# IRIS: Our prototype rule generation system

Lisa Singh*, Peter Scheuermann, Bin Chen

Northwestern University, ECE Department, 2145 Sheridan Road, Evanston, IL 60208

## ABSTRACT

Our goal is to design a knowledge discovery tool that has the ability to accurately generate rules using concepts and structured data values extracted from semi-structured documents. To date, two of our major contributions have been the design of a system architecture that facilitates the discovery of rules from HTML documents and the development of an efficient association rule algorithm that generates rule sets based on user specified constraints. This paper discusses each of these contributions within the framework of our prototype system IRIS. IRIS allows users to specify a set of constraints associated with a particular domain and then generates association rules based on these constraints. One of the unique features of IRIS is that it generates rules using the more structured component of the HTML documents, as well as the conceptual knowledge extracted from the unstructured blocks of text.

**Keywords:** textual knowledge discovery, semi-structured data mining, constrained rule generation, IRIS

## 1. INTRODUCTION

The explosion of the World Wide Web (WWW) has created a need to obtain knowledge from semi-structured data. Most Web pages contain a mix of lists, images, and large blocks of text. This data needs to be processed so as to allow knowledge discovery on the Web. Our goal is to design a KDD tool that has the ability to accurately generate rule sets using concepts and structured data values extracted from semi-structured HTML documents. Our Web mining tool, the Inductive Rule Identification System (IRIS) allows users to specify a set of constraints, either structured data values or concepts, and obtain a set of association rules that contain the constraints and concepts related to the constraint set. We believe that preconstraining the document set is necessary to efficiently generate rules for large Web document sets.

In order to accomplish this, the implementation of IRIS is based on an architecture that facilitates the discovery of rule sets. The architecture contains five major components that together maintain a condensed version of Web documents associated with one or more domains. IRIS also employs an efficient association rule algorithm that uses sparse matrices to generate large itemsets.

This paper presents the main features of IRIS, focusing on the architecture and association rule algorithm. The remainder of this paper is organized as follows. Section 2 provides background concepts and motivating examples that attempts to illustrate the need for semi-structured mining tools. Section 3 describes our architecture. Section 4 presents an overview of our implementation of IRIS. Section 5 highlights the fully constrained association rule algorithm that IRIS employs. The paper concludes in section 6.

## 2. MOTIVATION

### 2.1 Background Concepts

Magazine articles, research papers, and World Wide Web HTML pages are traditionally considered semi-structured information. Each of these examples contains some clearly identifiable features, including author, date, publisher, product name, product price, and/or WWW address. In this paper, we refer to these identifiable features as *structured attributes*. Each document also includes blocks of text that are considered unstructured components of the document, e.g. abstract, headings, lists, and paragraphs. We define a *concept* to be any meaningful word, phrase, acronym, or name that has been extracted from the unstructured components of a document.

* Email: {lsingh, peters, bchen @ece.nwu.edu}

Data mining is the search for hidden patterns in data sets. One form of pattern discovery is rule set discovery. Although many different types of rule sets exist, this paper will focus on algorithms for discovering association rules. *Association rules* provide information about relationships among different sets of values. Formally, [1] defines an association rule in transactional databases to be an expression of the form

$$X \rightarrow Y , \tag{1}$$

where X and Y are sets of items. The support of *itemset* XY is the probability of joint occurrence of X and Y, P(XY). A *large itemset* is one in which P(XY) is above minimum specified support. The confidence of Equation 1 is defined as the conditional probability of Y given X, P(Y|X).

The definition of association rules in a transaction database can be extended for use in the semi-structured domain. Specifically, each document can be viewed as a transaction and each structure value or concept as an item. Although semi-structured data can easily be modeled as a set of transactions, we use an adjusted model that is more conducive to efficient mining of semi-structured data sets. This will be discussed further in section 3.

### 2.2 Motivating Examples

The problem with text data is that limited insight about documents can be attained using only the structured document components. The following are examples of association rules IRIS is designed to generate efficiently. Because rules involving either structured attributes or unstructured concepts are more straightforward to determine, we will focus on the discovery of rules involving both structured attributes and unstructured concepts.

