ holds with degree $1 - c_0$ where $C_A = \{t \in r : t[A] = a\}$ and $C_B = \{t \in r : t[B] = b\}$.* •

Proof Each tuple $t \in C_A$ where $t[B] \neq b$ is a distance 1 from every tuple in C_B. Each tuple $t \in C_A$ where $t[B] = b$ is a distance 0 from every tuple in C_B. From this, we can show that:

$$\begin{aligned} D2(C_B[B], C_A[B]) &= \frac{|C_B|(|C_A| - |C_A \cap C_B|)}{|C_A||C_B|} \\ &= 1 - \frac{|C_A \cap C_B|}{|C_A|} \end{aligned}$$

$\frac{|C_A \cap C_B|}{|C_A|}$ is the confidence of R. Hence, the confidence of R is at least c_0 iff $D2(C_B[B], C_A[B])$ is less than or equal to $1 - c_0$. ⋈

5.2 Higher Arity Rules

We now turn to the application of Definition 5.1 to rules over multiple clusters. Let Y_1 =Stock-Price and Y_2=Time be two attributes of a relation and let $Y = \{Y_1, Y_2\}$. Let X be any set of attributes. To compute $C_X \Rightarrow C_Y$ for any two clusters, we must have a distance metric δ_Y over Y. Even if we cluster on Y_1 and Y_2 individually, to compute $C_X \Rightarrow C_{Y_1} C_{Y_2}$, we must have a distance metric on Y by Definition 5.1.

Although we have assumed that we are working with interval data, we have made no assumptions about the scales used or the relative distance between values in different attributes. We can talk about the distance between points within an attribute A, but it is not clear how to define the distance between points over multiple attributes. Consider two specific attributes defined over the domains Stock-Price and Time. If we wish to apply a distance measure (for example, a Euclidean or Manhattan distance) across these dimensions, it is not clear if Time should be expressed in seconds, hours, days or years or if a non-linear transformation should be used to standardize Time with respect to Stock-Price. To cluster over multiple attributes, the scales must be standardized so that distances in the different dimensions are comparable. There are numerous forms of statistical standardization but their use must be predicated on a detailed understanding of the data and relationships between data in different dimensions [Mil95]. The use of inappropriate standardization techniques may completely distort or destroy the clustering properties of the data. It is our assumption, and indeed one of the basic assumptions of data mining, that such a deep understanding of relative data semantics is not generally available across the full dataset [FPSM91]. Mining is a first step is trying to understand the data.

If a semantically meaningful distance metric across a set of attributes is available, we consider those attributes together and apply clustering to the set of attributes. For example, it may be reasonable to use the Euclidean distance to measure distance across the two attributes Latitude and Longitude or to use a linear function to standardize values in two Salary attributes representing salaries from different years. However, for most attribute combinations, we will not have a meaningful distance metric.

Indeed, this same reason prevents, in general, the application of clustering to arbitrary projections of the data set and the determining of rules from projections of these clusters. Rather than clustering on X and Y separately to find C_X and C_Y, and then trying to combine clusters, a more statistically appealing solution is to apply a clustering algorithm to the data projected on XY to determine clusters within the XY-dimensions. The lower dimensional clusters C_X and C_Y may then be defined as projections of C_{XY} onto X and Y, respectively. However, without a distance metric over XY, this solution is not possible.

To generalize Definition 5.1 to rules that have multiple clusters in the antecedent (R: $C_{X_1}, C_{X_2}, ...C_{X_x} \Rightarrow C_Y$), we need to ensure that the clusters in the antecedent are (collectively) strongly associated with the consequent. Consider the three clusters (which we describe by their bounding intervals): C_A is the interval [41-47] on the attribute years of age; C_D is the interval [2-5] on the attribute number of dependents; and C_C is the interval [10,000-14,000] on the amount of annual insurance claims. We want to understand

