UNIC: Unique Node-Item Counts for Association Rule Mining in Relational Data

Christopher Besemann
Department of Computer Science
North Dakota State University
Fargo, North Dakota 58105, USA
christopher.besemann@ndsu.nodak.edu

Anne Denton
Department of Computer Science
North Dakota State University
Fargo, North Dakota 58105, USA
anne.denton@ndsu.nodak.edu

ABSTRACT

An association rule mining algorithm is introduced that is specifically designed for relational data. Applying standard ARM techniques to the relational setting leads to an overwhelmingly large number of rules that can be explained by simpler considerations, such as neighbor correlations or ARM on isolated nodes. Worse, even rules that cannot by themselves be explained in such a way, in fact, may be a consequence of simpler associations or correlations. Our algorithm, UNIC, uses only items that are unique to one of the nodes under consideration. This strategy is highly effective at eliminating undesired contributions to support and confidence, while achieving most pruning at the itemset level.

1. INTRODUCTION

The practical relevance of data with a relational structure is easily demonstrated through the success of relational databases, and data mining techniques for relational data are increasingly receiving the attention they deserve [12]. Two main challenges have slowed down progress. In the relational setting distant objects can influence each other leading to a dramatic increase in computational complexity between non-relational algorithms and their relational counterparts. More importantly, the naive translation of a non-relational algorithm can often lead to skewed [15, 8] or irrelevant results in the relational setting. In this paper we focus on networks of objects for which interactions are predefined, corresponding to a many-to-many relationship of an entity with itself. Examples include web-pages that are connected through hyperlinks, social networks, scientific papers with their mutual citations, and protein-protein interaction networks. Some interactions can be represented through directed graphs and others by undirected graphs. Specialized graph data mining techniques exist that focus on graph structure, and typically assign no more than one property to a node. In contrast, objects in relational data are characterized by an entire record of data that can itself be used for data mining purposes.

Association Rule Mining [2], ARM, is a powerful technique for the discovery of patterns among sets of items that characterize an object. In traditional market basket analysis objects are shopping carts and items are the purchases of a customer. ARM has been applied to many other settings, including proteins and their annotations [20]. Proteins are annotated with functions, localizations and other properties. The standard ARM setting would look for rules that pertain to an individual protein, such as if the protein is involved in the function *TRANSCRIPTION(TRANS)* then we can assume with 66% confidence that it will be localized in the *NUCLEUS(nucl)*. To understand the relevance of interactions it is even more interesting to know which combinations of annotations in one protein are an indication of which properties in an interacting protein [20]. In our evaluation we use a similar data set that was generated from undirected protein-protein interactions in yeast [13] together with node data consisting of annotations that can be seen as Boolean variables. Note that the analysis of data on multiple nodes requires joining of tables. Depending on the hop neighborhood or path-length under investigation tables have to be joined differently. Throughout the paper we refer to hops as the number of interaction edges from one starting node to all neighbors (each edge forms a hop). Path-length refers to the same concept under the setting of linear paths rather then some radius of neighbors. For relational ARM this is rarely a problem since very few hop structures are likely to be relevantas the length increases. While Graph ARM [14, 17, 22, 7] is concerned with scaling as a function of the subgraph size, the major problem in relational ARM is scaling with respect to itemset size. Also due to the "small world" property of scale-free networks any protein can be reached from almost any other by means of no more than three interactions [4, 21]. Association rules that involve longer distances are therefore unlikely to produce meaningful results.

Some correlations are very common in relational data. It has been shown that relational neighbors typically have the same properties [18] and, similarly, that interacting proteins are likely to have the same functional annotations [11]. We will call rules that associate identical items in neighbors "boring". It can also be expected that association rules