***Example #1 : Digital Library Documents***

Given a document collection containing business abstracts and journal articles, suppose a potential article submitter or journal editor inputs a publication name {Harvard Business Review} and a single concept {corporate profile}. The following association rule might interest the user:

Rule A: *45% of the **corporate profile** articles published in **Harvard Business Review** focus on **reengineering**, while only 5% discuss **human resource management**.*

*Harvard Business Review* $\Lambda$ *corporate profile* $\rightarrow$
*reengineering : 45%*
*human resource management : 5%*

*Corporate profile*, *reengineering*, and *human resource management* are all concepts extracted from unstructured blocks of text. In contrast, *Harvard Business Review* may be stored as a structured value in the database, e.g. as an instance of a publication. Rule A cannot be generated without the following knowledge:

- Corporate Profile relates to Reengineering
- Corporate Profile relates to Human Resource Management
- Corporate Profile is broader than Reengineering
- Corporate Profile is broader than Human Resource Management

This example illustrates the importance of maintaining concepts and relationship information among concepts. Without concepts, the final rule set excludes much information attainable from a textual document. Without knowledge about concept relationships, we are unaware of which combination of concepts generate the most interesting rules. Determining concepts and relationships that exists among concepts is too time consuming as an on-line procedure. Therefore, in order to extract Rule A efficiently, we must predetermine this information and maintain this knowledge in our knowledge discovery system. Concept data can be stored in a database or another external data structure. As will be illustrated in section 3, we maintain this information in a concept library.

***Example #2 : WWW HTML Documents***

Suppose a user is interested in determining whether or not to open a Web based business, e.g. an on-line winery. He would be interested in determining some information about wineries that already exist on-line. The following rules might help him decide the similarity between his wines and those of other on-line wineries:

Rule B: *20% of the **Chardonny** wines are **dry**.*

$$Chardonny \rightarrow dry : 20\%$$

Rule C: *80% of the **fruity Merlot** wines are **thick**.*

$$Merlot \wedge fruity \rightarrow thick : 80\%$$

In this example, *Chardonny* and *Merlot* are examples of structured data values for the attribute wine type, while *dry*, *thick* and *fruity* are examples of unstructured concepts that must be extracted from the unstructured blocks of text on a WWW page. For different domains, different sets of structured values exist.

HTML documents on the Internet tend to be more diverse than documents in a digital library or an information retrieval system. First, those in the library tend to have a clear delineation between the structured data component and the unstructured data component. In contrast, because different sites maintain different HTML documents, complete uniformity does not exist across sites. Further, the information within a document in a digital library all relates to a single set of structured values. However, it is not unusual for HTML pages to contain information about a number of different sets of structured values. Because we are interested in generating rules from varying types of semi-structured data, our system design must store information in a manner conducive for both data sets to be mined easily. Given the additional complexity of an HTML data set, the remainder of the discussion in this paper will focus on the HTML data set.

## 3. UNDERLYING ARCHITECTURE OF IRIS

In our previous work, we introduced a system architecture that attempts to provide an infrastructure robust enough to facilitate the discovery of rules from semi-structured data sets [2]. Figure 1 illustrates the high level data mining tool architecture. Some advantages of this architecture include the following:

- Maintaining a compact representation of the document set
- Storing concept relationships
- Employing intelligent agents to collect data from across the Web
- Distinguishing general and specialized concepts
- Having the ability to be built above existing databases

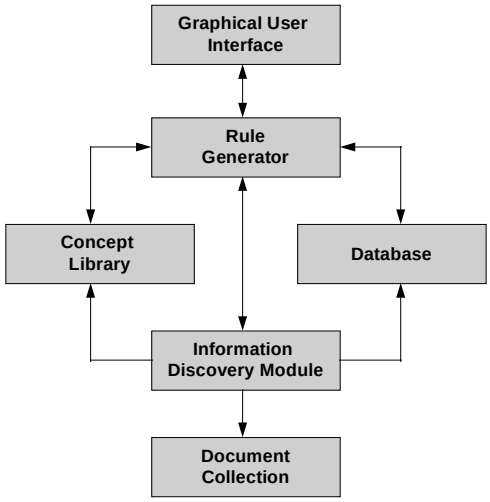

Figure 1: System Architecture

In this architecture, concepts are stored in the concept library, while structured values are stored in the database. This is a difference from traditional transaction systems that store all items with each transaction. Instead of storing each document transaction in the database, we choose to associate transaction ids with each item. In other words, for a given item (concept or structured value), we maintain a mapping to transaction (document) ids in the concept library and the database. For preconstrained textual mining, this structure facilitates the rule discovery process. Specifically, in Section 5 we will show that the problem of discovering large itemsets can be reduced to traversing sparse matrices.

