



**The University of Sydney**

**On Local Pruning of Association Rules  
Using Directed Hypergraphs**

**TECHNICAL REPORT NUMBER 537**

**August 2003**

Sanjay Chawla and Joseph Davis  
School of Information Technologies, University of Sydney

Gaurav Pandey  
ITT Kanpur, CSE Department

**ISBN 1 86487 582 8**

**School of Information Technologies  
University of Sydney NSW 2006**

# On Local Pruning of Association Rules Using Directed Hypergraphs

Sanjay Chawla, Joseph Davis  
University of Sydney  
Knowledge Management Research Group  
Sydney, NSW, Australia 2006  
{chawla,jdavis}@it.usyd.edu.au

Gaurav Pandey  
IIT Kanpur  
CSE Department  
IIT, Kanpur, India 208016  
gpandey@cse.iitk.ac.in

## Abstract

*In this paper we propose an adaptive local pruning method for association rules. Our method exploits the exact mapping between a certain class of association rules, namely those whose consequents are singletons and backward directed hypergraphs. The hypergraph which represents the association rules is called an Association Rules Network(ARN). We propose two operations on this network for pruning rules, prove several properties of the ARN and apply the results of our approach to two popular data sets.*

**Keywords:** Association Rules Pruning, Interestingness, Directed Hypergraphs

## 1 Introduction and Motivation

It is widely recognized that the support-confidence framework for association rule (AR) generation typically results in a large number of rules. Many of these rules turn out to be too obvious or uninteresting from the user/client perspective or redundant. This can often hamper the knowledge discovery process.

The data mining research community has addressed this problem by proposing a number of approaches to producing a parsimonious rule set. The approach based on closed sets is applied during itemset generation [21, 17]. Additional constraints based on interestingness measures are introduced during rule generation [12]. Finally, approaches based on clustering of ARs are introduced at the post mining stage [13, 9]. Each of these methods can be described as "global" in the sense that they compress the rule base  $R$  into a smaller one  $R'$  with the assurance that very little useful information is lost.

However, in practice, the data mining process is rarely carried in isolation without any reference to specific user goals or focus on target items of interest. This calls for an interactive strategy by which pruning of ARs is carried out

in the context of precise goals as the following example will make clear.

Our goal is to understand the itemsets that are frequently associated with the target item, 'income=below 50K'. Let us start with the following rule discovered from the Census Data of Elderly People [12] :

$$r1 : \text{immigrant=no} \rightarrow \text{income=below50k}$$

By itself this rule appears to contradict our general perception, at least in the United States. However by combining this pattern with the rule

$$r2 : \text{sex=female} \wedge \text{age} < 75 \rightarrow \text{immigrant=no}$$

provides a context which helps in interpreting the first rule. This forms a network which flows into the goal(income=below50k) and provides a better explanation of the goal as opposed to when the rule  $r1$  is viewed in isolation.

Now consider a third rule

$$r3 : \text{immigrant=no} \rightarrow \text{sex=female}$$

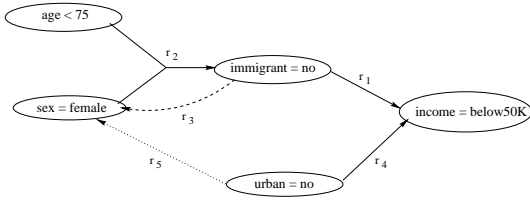
Clearly this rule "flows in the opposite direction", is redundant and does not help explain the goal. Graphically this rule participates in a cycle with rule  $r2$ .

Finally consider two more rules

$$r4 : \text{urban=no} \rightarrow \text{income=below50k}$$

$$r5 : \text{urban=no} \rightarrow \text{sex=female}$$

The rule  $r5$  in conjunction with rules  $r2$  and  $r1$  lead to another (redundant)path to the goal. By pruning rule  $r5$  the goal items remains reachable from the item  $\text{urban=no}$ . Figure 1 captures the preceding discussion. The directed hypergraph after the removal of edges  $r3$  and  $r5$  is called an Association Rules Network (henceforth referred to as ARN). We introduced ARNs in the context of a specific application in [19]. Here we develop this concept further in its full generality.



**Figure 1. A B-graph representing the rules  $r_1 - r_5$ . After the removal of rules  $r_3$  and  $r_5$ , this graph is called an Association Rules Network**

Our approach is based on formalising the intuition behind these common sense observations. It consists of mapping a set of association rules into a directed hypergraph and systematically removing circular paths and redundant and backward edges that may obscure the relationship between the target and other frequent items. This offers the following advantages:

1. The pruning process is adaptive with respect to the choice of the goal node made by the user.
2. Pruning is reduced to cycle and reverse edge detection in the hypergraph.
3. The resulting hypergraph can be transformed into a reasoning network to explain the goal node.

The remainder of this paper is organized as follows: In Section 2 we provide a brief overview of related research followed by an outline of the proposed approach in section 3. Section 4 provides background material on directed hypergraphs and details of association rule network (ARN) and associated algorithms are presented in section 5 and 6. We outline the strengths of ARNs in section 7 followed by some experimental results in section 8.

### 1.1 Problem Definition

The problem that we are addressing in this paper can be succinctly stated as follows:

**Given:** A set of association rules and a target item.

**Find:** A parsimonious set of rules which explain the frequency of the target item.

**Complexity:** The set of rules can be large and may contain many redundant rules which obscure the relationship between the target and other frequent items.