within objects are often stronger than rules involving neighbors. Such rules should be analyzed using ARM on isolated nodes rather than on joined tables. Each join on tables weights nodes according to the number of their interactions. While this is natural for rules involving neighbors, it does not produce results that appropriately describe associations within nodes. In relational ARM we have to be specific about hop settings and the shape and we will consider rules that do not fully employ the hop neighborhood in question as "out-of-scope". Eliminating rules that satisfy our criteria for being "boring" or "out-of-scope" does not yet resolve all related problems. Let us look at the association rule for proteins $\{TRANS\} \rightarrow \{nucl\}$. We have seen that proteins with property *TRANS* may be particularly likely to interact with other proteins with property *TRANS*. It should, therefore, not surprise to find rules such as $\{1.TRANS\} \rightarrow \{0.TRANS, 1.nucl\}$(R1) where 0 and 1 stand for two interacting proteins. It is quite likely that the confidence of the above rule is almost completely determined by the confidence of the boring rule $1.TRANS \rightarrow 0.TRANS$ and the out-of-scope rule $1.TRANS \rightarrow 1.nucl$. This relationship between confidence values will be discussed in more detail in Section 4. The rule (R1) is likely not to add information compared with boring and out-of-scope rules. Issues of redundant and misleading rules have extensively been discussed in the standard ARM setting [9, 24, 5]. In the relational setting these problems are aggravated by the fact that commonly the strongest and simplest rules are also the least interesting ones.

The problem is not limited to itemsets of the shape $\{0.A, 1.A, 1.B\}$ or their generalizations. The support of the itemset $\{0.A, 1.B\}$ is guaranteed to be at least as high as the support of $\{0.A, 1.A, 1.B\}$. Application of standard ARM techniques commonly produce a significant number of rules of the shape $0.A \rightarrow 1.B$ that are entirely due to high-confidence rules $0.A \rightarrow 1.A$ *and* $1.A \rightarrow 1.B$. One could consider defining a complex set of conditions to filter the output of a standard ARM algorithm. This alternative is not practical in most settings due to the high volume of irrelevant itemsets. Section 5 will demonstrate that standard ARM typically produces many orders of magnitude more irrelevant rules than relevant ones. Enabling pruning at the itemset level is therefore — as almost always in ARM — the key to achieving performance.

Our algorithm takes the following approach to itemset pruning: Only those itemsets are considered for which each item is unique to one node in a hop setting, which means it does not occur in any of the neighbors within the considered path. This goal is achieved by removing any items that occur on multiple nodes during construction of the joined table. In the example of a record $\{0.A, 1.A, 1.B\}$, item A would be removed from all nodes. In Section 4 we will show that this procedure will not only remove rules that involve itemsets of the shape $\{0.A, 1.A, 1.B\}$ and their generalizations. It will also adjust the support of itemsets $\{0.A, 1.B\}$ to eliminate the impact of $\{0.A, 1.A, 1.B\}$ and $\{0.A, 0.B, 1.B\}$. A simple example can illustrate the idea: Consider a study to find association rules involving people who communicate through e-mail, and consider the observation that people who publish KDD papers typically communicate with people who play the violin. This observation is clearly only relevant if it is based on people who do not themselves play the violin, or communicate with people who play the violin as well as publishing KDD papers. It is therefore natural to exclude the properties of violin-playing and KDD-paper-publishing that occur on both sides of an e-mail communication considering that learning about association rules involving e-mail communication is the goal.

Organization of this paper proceeds by describing related works to our approach as well as our contributions to the area in Section 2. We will then describe our algorithm which is mathematically shown to avoid some problem situations in Section 3. An independent approach to the problem is discussed in Section 4. After algorithm development, implementation is described in Section 5 where we used the Apriori algorithm as implemented in [6] as a core function and standard ARM benchmark. We performed mining on four different hop settings using a standard ARM and then using our Unique Node-Item Count algorithm. Analyzing the results we found that trivial modifications made great improvements over standard ARM and UNIC was able to make significant processing and readability improvements over the modified ARM. We close with conclusions and future expansions in the final section.

2. RELATED WORK

The basic framework on which our process is built is standard ARM as it was first established [2]. In this setting, the database can be represented as one relational table with tuples containing binary items from a basic set of items. Once we include the interaction relation among entities, the framework becomes more complex. For our purposes, we will look at the effects of creating a joined table by using the interaction table a given number of times. Three areas which deal with similar forms of such data are graph-based data mining, sequential pattern mining and inductive logic programming and multi-relational mining.

2.1 Graph-based ARM

Dealing with relational data that has an interaction relationship is close to many graph problems. A graph in the most basic setting is defined by a set of nodes and a set of node pairs which define edges between two nodes. This is analogous to our data where we have the interaction table which is just a standard edge table in graph terms.