The *rule generator* is the central component in the design. The rule generator parses a request submitted by the user and determines an execution strategy based on specified constraints. The strategy identifies the order and the amount of data requested from the concept library and the database. In more dynamic domains, e.g. WWW, the content of documents may change over time. For those cases, the information discovery module obtains up-to-date information that is also used by the rule generator. The rule generator processes all this information and generates association or classification rules in response to the original request. To date, IRIS only contains a constrained association rule algorithm. Over time we intend to expand

its functionality to include partially constrained, unconstrained association rule algorithms, as well as classification algorithms.

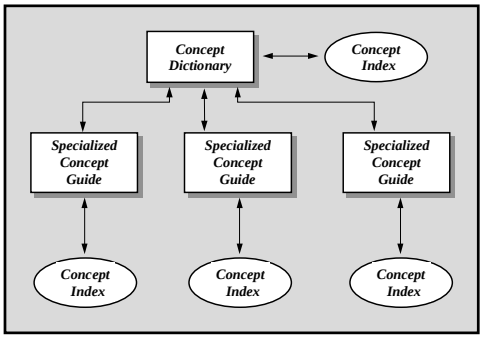


Figure 2: Concept Library Components

The *concept library* consists of three components: the *concept dictionary*, the *specialized concept guides*, and the *concept indices*. The *concept dictionary* contains common concepts extracted from a basic dictionary. In contrast, each *specialized concept guide* maintains domain specific concepts. Both the concept dictionary and the specialized concept guide maintain relationships among concepts. By distinguishing domain specific concepts from general concepts we do not have to 'rediscover' the worthiness of commonly used concepts for different domains. Further, because a large set of words use the same definition across domains, duplicating these concepts for each domain is a waste of disk space. Finally, the *concept indices* maintain lists of document ids associated with each concept in the concept library. Figure 2 shows the relationships among these three components. For more details about the concept library, refer to [2].

Our architecture also includes an *information discovery module*. This module consists of three components: the discoverer, the extractor and the refresher. The *extractor* populates the database and the concept library for every specialized domain. A domain expert gives specifications that can be used as a template for identifying structured values. An unsupervised domain specific extraction program can then use this knowledge to populate both the database and the concept library.

The discoverer and the refresher are specifically included for more dynamic domains, e.g. WWW. The *discoverer* is an intelligent agent that looks at different search engines and tries to find pages associated with different specialized domains. The *refresher* employs a dual update scheme to keep the data in the database and the concept library consistent. The first scheme is a proactive scheme that periodically checks the expiration date of different data and updates it if the expiration date is approaching. Since it is not always possible to keep everything up to date, the refresher may receive requests from the processor to update information in the database and the concept library. When it receives a request, it informs the extractor so that the document parsing process can begin.

## 4. IRIS OVERVIEW

We have developed IRIS (Inductive Rule Identification System), an initial prototype system that implements the system architecture presented in section 3. For this prototype we are using an Oracle 7 DBMS to store our structured attribute values. We use hash tables based on linear hashing to store the information in the concept library. The storage mirrors that of the extended concept hierarchy (ECH),[3] where a separate ECH exists for each specialized concept guide. The initial prototype domain is winery WWW pages.

The list of concepts and relationships in the concept dictionary was developed using WordNet. WordNet, created at Princeton University, contains lexicographer files that organize nouns, verbs, adjectives, and adverbs into groups of synonyms and describes relations between synonym groups [4]. Terms that are not on a *stoplist* are maintained in the concept dictionary. A stoplist contains a set of words that provide no insight into the content of a document. Examples include 'the', 'or', and 'a'.