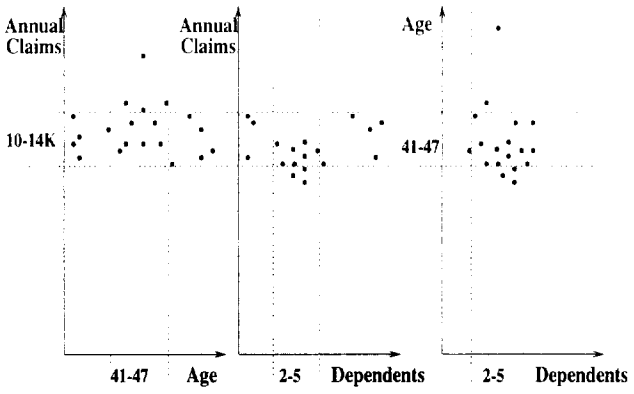

Figure 5: Clusters over Age, Dependents and Claims.

the conditions under which a rule $C_A C_D \Rightarrow C_C$ would hold. Our goal is a definition of association rules that models the semantics that people between the ages of 41 and 47 with 2-5 dependents are likely to have close to \$10K-\$14K of annual insurance claims. For strong rules, we will require that each of C_A and C_D be strongly associated with C_C and in addition, that C_A and C_D occur together. To quantify this, we will again use the distance between clusters and require that the distance between C_A and C_D on both the A and D attributes be small. This will ensure that there are tuples that satisfy (or come close to satisfying) the predicate $Age \in C_A \wedge Dependents \in C_D$. These conditions are depicted graphically in Figure 5. The associate between Age and Claims and between Dependents and Claims is one-way. That is, we are not concerned about whether all people with annual claims between \$10K-\$14K lie within C_A or C_D. However, for C_A and C_D we require that each set be closely associated with the other.

We have replaced the notion of support from classical association rules with a distance based measure. In classical association rules, the threshold support for both a 1-itemset (e.g., [41-47] year-olds) and a 2 or n-itemset (e.g. [41-47] year-olds and [2-5] dependents) is set at the same value. From our practical experience, we have noticed that for higher arity rules, a higher distance threshold (or lower support threshold) is acceptable. We revisit this issue in Section 6. For our definition, we adopt the convention from classical rules of using the same threshold value that was used to define clusters (1-itemsets). We state a more general definition for DARs below for rules from many clusters to a single cluster.

Dfn 5.2 N:1 Distance-based Association Rule
Let $X_1, X_2, ..., X_x$ and Y be pairwise disjoint sets of attributes. A **N:1 distance-based association rule (DAR)** *is a rule of the form R: $C_{X_1}, C_{X_2}, ...C_{X_x} \Rightarrow C_Y$ where each C_{X_i} is a cluster on X_i and C_Y a cluster on Y. R holds with degree of association D_0 if:*

$$\begin{aligned} D(C_Y[Y],\ C_{X_i}[Y]) &\leq D_0 \quad 1 \leq i \leq x \\ D(C_{X_i}[X_i]\ C_{X_j}[X_i]) &\leq d_0^{X_i}, \quad \forall\, i \neq j \end{aligned}$$ •

The degree of association replaces the traditional notion of confidence and the density threshold (on clusters and pairs of clusters) replaces the notion of support satisfying

Goal 3 of Section 2. The frequency threshold (which corresponds to support) is only retained on the initial clusters.

N:1 DARs have a number of important applications. Often in mining, we are interested in discovering which attributes determine one of a specific set of target attributes. For example, in an insurance data set, hundreds of attributes may be recorded about drivers and their characteristics. However, associations between these attributes are not interesting (e.g., whether drivers who earn between 50K and 65K and have 2 dependent drivers on their policy are likely to be close to 45 years of age). Rather, an insurance agent wants to find associations between driver characteristics and a specific variable such as number of accidents or amount of annual claims (e.g., drivers who earn between 50K and 65K and have 2-3 dependent drivers on their policy are likely to have about $10K worth of claims annually).