Graph-based ARM uses relationships to build a graph representation of the database, usually in the form of adjacency matrices, and then finds frequent substructures of the graph in Apriori fashion. Much progress has been made in this area as shown in various works [14, 17, 22, 7]. Graph-based ARM concerns itself with identifying patterns of nodes and edges, which places the focus on the structure of interactions. Most settings allow edge labels and node labels in the definition. Both UNIC and graph-based approaches depend on the edge relation to extract information about structure.

Our approach places focus on the node content more than relative graph structure and thus we require a record item for each node. There are methods where one might include such itemset information in a more complex graph structure and directly apply graph-based ARM without using a record. These methods are inefficient to mine and they still require additional interpretation steps to transform from a

structural format to attribute information format. Graph-based ARM does not perform prejoin operations, but incrementally grows each pattern. In our case this pattern growing could be a complex and long process, instead we tend to focus on a particular pattern size at a time.

2.2 Sequential Pattern Mining

Sequential pattern or episode mining can be done in similar form to graph-based mining. Sequential pattern mining [3] shares the same basic framework as traditional ARM, but is extended to deal with an ordering relationship among the entities. Sequences are linear graph structures or paths. This type of mining has received as much attention as association rule mining in general [23, 19, 25].

It can be seen that sequential pattern mining has the capability to mine a subset of the transactions we may produce for our algorithm. The subset is that of all linear patterns as part of Figure 1. If the operation of sequential pattern mining is constrained enough by only allowing strict matching of patterns it becomes the same as standard ARM applied to our data tables. Noted is the additional fact that we do not apply ARM to data tables we have directly.

2.3 Multi-relational and ILP

Approaches including inductive logic programming (ILP) and multi-relational mining [10, 16, 8] also address the problem of extracting information from the relation or interaction schema and tables. UNIC addresses a special case in relational mining but can benefit from some of the insights gained from established relational work.

Multi-relational algorithms call to light some problems of single table mining and solutions. Among these is the case where a rule does not involve every table in the relation. The approach then has been to create the environment of only the involved tables as a base for the rule. Multi-relational mining techniques essentially generate units of interest - through operations which traverse the schema or table structure. Our situation calls to focus not only the collapsed, single table problems but considerations that must be taken when the relational joins are recursively defined. We have followed some of relevant table inclusion principles already defined and additional pruning to produce a cleaner picture for the analyst. Not only do we wish to produce "correct" results in the relational environment, but we address issues of interest and relevance of these results which is distinguished from the goal of other works. On-the-fly joins are one method used in this context. While we have considered this approach, we have found better results by performing a single pre-join.

3. UNIC ARM ALGORITHM

Unique Node-Item Count is a coupled algorithm which mines patterns based on differences among interacting nodes reflected in attributes that are in a given hop neighborhood. These differences are easily reflected in item attributes that are unique to one node in a neighborhood of interest. The unique property gives insight into the consequences of interactions and also the causes of those interactions. Following in Section 3.1 we give a context and groundwork for our algorithm. We proceed in Section 3 with algorithmic details.

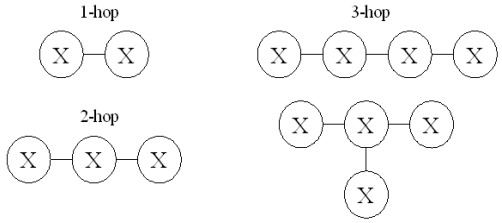


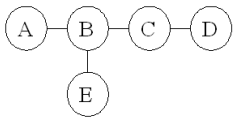
Figure 1: Neighborhoods in Hop Settings

3.1 Context

For our purpose we consider the case where mining takes place over relations which have a defined interaction of a single entity with itself. This can be viewed at its simplest as an equi-join over an entity relation and its interaction relation causing linear paths of a given hop distance to be formed. In order to place a bound on the patterns generated and the number of hops needed to exhaust the space we remove and terminate cycles at the join level. Throughout this paper we will often refer to transactions by the length of their graph path and the number of joins performed denoted by the number of hops it takes travel the path. In Figure 1 we show different hop settings. A particular hop setting can be viewed by looking at each node position in the path as made up of the basic attribute set with the position annotated to it. For example node position 0 may have $\{TRANS\}$ as an item, so this item would then be $\{0.TRANS\}$. In this way there is in effect a different attribute set for each position. The network, as in Figure 2, is central to our model where every node can be considered a starting position in a path. Whether we are dealing with directed or undirected data effects how much redundant work can be avoided in the mining process. As we will explain later, removing trivial redundancy that forms out of undirected behavior is a necessary step to provide effective results.