## 2 Related Work

Association Rules Mining is considered a cornerstone of data mining research [2, 5]. As mentioned in the introduction much of the research in association rule mining has focused on algorithms for discovery and more recently, pruning [12, 21, 17].

Liu, Hsu and Ma [14] were the first to propose an integration of classification and association rules. The resulting rules are called class association rules (CARs). The key operation in their approach finds all the rule items of the form  $\langle \text{condset}, y \rangle$  where  $\text{condset}$  is the set of items and  $y$  is a label.

Han, Karypis and Kumar [10] proposed a method of integrating association rules and clustering in an undirected hypergraph. The frequent itemsets were modeled as hyperedges and a min-cut hypergraph partitioning algorithm was used to cluster items.

There has been some theoretical work relating hypergraphs with association rules [20, 8]. Here the relationship between frequent itemset discovery and the undirected hypergraph transversal problem has been noted.

Directed hypergraphs [7, 6] extend directed graphs and have been used to model many-to-one, one-to-many and many-to-many relationships in theoretical computer science and operations research. Directed hypergraphs have also appeared with different names including “labeled graphs” and “And-Or” graphs.

Association rules can be considered as a data induced rule system. To detect *structural errors* like *circularity*, *unreachable goals*, *deadends*, *redundancy* and *contradiction* [15] has modeled the rule base as a directed hypergraph.

It is well known that the standard measures of *support* and *confidence* generate many redundant rules. In general there are three approaches for pruning redundant rules. The first approach exploits the concept of closed-sets [21]. These are maximal itemsets all of whose subsets have the same support. It turns out that the set of all frequent closed itemsets span the set of all frequent itemsets. The second approach is based on the satisfaction of an additional *interestingness* measure in addition to support and confidence. For example, [16] proposed that a rule  $A \rightarrow B$  is interesting if  $\frac{P(A,B)}{P(A)P(B)} > 1$ . Several other statistical measures of interestingness have been proposed. The third approach which will be used to justify removal of cycles from ARNs is based on clustering of rules using a suitable distance measure [9]. Theoretical work on distance measures for categorical data has been presented in [18].

## 3 Outline of Proposed Approach

We briefly describe our method for structuring association rules as a backward directed hypergraph (hence re-

ferred to as B-graphs), pruning it to generate the ARN and transforming it for reasoning. The method consists of four steps which will be expanded in subsequent sections.

**Step A** Given a database  $D$  and the minimum support and confidence we first extract all association rules using a standard algorithm like Apriori [1] or FP-Growth [11].

**Step B** Choose a frequent item  $g$  which appears as a singleton consequent in the rule set and build a leveled B-graph which recursively flows into the goal  $g$ .

**Step C** Prune the B-graph generated in Step B of cycles and reverse edges. The resultant B-graph is called an Association Rules Network(ARN). In Section 6 we will give a formal justification for this step.

**Step D** Find shortest paths between the goal node and nodes at maximal level in the ARN. The set of these path represents the reasoning network for the goal node.

## 4 Preliminaries

In this section we provide background material on association rules and directed hypergraphs [7].

### 4.1 Association Rules

Association rules are generally described in the framework of market basket analysis. Given a set of items  $I$  and a set of transactions  $T$  consisting of subsets of  $I$ , an Association Rule is a relationship of the form  $A \xrightarrow{s,c} B$  where  $A$  and  $B$  are subsets of  $I$  while  $s$  and  $c$  are the minimum support and confidence of the rule.  $A$  is called the *antecedent* and  $B$  the *consequent* of the rule. The support  $\sigma(A)$  of a subset  $A$  of  $I$  is defined as the percentage of transactions which contain  $A$  and the confidence of a rule  $A \rightarrow B$  is  $\frac{\sigma(A \cup B)}{\sigma(A)}$ . Most algorithms for association rule discovery take advantage of the anti-monotonicity property exhibited by the *support* level: If  $A \subset B$  then  $\sigma(A) \geq \sigma(B)$ .

Our focus is to discover association rules in a more structured and dense relational table. For example suppose we are given a relation  $R(A_1, A_2, \dots, A_n)$  where the domain of  $A_i$ ,  $dom(A_i) = \{a_1, \dots, a_{n_i}\}$ , is discrete-valued. Then an *item* is an attribute-value pair  $\{A_i = a\}$ . The ARN will be constructed using rles of the form

$$\{A_{m_1} = a_{m_1}, \dots, A_{m_k} = a_{m_k}\} \rightarrow \{A_j = a_j\} \text{ where } j \notin \{m_1, \dots, m_k\}$$

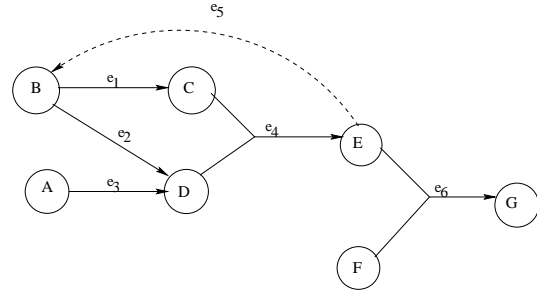
## 4.2 Directed Hypergraphs

A hypergraph is a pair  $H = (N, E)$  where  $N$  is a set of nodes  $\{n_1, n_2, \dots, n_k\}$  and  $E \subset 2^N$  is the set of hyperedges. Thus each hyperedge  $e$  can potentially span more than two nodes. Contrast this with a directed graph where the edge set  $E \subset N \times N$ .