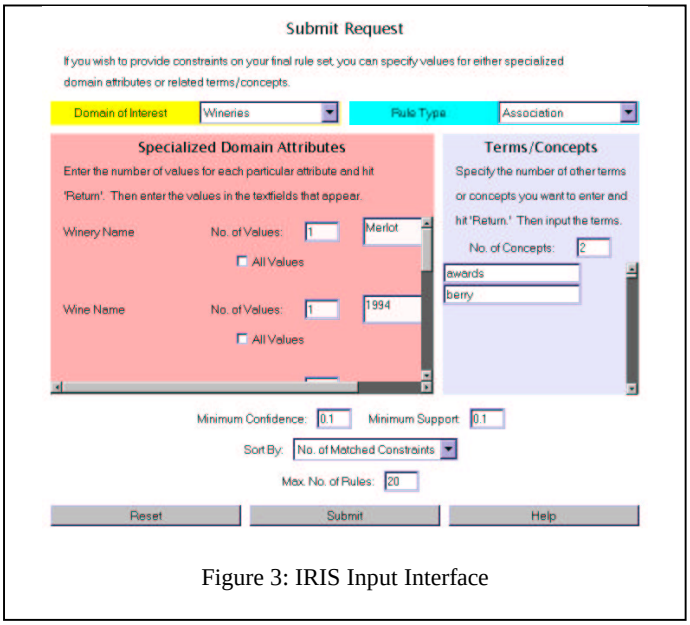


Figure 3: IRIS Input Interface

In contrast, the information in the specialized concept guide is extracted from the documents themselves. During the extraction process, we compare each meaningful concept to concepts in the concept dictionary. If the concept is in the concept dictionary, we initially assume it is a general concept. If the concept is not in the concept dictionary, it is placed in the specialized concept guide. As concepts are added, their specialized relationships to other concepts are determined using weighted frequency of co-occurrence measures[5]. Because this does not always result in semantically consistent relationships, we are investigating other approaches for generating specialized relationships, including synonym detection using natural language processing. Clearly, if ontological information exists for a specialized domain, it should be incorporated into the specialized concept guide.

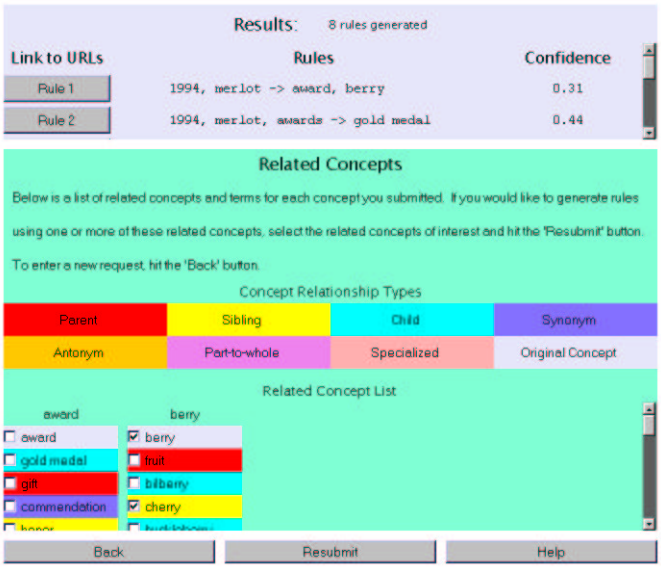

Figure 4: IRIS Output Interface

To populate the database and the specialized concept guide, we identified online wineries using WWW search engines. Currently, our discoverer module only uses the Yahoo search engine. We hope to increase the number of search engines checked by the discoverer in the future. For this initial prototype, we manually collected sites from other search engines. We then developed templates for extracting structured values (price, wine type, year, etc) from the HTML documents. During the same pass of the document, our extraction tool also identifies meaningful unstructured concepts. Using HTML tags and information retrieval parsing techniques, the weight or importance of each concept within a document was determined. We then employ a heuristic that uses this information along with frequency of occurrence information to identify relationships among domain concepts. At this stage, we have not implemented the refresher component of the architecture.

Figure 3 illustrates our user input interface and some sample input values from the winery domain. This GUI is implemented using JAVA. On this screen, the user can specify the following information:

- The domain of interest (wineries)
- The type of rule to be generated (association, classification, etc.)
- Values for structured attributes stored in the database (winery name, wine name, etc.)
- Values for related unstructured concepts from the concept library
- A minimum support
- A minimum confidence
- A maximum number of rules
- A sort by ordering for the final set of rules

Figure 4 shows the output interface with some sample output. First, all the rules and their confidences are listed. In addition, a button next to each rule links to a list of documents associated with the rule. This allows the user to easily link to pages supporting the rule of interest. In this example, rule 1 is a general rule that uses all the values specified by the user. Rule 2 is a child rule that incorporates knowledge about the child concept *gold medal*. This rule can be generated because of the concept relationship information maintained in the concept library. The next section describes the algorithm used to generate these rules.