5.3 Distance-Based Association Rules

To generalize our definition to rules of arbitrary arity (rules of the form, $C_{X_1}C_{X_2}...C_{X_x} \Rightarrow C_{Y_1}C_{Y_2}...C_{Y_y}$), we will require that the clusters in the antecedent collectively occur together and similarly for the clusters in the consequent. We will also require that clusters in the antecedent each be strongly associated with each cluster in the consequent.

Dfn 5.3 Distance-based Association Rule
Let $X_1, ..., X_x, Y_1, ..., Y_y$ be pairwise disjoint sets of attributes. A **distance-based association rule (DAR)** *is a rule of the form R: $C_{X_1}C_{X_2}...C_{X_x} \Rightarrow C_{Y_1}C_{Y_2}...C_{Y_y}$ where each C_{X_i} and C_{Y_i} are clusters on X_i and Y_i respectively. R holds with degree of association D_0 if:*

$$\begin{array}{rcll} D(C_{Y_j}[Y_j],\ C_{X_i}[Y_j]) & \leq & D_0 & 1 \leq i \leq x, 1 \leq j \leq y \\ D(C_{X_i}[X_i],\ C_{X_j}[Y_i]) & \leq & d_0^{X_i} & \forall i \neq j \\ D(C_{Y_i}[Y_i],\ C_{Y_j}[Y_i]) & \leq & d_0^{Y_i} & \forall i \neq j \end{array}$$ •

6 Algorithm

Our algorithm for finding DARs is a modification of the algorithm of Section 4.3. It retains the two phase structure: in the first phase, clusters are identified and in the second phase, clusters are combined to form rules. It again makes use of a single partitioning of the attributes of R into disjoint sets ($R = \{X_1, ..., X_m\}$) over each of which there is a distance metric δ_{X_i}. The criteria for rule formation for DARs is based on the distance between clusters projected onto the same attributes, rather than on counts of set sizes and intersections of those sets. To facilitate the computation of these distances, we have modified the clustering algorithm to maintain additional information about a cluster.

6.1 Phase I – Identifying Clusters

In Birch, summary information about a cluster projected onto the clustering attributes is maintained. We have extended this to maintain additional information about the cluster projected onto other attribute sets. We store summary information in a data structure called an *association clustering feature* (ACF). For $C_X = \{t_1, ..., t_N\}$, the ACF includes the clustering feature (Equation 3) and for all attribute sets Y where $Y \neq X$:

$$\sum_{i=1}^{N} t_i[Y], \sum_{i=1}^{N} t_i[Y]^2 \qquad (7)$$

Like CFs, ACFs may be incrementally maintained. The Additivity Theorem [ZRL96] for CFs may be extended to ACFs. An ACF-tree is a CF-tree with the leaf nodes modified to be ACFs. The internal nodes remain CF nodes. An ACF-tree is maintained for each X_i. The clustering algorithm is unchanged from Birch.

6.2 Phase II – Combining Clusters to Form Rules

The set of clusters C discovered in Phase I that satisfy the frequency threshold are the input to Phase II. In combining clusters to form rules, we use only the ACF vectors of this set. The data is not scanned again.

To aid in our search, we make use of the following graph.

Dfn 6.1 *A* **clustering graph** *$G = (N, E)$ of C contains a node n_c for each cluster $c \in C$. An edge e from n_{c_X} to n_{c_Y} if $D(C_X[X], C_Y[X]) \leq d_0^X$ and $D(C_X[Y], C_Y[Y]) \leq d_0^Y$.* •

We do not require that the frequency thresholds used to define the clustering graph be identical to the frequency thresholds of Phase I. Empirically, we have found that using a more lenient (higher) threshold in Phase II produces a better set of rules.

Theorem 6.1 ACF Representativity Theorem *Given the ACFs of all clusters, the clustering graph can be computed using any of the distance metrics of Section 5.* •

From the clustering graph, we find all maximal cliques. These cliques correspond to large itemsets for DARs. Specifically, a clique and any of its subsets may form the antecedent or consequent of a DAR as per Definition 5.3. The set of all maximal cliques forms a partition of the clusters that covers C (by definition a single vertex is a trivial 1-clique).