In the context of association rules, each node corresponds to a frequent item and frequent itemsets are mapped to hyperedges.

In a directed hypergraph the nodes spanned by a hyperedge  $e$  are partitioned into two parts, the head and the tail denoted by  $H(e)$  and  $T(e)$  respectively. A hyperedge  $e$  is called *backward* if  $|H(e)| = 1$ . Similarly an edge is called *forward* if  $|T(e)| = 1$ . A directed hypergraph is called *B-directed* hypergraph if all its hyperedges are backward. In the rest of the paper we will refer to them as B-graphs. Thus the set of association rules whose consequents are singletons map neatly into a B-graph. Each rule  $r$  is represented by a hyperedge  $e$ , the antecedents of  $r$  by  $T(e)$  and the consequent by  $H(e)$ .

We will also consider the antecedent of a rule as a single entity. For that we define the notion of a hypernode. Given a B-graph  $B$  with hyperedges  $\{e_1, \dots, e_m\}$ , the hypernodes induced by the hyperedge  $e_i$  are the tail  $T(e_i)$  and the head  $H(e_i)$  considered as a single entity. The set of all hypernodes is denoted by  $V$ .



**Figure 2. An example B-graph with all the major features**

**Example:** As can be seen in Figure 2,  $N = \{A, B, C, D, E, F, G\}$ ,  $E = \{e_1, \dots, e_6\}$  and  $V = \{\{A\}, \{B\}, \{C\}, \{D\}, \{C, D\}, \{E\}, \{E, F\}, \{G\}\}$ . Thus  $F$  is a node but not a hypernode.

We now define a hyperpath and a hypercycle for a B-graph. A hyperpath is defined as a sequence  $P = \{v_1, e_1, v_2, e_2, \dots, e_{n-1}, v_n\}$ , where  $\forall i$ ,  $v_i$  is a hypernode and  $e_i$  is a hyperedge. Furthermore for  $1 \leq i \leq n-2$ ,  $v_i = T(e_i)$ ,  $H(e_i) \in v_{i+1}$ ,  $v_{n-1} = T(e_{n-1})$  and  $v_n = H(e_{n-1})$ . A hyperpath is a hypercycle if  $v_n \in v_0$ .

Again, as can be seen in Figure 2,  $P =$

$\{\{A\}, e_3, \{C, D\}, e_4, \{E, F\}, e_6, \{G\}\}$  is a hyperpath and  $C = \{\{B\}, e_1, \{C, D\}, e_4, \{E\}, e_5, \{B\}\}$  is a hypercycle.

Finally, we define the size of a hyperpath  $|P|$  as the total number of hypernodes appearing on  $P$ . Continuing with the previous example,  $|P| = 4$  and  $|C| = 4$ .

## 5 Association Rules Network

In this section we will formally define an Association Rules Network(ARN), present an algorithm to generate it from a set of association rules and prove some important properties of the ARN.

### 5.1 Definition and Related Concepts

Here we give the definition of an ARN in terms of B-graphs.

**Definition 1** Given a set of association rules  $R$  and a frequent goal item  $z$  which appears as singleton in a consequent of a rule  $r \in R$ . An association rule network,  $ARN(R, z)$ , is a weighted B-graph such that

1. There is a hyperedge which corresponds to a rule  $r_0$  whose consequent is the singleton item  $z$ .
2. Each hyperedge in  $ARN(R, z)$  corresponds to a rule in  $R$  whose consequent is a singleton. The weight on the hyperedge is the confidence of the rule.
3. Any node  $p \neq z$  in the ARN is not reachable from  $z$ .

We now define the notion of a level which will be used to exclude hypercycles from the ARN while retaining reachability to the goal node.

**Definition 2 A** The level of the goal node is zero.

**B** The level of a non-goal node  $v$ , is defined as

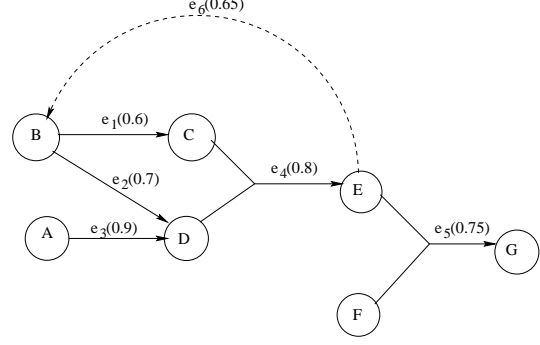
$$l(v) = \min\{l(u) \mid \exists e \text{ such that } v \in T(e) \text{ and } u = H(e)\}$$

**C** The level of a hypernode  $c$  is defined as

$$l(c) = \min\{l(s) \mid s \in c\}$$

**Example:** For the ARN in Figure 3,  $l(G) = 0$ ,  $l(F) = l(E) = 1$ ,  $l(C) = l(D) = 2$  and  $l(B) = l(A) = 3$ . For hypernodes,  $l(\{C, D\}) = 2$  and  $l(\{E, F\}) = 1$ .