When performing joins across the interaction relation we notice that it is not possible to summarize a lower level with higher one or use a lower level table to build the higher level one if we have done UNIC processing on it. We rely on knowing the initial state of a transaction to perform the UNIC processing. If a lower level table has not been modified we can use it to build the next level. Each hop setting is a different environment with some paths branching and others terminating. The setting an itemset was joined in is important under the UNIC setting. Whenever speaking of a rule, its hop context should also be identified.

Without actually traveling the graph structure completely we cannot predict where high branching and termination occur. Branching occurs when a node has more then one choice for a neighbor as in Figure 2 where the number of times a node shows up in a position changes. This causes the multiplicity of previous nodes in the path to increase since transactions are added for each choice by the join method. Termination occurs when a path no longer has options for neighbors to extend it. In our case of inner-join this causes the nodes in that path to be eliminated. The effect these activities have on attribute counts for rules is that the counts fluctuate according to the number of hops as seen in the



1-hops	2-hops	Linear 3-hops	Graph 3-hops
A B	A B C	A B C D	A B C E
B C	A B E	E B C D	
C D	B C D		
E B	E B C		

Figure 2: Effects of Branching and Termination

table representation in Figure 2.

Complexity of our search space grows since we have to consider basic attribute sets for every path position and every position must be considered for each join environment it is present in. One might be concerned that the number of joins to be performed is not defined. A high number of joins definitely is possible with increasing complexity. The work we present is general to any number of hops; however, in consideration both for ease of translation and practical need, most applications would not need to go much beyond a two or three hops. This is supported by two situations. First, as the path length increases so does the complexity in translating rule results, therefore the effort versus information tradeoff for larger paths degrades rapidly. Second, many natural datasets have a small-world graph characteristic [21, 4]. This means that the diameter of the node graph is small. Heuristically we can state that once the diameter or radius is reached patterns will begin to repeat themselves. Practically, using our algorithm causes all items to be pruned near this limit given a fixed support threshold.

3.2 Scope of Rules

One detriment to standard ARM is that joining tables leads to redundancy that increases computational complexity and furthermore leads to a weighting of attributes based on their number of neighbors that is inappropriate for the discussion of properties within nodes or in lower hop situations. This follows from a generality that different attributes within any one node of the interaction graph are typically more strongly correlated than different attributes between neighboring nodes. Discovering association rules that hold within any one node should be done based on the nodes alone and without considering their relations.

In multi-relational mining applications, correct rule setting is brought up. Some works have cited that a flaw in flat table mining of relations is that rules containing items of a subset of tables are graded on the full join [8]. Rather, they recommend that such rules are evaluated on the relevant tables of scope. Similar to this concept we introduce rules which are called *out-of-scope*. Such rules are made of items which are expressed in a lower number of hops than they are being reported on. In other words if a rule does not contain items from every node position of the current hop setting and no more, then it is out-of-scope. This supports our findings and judgments made in other research [20].

Table 1: Frequent Items

Items	Support
META	25.37%
lethal	20.89%
nucl	20.34%
TRANS	17.32%
CCaDP	14.67%

3.3 Interesting Rules

A second problem situation is that values of a particular attribute are typically strongly correlated between neighbors. Studies have shown that a property can often be predicted based on the value of that attribute in neighbors [18]. Results from our model data set supported this by showing a correlation of an item in a node and its neighbor.

Standard ARM rules are selected based on criteria of interest. Classically these criteria are that of high confidence and support. In our case we know the trivial fact of neighbor correlation. General application can call any trivial situation non-interesting and if trivial rules can be removed without adverse effect to non-trivial rules then it should be done. We call such trivial rules not interesting but *boring*. Boring rules contain the same basic item in more than one path node (i.e. it is on relative neighbors). Since it is already known that the relation promotes such activity we wish to hide these effects since they multiply over joins.