Let Q1 and Q2 be two cliques corresponding to the two sets of clusters $C_X = \{C_{X_1}, C_{X_2}, ..., C_{X_x}\}$ and $C_Y = \{C_{Y_1}, C_{Y_2}, ..., C_{Y_y}\}$. To compute rules from these cliques we consider $D(C_{Y_j}[Y_j],\ C_{X_i}[Y_j])$ for $1 \leq i \leq x, 1 \leq j \leq y$. Let $assoc(C_{Y_j}) = \{C_{X_i} | D(C_{Y_j}[Y_j],\ C_{X_i}[Y_j]) \leq D_0\}$. If for any C_{Y_j}, $assoc(C_{Y_j}) = \emptyset$, we omit this cluster from further consideration.

For all $s \subseteq assoc(C_{Y_j})$, $s \Rightarrow C_{Y_j}$ is a DAR. Similarly, if a set of clusters from C_X is in both $assoc(C_{Y_j})$ and $assoc(C_{Y_k})$ then $s \Rightarrow C_{Y_j}C_{Y_k}$.

Let $C_{Y'} \subseteq C_Y$ be any subset of a clique Q2. For each subset $C_{X'} \subseteq C_X$ of the clique Q1, we produce the rule $C_{X'} \Rightarrow C_{Y'}$ if $C_{X'} \subseteq \cap_{C_{Y_j} \in C_{Y'}} assoc(C_{Y_j})$. This process is repeated for all pairs of cliques.

Note that our criteria guarantee that individual clusters have sufficient support under the classical definition. They do not guarantee that the intersections of clusters have sufficient support. If this additional frequency requirement is made then these rules are only candidate rules. In a post processing step, we can rescan the data (once) and count the frequency of all candidate rules.

Reducing the cost of Phase II Image clusters with large diameters (poor density) are unlikely to contribute edges to the graph. Depending on the distance metric used, this can be quantified relative to the size of the cluster. In an initial pass over the ACFs, we can determine if edges from a given node need to be computed, dramatically reducing the number of node comparisons required.

7 Evaluation

We provide an analysis of our DAR algorithm and overview some of our experience using the algorithm on a large, highly multidimensional dataset.

7.1 Complexity analysis

Phase I - CPU Let N be the number of data tuples and M be the number of dimensions of the data set. Phase I of our algorithm needs to scan through N data tuples. Each data tuple may travel at most $log_L N$ steps down the ACF tree, where L is the branching factor of an internal node in the ACF tree. The number of steps for insertion can be bounded by the size of memory allocated for each ACF-tree so the complexity will be much less than $log_L N$. All the above operations are duplicated for each of the M dimensions giving a time complexity of $O(MN(log_L N))$. In addition, the tree may need to be rebuilt with a higher threshold if it becomes too large. The time for the rebuilding can be done in $O(M(log_L N)^2)$ [ZRL96]. So the worst case complexity of Phase I is $O(M(log_L N)^2)$.

Phase I - IO In Phase I, we scan the data set once. If the ACF trees grow too large, the thresholds are adjusted to condense the trees. Hence, a single scan of the data is all that is required beyond the paging out and reincorporation of infrequent clusters. The space allocated for infrequent clusters is a small fraction of the data set size.

Phase II If C is the number of clusters found in Phase I, then our algorithm considers all possible pairs of clusters (over different attributes) to construct the clustering graph. So the number of comparisons is exponential in the number of clusters if the heuristic of not considering images of poor density is not used. Note that Phase II is done entirely in memory using cluster summaries (ACF vectors).