**Definition 3** A hyperedge  $e$  in an ARN is called a reverse hyperedge if  $l(T(e)) < l(H(e))$ .



**Figure 3.** An example ARN.  $e_6$  is not a part of the ARN because it is a reverse hyperedge and also participates in a hypercycle.

**Example:** In Figure 3,  $e_6$  is a reverse hyperedge.

**Note:** We now introduce two conditions which will prevent redundancies (hypercycles and reverse hyperedges) from appearing between hypernodes at different levels while preserving the reachability to the goal node in an ARN.

**Condition 1** A node which has served as a consequent during ARN generation cannot be an antecedent across levels for a rule whose consequent is at a higher level.

**Condition 2** The goal attribute can appear with only one value in the ARN, namely the one in the goal node.

### 5.2 ARN Construction Algorithm

We now describe a breadth-first like algorithm to construct an ARN from a rule set  $R$  and a goal node  $c$ .

Algorithm ARNConst(shown below) takes as input the rule set  $R$  and a goal node  $c$  which appears as singleton consequent in  $R$ . The  $Rules.getRules(R, u, s)$  is a function that returns all rules in  $R$  whose consequent is  $u$  but antecedent does not contain  $s$ . For each of these rules,  $Rules.getAntecedents(u)$  returns the set of all antecedents. The level of each these antecedent elements is determined on the basis of *Condition 1* and *Definition 2*. For all the rules satisfying *Condition 1* a hyperedge is added to the ARN. Finally hypercycles between hypernodes which are on the same level are removed using  $Rules.removeLevelCycles()$ . The algorithm returns the generated ARN.

### 5.3 Results about ARN Construction Algorithm

**Theorem 1** The time complexity of ARN generation is  $O(nk)$  where  $n$  is the number of frequent items and  $k$  is the number of rules whose consequents are singletons.

```

Data : Rules  $R$ , Consequent  $c$ 
Result : A Directed hypergraph  $G$  representing an
AR Network which flows into  $c$ 
 $visited[i] = 1$  if  $i$  has been visited as a consequent;
 $u \leftarrow c$ ;
 $s \leftarrow c$ ;
Add  $u$  to queue  $q$ ;
 $visited[u] \leftarrow 1$ ;
repeat
   $RuleSubset \leftarrow Rules.getRules(R, u, s)$  /* Get
all rules whose consequent is  $u$  but antecedent
does not contain  $s$  */;
  for all rules  $r \in RuleSubset$  do
     $a \leftarrow Rules.getAntecedents(u)$ ;
     $level[a] = \text{inf}$ ;
    for all elements  $w \in a$  do
      if  $level[w] < level[a]$  then
         $level[a] = level[w]$ ;
      end
    end
    if  $level[a] \geq level[u]$  then
      for all elements  $w \in a$  do
        if  $visited[w] = 0$  then
          Add  $w$  to  $q$ ;
           $visited[w] = 1$ ;
           $level[w] = level[u] + 1$ ;
        end
      end
       $G.addHyperEdge(r, u)$  /* directed hy-
peredge flowing into  $u$  */;
    end
  end
  if  $q$  is empty and  $G$  is singleton then
    return  $G = \emptyset$ ;
  else if  $q$  is empty and  $G$  is not singleton then
     $G.removeLevelCycles()$  /* remove
level hypercycles based on confidence */;
    return  $G$ ;
  end
  Delete  $u$  from  $q$ ;
until false;

```

**Algorithm 1:** Association Rule Network(ARN) constructed from a rule set  $R$  and flowing into consequent  $c$  using a breadth-first like strategy.

**Proof:** For each node in the ARN the hyperedges flowing into it are found by searching the set of all rules whose consequents are singletons. Furthermore the number of nodes appearing in the ARN is bounded by the number of frequent items. Hence the complexity is  $O(nk)$ . Also  $k$  is bounded by  $\sum_{i=2}^l n_i * i$  where  $l$  is the length of the longest frequent item set and  $n_i$  is the number of frequent itemsets of size  $i$ .

**Theorem 2** *The ARN generated is unique, i.e., it does not depend upon the sequence in which the nodes are explored.*

**Proof:** Let  $u$  and  $v$  be two nodes at the same level. We want to show that by exploring  $u$  and  $v$  in different orders we are not losing an edge whose consequent is one of them and the antecedent is the other.

WLOG assume  $u$  is explored before  $v$ . Let  $\{e_1, \dots, e_{uv}\}$  be the set of edges for which  $u$  is a consequent and  $v$  is one of the antecedents. Then by the definition of level,  $level(T(e_i)) = level(v)$  for all  $1 \leq i \leq uv$ . Note that there cannot be any edge whose consequent is  $u$  and any of whose antecedents have level less than  $u$  (which is the same as the level of  $v$ ).

Now, once we explore  $v$ , all the edges whose consequent is  $v$  and whose antecedents contain  $u$ , are free to flow into  $v$  because of Condition 1. Thus we have not lost any hyperedges between  $u$  and  $v$  since they are at the same level. A similar argument holds when  $v$  is explored before  $u$ . Thus the set of hyperedges and hypernodes remains the same in both cases.