## 5. LARGE ITEMSET DISCOVERY

A small amount of literature exists about semi-structured text mining algorithms and applications.[3,6,7,8,9] Of those, [3] is the only algorithm that uses prespecified constraints in the algorithm. The other works propose more unsupervised procedures that can be costly. Our previous work in [3] identified associations among concepts and a single structured value. We will now briefly discuss the basic idea behind our constrained association rule algorithm that finds associations among an

unlimited number of concepts and structured values.[10] Figure 5 highlights the main steps of the algorithm.

After the user inputs a set of constraints, our algorithm creates matrices and lists based on these prespecified constraints. The matrices are then used to generate large itemsets based on the original set of constraints. For data sets we have tested, we found that matrices containing constraints tend to be sparse. We can take advantage of the sparsity of these matrices and generate higher order large itemsets faster than if we intersect all subsets of constraints.

Algorithm: Generalized Constrainted Association Rules

1. Get document ids for all concept constraints
2. Get document ids for all structured value constraints
3. Remove any constraints below minimum support.
4. Create structured value matrix
5. Determine structured value large item sets
6. Create concept matrix
7. Determine concept large itemsets
8. Identify structured value – concept large itemsets
9. For each large itemset,
   create relative concept list
   Generate additional large itemsets based on r-list
10. Calculate confidences

Figure 5: Constrained Association Rule Algorithm

The algorithm begins by obtaining the document ids of all the structured values and all the concepts originally requested by the user. The structured value information is obtained from the database, while the concept information is obtained from the concept library. For all the specified constraints, we check the counts of the document list to determine large 1-itemsets and remove constraints that are not. We create 2 lists, one containing structured value large itemsets and the other containing concept large itemsets. Currently, we have only large 1-itemsets. At this stage, a structured value matrix is created using large 2-itemsets generated from the list. Once populated, this matrix will be used to determine higher order structured value large itemsets. Once the matrix is fully populated, the sum of the cells in each column is calculated. This *pair count* identifies the number of large 2-iemsets existing in the column.

We will now use the matrix to generate higher order large itemsets. The advantage of this matrix is that higher level large itemsets can be determined without first determining the intersections of every subset of structured values. Since all subsets of a large itemset must be large, we can use the information to find these larger sets. There are three criteria for determining the large itemsets of size N, where N is larger than 2:

- The pair count for one of the N columns containing the data values is at least N - 1.
- Each pair in the set must have a a corresponding 1 in the matrix.
- The set must have support above the minimum specified support.

Usually, the association matrix is very sparse. Therefore, much work is saved by not testing every combination of the input set. For semi-structured data the matrix is a simple method to generate larger sets from pairs. A matrix is also used to identify large itemsets involving concepts. However, the process is less costly because intersections of document id lists are avoided when creating the matrix. This results because we have relationship information in the concept library.

In order to create the matrix, we look at if the weight of the relationship between every pair of concepts in the concept list is above the minimum specified weight. Similar to the structured value matrix, the concept matrix maintains pair counts. Once the matrix is generated, we find the higher order large itemsets using the same approach as that for structured values. The major distinction is that in this case, the potential large itemsets do not need to be verified. They are the actual large itemsets. We avoid this because a semantic relationship is stronger than a relationship supported by the document collection. In other words, a semantic relationship still holds even though the data does not have enough support for it. Therefore, if every pair of concepts in a set has a strong relationship, the set is a large itemset. For this reason, we union the document lists of the concepts in the large itemset. This is now considered the document list of the large itemset. In this way, we assure that the semantic set is also a large itemset. All concept large itemsets are added to the concept list. At this stage in the algorithm, we have two lists, one consisting of structured value large itemsets and the other of concept large itemsets.

For every structured value set we attempt to generate large itemsets involving a concept set. We calculate the joint probability of occurrence by intersecting the document lists of the structured value large itemset and the concept large itemset. Once again, if the joint probability of occurrence is above the minimum support, a new large itemset has been found. This large itemset can now be used in rules.