7.2 Performance Results

We present the results of performance experiments designed to test the scalability of the algorithm. In this study, the data set was the Wisconsin Breast Cancer Data (WBCD) obtained from UCI Machine Learning Repository [WM92]. We used a subset of the WBCD data with 500 tuples[4]. We used 30 of the 32 total attributes removing the key (tuple identifier) attribute and the binary outcome attribute. We then studied the performance of the algorithm by increasing the number of points per cluster and proportionally the number of irrelevant (or outliers) points in the data. In this way, we were able to hold the data complexity constant (that is, the number and form of the clusters and rules contained in the data) and study the performance as the data size was increased. All adjustable parameters with the exception of the density and frequency thresholds were fixed.

The frequency threshold is used to select which clusters identified in Phase I are frequent enough to be considered in Phase II. To keep the number of clusters constant as the data size was increased, we set this threshold to 3% of the number of tuples. This experiment was performed on a Sun Sparc 10. The memory limit used in Phase I was set to 5MB. If the memory required to store the ACF trees grew to exceed 5MB, the thresholds were reset and the tree rebuilt using the new thresholds. The running time of Phase I is plotted in Figure 6 for different relation sizes up to a half million tuples. The performance scales linearly with the size of the data set.

We verified that the data complexity was indeed kept relatively constant in the experiment by examining the number of clusters found in Phase I. The number of ACFs found in Phase I over runs ranging from 100K to .5M tuples varied about 5% and was approximately 1050. There was a small difference (typically less that 4%) in the centroid of the clusters due to the use of a non-optimal clustering strategy. This difference grew slightly with the data size.

For Phase II, the number of non-trivial cliques found was approximately 90. The time to identify cliques was roughly constant (about 7 seconds) as expected since the complexity of the data was held constant. While the clustering graph can potentially have an exponential number of edges, in practice we found the number of edges in the graph to be only a small constant times the number of nodes.

To present rules to a user, some description of the clusters is required. A cluster can be described by its centroid, but we have found that this is not the most meaningful description. In our applications, the number of attributes in a partition is typically small. Hence, we have chosen to describe a cluster by its smallest bounding box.

8 Conclusions

We have demonstrated the importance of respecting the semantics of the underlying data when formulating rule mining problems. To this end, we have contributed a new definition of association rule mining that is appropriate for interval data. We have presented an algorithm for finding such rules that adjusts the distance-based quality and rule interest measures based on the available memory. Finally, we presented an overview of our initial experience applying the algorithm to large data sets.

We are currently extending our techniques to consider the mining of rules over mixed variable data including interval and qualitative data. This involves combining the quality and interest measures used for different types of data. We are also extending our performance results to provide a comprehensive study of the sensitivity of our algorithm to different input threshold values and an analysis of the robustness of our techniques.

References

[AIS93] R. Agrawal, T. Imielinksi, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, Washington, DC, May 1993.

[AS94] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, Santiago, Chile, September 1994.

[CNFF96] D. W. Cheung, V. T. Ng, A. W. Fu, and Y. Fu. Efficient Mining of Association Rules in Distributed Databases. *IEEE TKDE*, 1996. To appear.

[4]The full data set contains 569 tuples.