Hence same ARN will be generated irrespective of the order in which the nodes at the same level are explored.

**Theorem 3** *The goal node is reachable from any node in the ARN.*

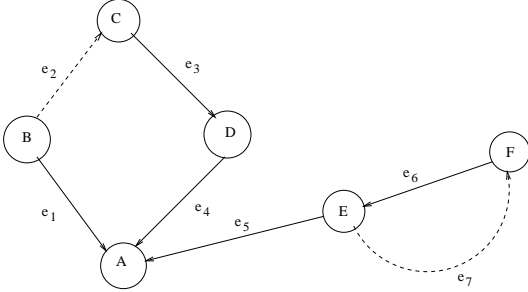
**Proof:** By induction on level. By definition the level of the goal node is zero and trivially reachable from itself. Assume that the goal node is reachable from any node at level  $i$ . For each node  $u$  at level  $i + 1$  there exists atleast one hyperedge  $e$  such that  $l(H(e)) = i$  and  $u \in T(e)$ . Thus the result holds for all  $i + 1$  and hence for all levels  $l < \max\{l(u) | u \in N\}$ .

**Theorem 4** *The ARN generated for the algorithm is free of cycles across levels*

**Proof:** By contradiction. Let  $C$  be a hypercycle across levels, i.e.,  $C = \{v_0, e_1, \dots, e_n, v_n\}$  where  $v_n \subset v_0$ . Let  $v_i$  be the first edge in  $C$  where  $l(v_i) < l(v_0)$ . Then by construction of ARN,  $l(v_{n-1}) < l(v_i)$ . Therefore  $l(v_{n-1}) < l(v_0)$ . Now  $l(v_n) \geq l(v_0)$  because  $v_n$  is a singleton and  $v_n \subset v_0$ . Therefore  $e_n$  is a hyperedge from an antecedent at a lower level to a consequent at higher level. This contradicts condition A.

## 6 Cycle removal and rule pruning

In this section we provide a justification for the removal of hypercycles and reverse hyperedges. We will show that in either case the information provided by the hyperedge pruned, or the *interestingness* of the corresponding rule, is small, given the rest of the ARN.



**Figure 4. An example which distinguishes reverse hyperedges ( $e_2$ ) and hypercycles ( $e_7$ ).**

Consider the example in Figure 4. Given the goal node  $A$ , during ARN construction, the two hyperedges  $e_2$  and  $e_7$  are removed. The removal of hyperedge  $e_7$  is more “crucial” than  $e_2$  because it is part of a hypercycle; otherwise  $A$  will never be reachable from  $F$  violating the definition of the ARN. On the other hand, removal of the reverse hyperedge  $e_2$  is justified because it is part of a redundant path from  $B$  to  $A$ . Hence removal of  $e_2$  will not destroy the reachability condition of the ARN. This illustrates the difference between the two kinds of pruning operations. We provide separate arguments for each of the two operations.

### 6.1 Removal of Hypercycles

We first note that a hypercycle  $\mathcal{H}$  in a B-graph cannot be of size less than three (see Section 4.2). We will provide the desired justification for two cases, one in which size of  $\mathcal{H}$  is three and the other in which size of  $\mathcal{H}$  is four. From transitivity, it follows that the removal of any hypercycle of size bigger than four can be justified using the arguments presented for the second case. We will now consider the two cases.

**CASE 1:** Consider two rules of the form  $A \rightarrow B$  and  $B \rightarrow A$ . If  $A$  and  $B$  are the same level then WLOG remove the hyperedge with lower confidence to break the hypercycle. Else WLOG assume  $level(A) > level(B)$ . Since we are exclusively dealing with rules whose antecedents are a conjunction of items and the consequent is a singleton item, for a vast majority of these rules,  $P(A) < P(B)$ . In such a scenario,

$$conf(A \rightarrow B) > conf(B \rightarrow A)$$

This justifies the removal of the hyperedge representing  $B \rightarrow A$ .

**CASE 2:** Consider three rules of the form  $A \rightarrow B, B \rightarrow C$  and  $C \rightarrow A$ . Our argument is based on the concept of information gain. Assume we have an information channel  $I$  which is a sequence of transactions. Then given that the pair of itemsets  $(A, B)$  and  $(B, C)$  are frequent, i.e., they appear close to each other with high probability then the information that  $(A, C)$  is frequent is not surprising. We formalize this argument as follows.

**Definition 4** Let  $F$  be the set of all itemsets. Let  $d : F \times F \rightarrow [0, 1]$  be defined as

$$d(A, B) = 1 - \frac{P(A, B)}{P(A) + P(B) - P(A, B)}$$

where  $A, B \in F$  and  $P(A)$  is the support of the itemset  $A$ .

**Theorem 5** The function  $d$  is a metric on the space  $F$ .

**Proof:** The function  $d$  is identical to the distance measure  $\rho$  defined in [18]. This proof follows directly from the proof of Lemma 3.2 in [18].

**Corollary 1** For  $0 < \delta \ll 1$ , the information gain from observation that the pair  $(A, C)$  are close to each other is small given that  $d(A, B) < \delta$  and  $d(B, C) < \delta$ .