To generate large itemsets including related concepts or themes not prespecified by the user, *a relative concept list*, must be determined. This list contains relatives of concepts with strong relationships. The purpose of generating these itemsets is to facilitate the discovery of new concept sets that might otherwise have been overlooked by the user.

The first step in creating the relative list is to add the relatives of one of the concepts $C_i$ in a large itemset to the list. A counter associated with each relative concept is set to 1. Then for every other concept in the set, the relatives are compared to those in the relative concepts. If it appears in the relative list, the counter is incremented. Once all the relatives for each concept in the large itemset are accounted for, the relative list is scanned to find those concepts that have a count equal to the size of the current large itemset. If one exists, a new large itemset composed of the current large itemset and the relative is created. This step is repeated for each structured value - concept large itemset.

The final step involves generating the actual rules. Similar to other works, we specify rules by finding antecedents and consequents above minimum confidence. One additional step we include involves checking the relationship type information to attempt to identify a rule with a meaningful semantic relationship.

There are a number of advantages to our algorithm. First, it incorporates knowledge from the structured and unstructured component of the document set. Second, because users specify constraints of interest, the entire document space does not need to be searched. This in turn reduces the execution time of the algorithm and returns a more condensed set of rules to the user. Finally, because we maintain concept relationship information, rules that contain related concepts are also generated.

## 6. CONCLUSIONS

This paper describes IRIS, a prototype rule generation system. IRIS allows users to specify a set of constraints associated with a particular domain and then generate association rules based on these constraints. The final rule set also includes concepts related to the original constraints. IRIS implements an architecture designed specifically for semi-structured data mining applications. Because the requirements of semi-structured data mining differ from structured data mining using a specialized architecture increases the efficiency of our mining algorithms.

Although the initial implementation of IRIS is complete, we still hope to enhance the system capability in the future. As mentioned earlier in the paper, there are still a few components of the architecture that we need to implement or improve. We also want to increase the types of rule generation tasks that IRIS can conduct. We intent to add functionality for partially constrained association rules and unconstrained association rules. Partially constrained association rules differ in that the user specifies some specific constraints, i.e. structured values or concepts and also specifies the attribute or concept space over which to generate large itemsets. Using our winery example, a user may specify a wine type, e.g. Chardonny and a year, e.g. 1990 and specify the entire concept space and the attribute space of wineries. This means that the large itemsets generated will include at least one specified constraint, Chardonny or 1990 and/or any concept in the winery domain and/or any winery. In this manner, we still avoid searching the entire database, while allowing flexibility in the final rule set. Of course, times exist when it is necessary to search for all possible combinations of large itemsets. For these cases, we hope to implement an unconstrained association rule algorithm. Of course, classification and clustering algorithms are also features we would like to incorporate into IRIS.

## REFERENCES


1. R. Agrawal and R. Srikant, "Fast algorithms for mining association rules", *Proceedings of the International Conference of Very Large Databases*, September 1994.
2. L. Singh, B. Chen, R. Haight, P. Scheuermann, and K. Aoki, "A robust system architecture for mining semi-structured data", *Proceedings of International Conference on Knowledge Discovery and Data Mining*, 1998.
3. L. Singh, P. Scheuermann, and B. Chen. "Generating association rules from semi-structured documents using an extended concept hierarchy", *Proceedings of the International Conference on Information and Knowledge Management*, November 1997.
4. G. Miller. WORDNET: An on-line lexical database. *International Journal of Lexography*, 1990.
5. G. Salton. and M. McGill, *Introduction to Modern Information Retrieval*, New York, New York: McGraw-Hill, 1983.
6. Amir, R. Feldman, R. Kashi, "A new and versatile method for association generation", *Information Systems*, Vol 22, 1997.

7. R. Feldman and I. Dagan, "Knowledge discovery in textual databases", *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, 1995.
8. R. Feldman and H. Hirsh, "Mining associations in the presence of background knowledge", *Proceedings of the International Conference on Knowledge Discovery in Databases*, 1996.
9. K. Lagus, T. Honkela, S. Kashi and T. Kohonen. "Self-organizing maps of document collections: a new approach to interactive exploration", *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, 1996.
10. L. Singh, B. Chen, R. Haight, and P. Scheuermann, "An algorithm for constrained association rule mining in semi-structured data", *to appear in the Lecture Notes Series in Artificial Intelligence*, 1999.