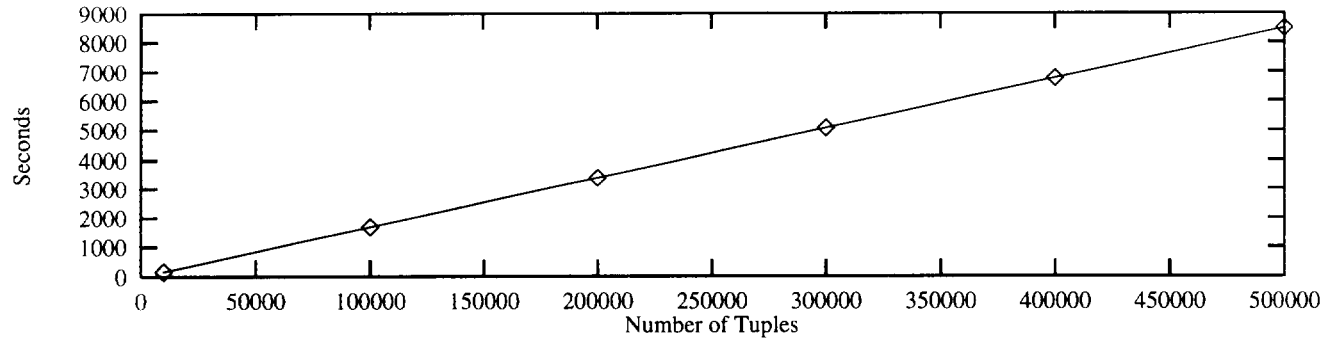

Figure 6: Phase I running time

[EKX95] M. Ester, H.-P. Kriegel, and X. Xu. A Database Interface for Clustering in Large Spatial Databases. In *Proc. of the Int'l Conf. on Knowledge Discovery & Data Mining*, 1995.

[Eve93] B. S. Everitt. *Cluster Analysis*. Edward Arnold and Halsted Press, New York – Toronto, 1993.

[FPSM91] W. J. Frawley, G. Piatetsky-Shapiro, and C. J. Matheus. Knowledge Discovery in Databases: An Overview. In *[PSF91]*, 1991.

[HCC93] J. Han, Y. Cai, and N. Cercone. Data-Driven Discovery of Quantitative Rules in Relational Databases. *IEEE Transactions on Knowledge and Data Engineering*, 5(1):29–40, 1993.

[HF95] J. Han and Y. Fu. Discovery of Multiple-level Association Rules from Large Databases. In *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, Zurich, Switzerland, September 1995.

[HS95] M. Houtsma and A. Swami. Set-oriented Mining of Association Rules. In *Proc. of the Int'l Conf. on Data Engineering*, Taipei, Taiwan, March 1995.

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

[KR90] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, Inc., NY, 1990.

[Mil95] G. W. Milligan. Clustering Validation: Results and Implications for Applied Analyses. In P. Arabie, L. J. Huber, and G. DeSoete, editors, *Clustering and Classification*, pages 345–375. World Scientific Publishing, River Edge, NJ, 1995.

[MTV94] H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient Algorithms for Discovering Association Rules. In *Proc. of the AAAI Workshop on Knowledge Discovery in Databases*, pages 181–192, Seattle, WA, July 1994.

[NH94] R. T. Ng and J. Han. Efficient and Effective Clustering Methods for Spatila Data Mining. In *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, 1994.

[PCY95] J. S. Park, M.-S. Chen, and P. S. Yu. An Effective Hash Based Algorithm for Mining Association Rules. In *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, San Jose, CA, May 1995.

[PS91] G. Piatetsky-Shapiro. Discovery, Analysis, and Presentation of Strong Rules. In *[PSF91]*, pages 229–248, 1991.

[PSF91] G. Piatetsky-Shapiro and W. J. Frawley. *Knowledge Discovery in Databases*. AAAI Press/MIT Press, Cambridge, MA, 1991.

[SA95] R. Srikant and R. Agrawal. Mining Generalized Association Rules. In *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, Zurich, Switzerland, September 1995.

[SA96] R. Srikant and R. Agrawal. Mining Quantitative Association Rules in Large Relational Tables. In *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, Montreal, Canada, 1996.

[SON95] A. Savasere, E. Omiecinski, and S. Navathe. An Efficient Algorithm for Mining Association Rules in Large Databases. In *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, Zurich, Switzerland, September 1995.

[Toi96] H. Toivonen. Sampling Large Databases for Association Rules. In *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, Bombay, India, 1996.

[WM92] W. H. Wolberg and O. Mangasarian. Wisconsin Breast Cancer Database. In P. M. Murphy and D. W. Aha, editors, *UCI Repository of Machine Learning Databases*, Irvine, CA, 1992. University of California, Department of Information and Computer Science. http://www.ics.uci.edu/~mlearn/MLRepository.html.

[ZRL96] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, Montreal, Canada, 1996.

[ZRL97] T. Zhang, R. Ramakrishnan, and M. Livny. Data Clustering System BIRCH and Its Applications. Submitted for publication, 1997.