**Proof:** Follows directly from the triangle inequality of  $d$ .

Now WLOG assume that the  $level(A) > level(C)$ . The fact that  $d(A, C)$  is small given that  $d(A, B)$  and  $d(B, C)$  are small is an indication of the fact that the rule  $A \rightarrow C$  and  $C \rightarrow A$  can be derived from the rules  $A \rightarrow B$  and  $B \rightarrow C$  which are already in the ARN. Thus  $C \rightarrow A$  can be safely pruned without violating the reachability constraint of the ARN.

### 6.2 Removal of Reverse Hyperedges

The removal of reverse hyperedges in an ARN can be justified on the basis of the fact that they generate redundant paths from a node to the goal. This can be formally proved as follows.

**Theorem 6** Let  $a$  be a node in the ARN from which a reverse edge  $e$  originates. Then, there exists a path  $P$  from  $a$  to the goal node such that  $e \notin P$  and the size of  $P$  is smaller than the size of the path from  $a$  to the goal node in which  $e$  participates.

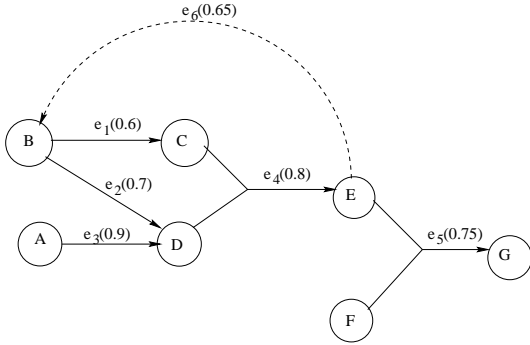
**Proof:** Let  $l(a) = l_1$  and  $l(H(e)) = l_2$ . Since  $e$  is a reverse hyperedge,  $l_1 < l_2$ . We observe that the path of smallest size from any node at level  $l$  in the ARN to the goal node is of size  $l + 1$  (follows from the definition of level). Let this path of smallest size from  $a$  be  $P_1$  and the one from  $H(e)$  be  $P_2$ . Clearly, the size of  $P_1$  is  $l_1 + 1$  and that of the new path  $b, e \cup P_2$  is  $l_2 + 2$ . Thus  $P_1$  is the required path.

This theorem justifies the redundancy of the rule represented by a reverse hyperedge and hence its removal.

## 7 Benefits of ARN

In the introduction we raised the question of the utility of association rules beyond simple exploratory analysis. ARNs also are a tool for exploratory analysis but they provide a context for understanding and relating the discovered rules with each other. ARN offers the following benefits.

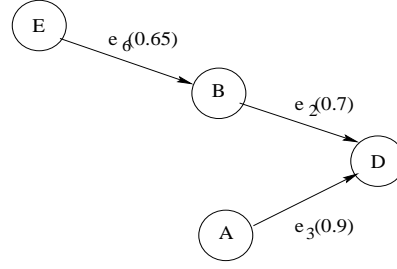
**Local Pruning** ARNs provide a graphical method to prune rules by associating redundant rules with hypercycles and reverse hyperedges. Furthermore the pruning takes place in the context of a goal node. Thus a rule which is redundant for a particular goal node may become relevant for another goal. This is more flexible than pruning based on statistical measures of interestingness.



**Figure 5. ARN with goal node  $G$ . The edge  $e_6$  is not part of the ARN because it participates in a hypercycle**

Consider the B-graph in Figure 5 and Figure 6. When the goal node is  $G$ , the edge  $\vec{E}B$  is a redundant rule which may be eliminated. On the other hand when the goal node is  $D$  the same edge  $\vec{E}B$  is relevant. Thus pruning of a rule as per our notion becomes dependent upon the context of the goal node. We refer to this kind of pruning as *local pruning* and it may be more flexible than global pruning based simply on measures of interestingness.

**Reasoning using Path Traversal** An ARN is a weighted hypercycle free B-graph. Hyperpaths which lead to the



**Figure 6. ARN with goal node  $D$ . The edge  $e_6$  is now part of the ARN. This illustrates the adaptive nature of local pruning**

goal node can be interpreted as providing an explanation for the goal node.

Formally let  $V_{max}$  be the set of all maximum level nodes in the ARN. For each  $v \in V_{max}$  let  $P_v$  be the set of all hyperpaths from  $v$  to the goal node  $g$ . We can define two cost measure on each hyperpath  $p$

1.  $Weight(p) = \sum_{e_i \in p} \log(conf(e_i))$
2.  $Info(p) = - \sum_{e_i \in p} conf(e_i) \log(conf(e_i))$

where  $e_i$  is a hyperedge and  $conf(e_i)$  is the confidence of the rule represented by  $e_i$ .

The reason for introducing two cost functions is that they provide different kinds of information depending upon the context. For example  $Weight(p)$  can be interpreted as the strength of the correlation between the source and the goal node. Similarly  $Info(p)$  can be interpreted as the total information gain along the path from the source to the goal node.

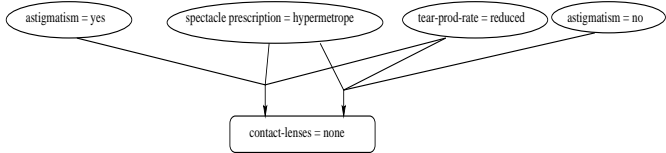
Now the optimal path in  $P_v$  using  $Weight(p)$  or  $Info(p)$  is the best explanation for dependence of the goal node on  $v$ . Computing these optimal hyperpaths for all  $v \in V_{max}$  provides a reasoning network for the goal node  $g$ . The problem of optimal hyperpaths in B-graphs has been studied by [3] where they have reported an algorithm of time complexity  $O(|H| + n \log n)$  where  $|H|$ , size of the hypergraph is  $\sum_{e_i \in E} (|T(e_i)| + 1)$ .

## 8 Applications

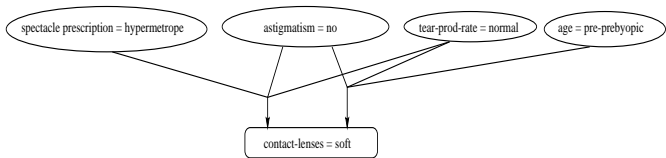
We now show the advantage of our approach by constructing ARNs for two well known data sets. The results, as we will describe below, vindicate our original thesis that a network of association rules provides a context for interpreting the rules in a more coherent fashion than if they were viewed in isolation. The two data sets that we considered were the Lens and the Mushroom data bases which are part of the UCI machine learning repository [4].

## 8.1 Lens Database

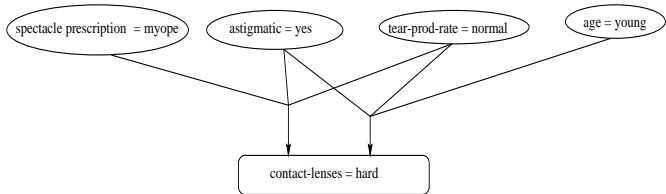
The Lens database has twenty four rows where each row has five attributes which are either binary or ternary-valued. Our goal attribute was `contact-lenses` which denotes whether a person should be fitted with either one of two kinds of lenses: `hard` and `soft`, or neither.



**Figure 7. ARN with goal node**  
`contact-lenses=none`



**Figure 8. ARN with goal node**  
`contact-lenses=soft`



**Figure 9. ARN with goal node**  
`contact-lenses=hard`

We make the following observations based on the three ARNs in Figures 7, 8 and 9.

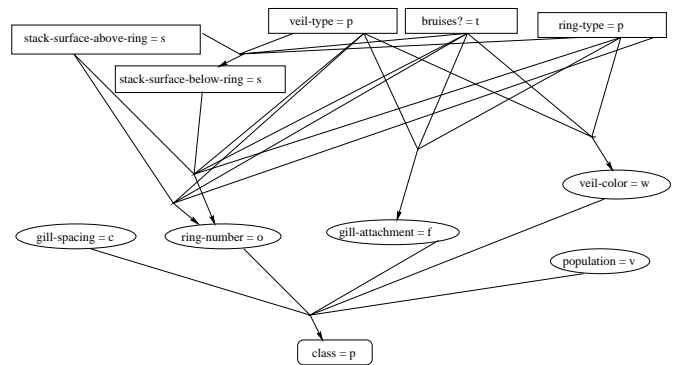
1. As we had mentioned in the introduction, an ARN provides a context for interpreting the rules. A change in the value of the goal attribute is correlated with changes in several antecedents. The ARNs clearly capture these simultaneous changes.
2. Also notice that the structure of the ARN does not change drastically when the value of the goal attribute changes. This suggests that the actual network is at the variable (type) level rather than at the instance level.

3. We finally note that when the goal value is `contact-lenses=none`, `spectacle prescription=hypermetrope` and `tear-production-rate=reduced` participate with different in two distinct three-itemsets. This suggests a strong correlation between them and the goal value.

The observations made are consistent with the expected benefits of using an ARN. This justifies the applicability of an ARN for synthesizing association rules.

## 8.2 Mushroom Database

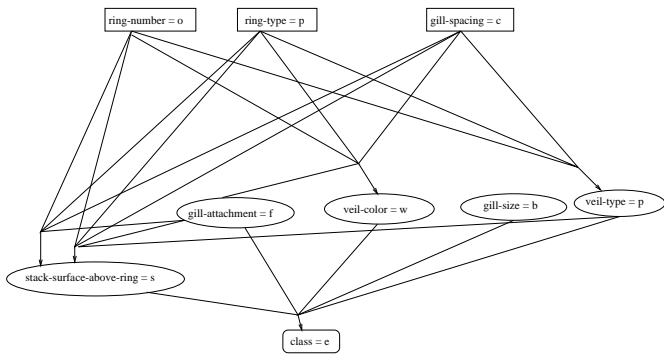
We also constructed an ARN for the Mushroom database. This data set has approximately eight thousand instances and twenty three attributes. We chose the binary attribute `class=edible`, `poisonous` as our goal attribute. The ARNs generated are shown in Figure 10 and 11 for classes `poisonous` and `edible` respectively. However, to save space, we have abbreviated the attribute values by their first letters. For example  $p = \textit{poisonous}$  and  $e = \textit{edible}$ .



**Figure 10. The ARN when the goal attribute is**  
`class=edible`

The observations that we made for the Lens database are applicable here as well. However since the database is bigger we were able to observe more interesting patterns which we now discuss.

1. The set of contributing attributes for the two goal values are mostly the same but some attributes contribute to one and not to the other. More particularly, `gill-size` is present in the ARN for `class=edible` but not in the one for `class=poisonous`. Similarly, `bruises?`, `population` and `stack-surface-above-ring` appear in the ARN for `class=poisonous`, but not for `class=edible`.



**Figure 11. The ARN when the goal attribute is `class=poisonous`**

2. Notice how the item `bruises?=true` appears as a level two item in the ARN for `class=poisonous`. This clearly highlights the advantage of an ARN because it reveals that even though `bruises?=true` does not appear as an antecedent in a rule in which the consequent is `class=poisonous`, it seems to have an important influence in determining the value of the `class` attribute. In other words this shows the transitivity in association rules which the ARNs are able to capture.
3. The level of certain items changes depending upon the value of the goal attribute. For example `ring-number=one` appears as a second level node in the ARN for `class=edible` but as a first level node in that for `class=poisonous`. This change in the level of the item reflects the change in its correlation with the value of the goal attribute.
4. Also notice that there are several paths from the higher level nodes to the goal node in both the ARNs. By using the measures described in Section 7 we can choose the optimal paths between the maximal level nodes and the goal node. This will create the reasoning network and sparsify the graph.
5. Finally, these ARNs reveal the utility of local pruning as the resulting rules are semantically meaningful. Thus the two operations, hypercycle and reverse hyperedge removal, result in a meaningful network.

The observations that we have highlighted above illustrate the benefits of an ARN.

## 9 Conclusions and Future Work

Association Rules Network provides a mechanism for synthesizing association rules in a structured manner. The

important features of an ARN are (1) their ability to prune rules in the context of a goal, (2) the pruning mechanism is based on simple graph operations and (3) the ARN can serve as a basis of reasoning with discovered rules.

For future work we would like to convert our intuition about reasoning using ARNs into a more theoretical framework. We would also like to design a layout algorithm specifically for ARNs. We are also working on a cycle-detection algorithm for general B-graphs which can be applied to ARNs.

## References

- [1] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.
- [2] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.
- [3] Giorgio Ausiello, Giuseppe F. Italiano, and Umberto Nanni. Hypergraph traversal revisited: Cost measures and dynamic algorithms. *Lecture Notes in Computer Science*, 1450, 1998.
- [4] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.
- [5] L. Feng, J. Yu, H. Lu, and J. Han. A template model for multi-dimensional, inter-transactional association rules. *VLDB Journal*, 11(2):153–175, 2002.
- [6] G.F. Italiano G. Ausiello and U. Nanni. Dynamic maintenance of directed hypergraphs. *Theoretical Computer Science*, 72(2-3):97–117, 1990.
- [7] Giorgio Gallo, Giustino Longo, and Stefano Pallotino. Directed hypergraphs and applications. *Discrete Applied Mathematics*, 42(2):177–201, 1993.
- [8] Dimitrios Gunopulos, Heikki Mannila, Roni Khardon, and Hannu Toivonen. Data mining, hypergraph transversals, and machine learning (extended abstract). In *Proc. PODS 1997*, pages 209–216, 1997.
- [9] G. Gupta, A. Strehl, and J. Ghosh. Distance based clustering of association rules. In *Intelligent Engineering Systems Through Artificial Neural Networks (Proceedings of ANNIE 1999)*, ASME Press, November 1999., volume 9, pages 759–764, 1999.

- [10] Eui-Hong Han, George Karypis, Vipin Kumar, and Bamshad Mobasher. Clustering based on association rule hypergraphs. In *Proceedings SIGMOD Workshop Research Issues on Data Mining and Knowledge Discovery(DMKD '97)*, 1997.
- [11] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In *2000 ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12. ACM Press, 05 2000.
- [12] S. Jaroszewicz and D. A. Simovici. Pruning redundant association rules using maximum entropy principle. In *Advances in Knowledge Discovery and Data Mining, 6th Pacific-Asia Conference, PAKDD'02*, pages 135–147, Taipei, Taiwan, May 2002.
- [13] Brian Lent, Arun N. Swami, and Jennifer Widom. Clustering association rules. In *ICDE*, pages 220–231, 1997.
- [14] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.
- [15] S. Sarkar M. Ramaswamy and Y. Chen. Using directed hypergraphs to verify rule-based expert systems. *IEEE TKDE*, 9(2):221–237, 1997.
- [16] G. Piatetsky-Shapiro and C. Matheus. The interestingness of deviations, 1994.
- [17] V. Pudi and J.R. Haritsa. Reducing rule covers with deterministic error bounds. In *Proceedings of the 7th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 313–324. Springer, 2003.
- [18] M.D. Rice and M. Siff. Clusters, concepts, and pseudometrics. In *Electronic Notes in Theoretical Computer Science*, volume 40. Elsevier, 2002.
- [19] S.Chawla, B.Arunasalam, and J. Davis. Mining open source software(oss) data using association rules network. In *Advances in Knowledge Discovery and Data Mining, 7th Pacific-Asia Conference, PAKDD'03*, pages 461–466. Springer, 2003.
- [20] M. Zaki and M. Ogihara. Theoretical foundations of association rules. In *Proceedings of 3rd SIGMOD'98 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD'98)*, Seattle, Washington, USA, June 1998., 1998.
- [21] Mohammed J. Zaki. Generating non-redundant association rules. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 34–43. ACM Press, 2000.