3.4 Rule Information

A result of the above problems and the combinatory nature of interaction relations is the low information content or readability of the rule results. In a preliminary study of our data we found strong itemsets under a non-relational setting Table 1. When we applied a basic relational setting to the same dataset, these itemsets dominated the results in combinations corresponding to the interaction setting we worked with (see Table 4 in Section 5). When there are trivial facts included with rules to evaluate, the information content is low because of irrelevance or misleading results. If the trivial facts are not included, the information content increases and we can make more confident choices.

An interesting aspect of UNIC is that it allows not only viewing the linear paths that can be generated from joins but also can take into account graph substructure representations. When the distance of our neighborhood increases, we are faced with some choices of what graph structures we wish to include. For example, consider the diagram in Figure 1. When using all substructures having a maximum of 3-hop neighbors, we can add an additional structure to the linear one as shown. These additional shapes also contain useful information on the activity of an interaction network.

3.5 Unique Alternatives

When considering how to approach the problem of trivial rule influence and comparison of network neighborhood rules we can identify a number of alternative methods. In this section we discuss some of these alternatives as Unique Node-Item, Unique Neighbor-Item and Unique Neighborhood-Item. For this paper we work with Unique Node-Items for reasons that will be discussed shortly. The differ-

ence between each of these methods is what is done to each transaction to alleviate the problems we have mentioned and present good information in the content of rules. It is noted that there are other possible treatments, but these are the most useful.

Unique Node-Items are identified as those items of a transaction that only occur in one node. In treatment with our algorithm those items that are not unique node-items are removed from the transaction. Unique Neighbor-Items are items that are unique for each particular pairing of neighbors. If an item does not match this criteria it is removed from the only the neighbors involved simultaneously. The last alternative we discuss here is that of Unique Neighborhood-Items. These items are Unique Neighbor-Items for the entire neighborhood. That is if an item is not a unique neighbor-item in *every* pairing in a transaction then it is not a unique neighborhood item. Such non-unique items are removed from the transaction.

When comparing the methods to determine which to apply, we consider where they differ. At the 0-hop and 1-hop levels, the effects are the same since there are only direct neighbors to deal with. Once we move to the 2-hop level we can see the difference between Unique Node-Items and the other Unique methods. In effect the only time the result is different in the two groups is when there is an item that appears in the 0 position and the 2 position only. For example, in the 2-hop transaction {*0.A, 1.B, 2.A*}, A is removed in the Unique Node-Items setting and kept in the others. Actual figures from our data can illustrate a fact that supports our decision to use Unique Node-Items. We can notice that the presence of the same item on indirect neighbors is highly correlated as direct neighbors for some strong itemsets. Since we are trying to eliminate such trivial noise we choose the Unique Node-Item approach.

If one wanted to keep the identification of these "gapped" patterns then the choice is to either use Unique Neighbor-Items or Unique Neighborhood-Items. At the 3-hop level we can distinguish between the two methods. In Unique Neighbor-Items we allow a position of the item to be present even if it is not unique in some other pair. Take the case of {*0.A, 1.A, 1.B, 2.C, 3.A*}. Using Unique Neighbor-Items would allow the *2.A* item to remain in the transaction, while using Unique Neighborhood-Items would cause A to be entirely removed because of the 0,1 neighbor. Under analysis the difference created here will end up effecting the weight of the items in some possible rules. Understanding this weight difference would identify which method to use in this case. Any of the Unique methods show similar benefits in testing.

3.6 Unique Node-Item Count

The relational interaction association rule mining setting can formally be stated as follows:
Given a database, D, and a hop-length, h. Let
$I = \{i_1, i_2, \cdots, i_m\}$ be a basic set of items, and let $T = \{t_1, t_2, \cdots, t_n\}$ be the set of transaction identifiers. Also let IS be an itemset such that $IS \subseteq I$. IS_y is an itemset at node position y and is also noted by $y.t_x$ where it is the y position in transaction t_x. If the transaction is obvious a single item, i_1, can be given with its position of y as $y.i_1$. D is the set T.

Definition 1. Hop-Itemset
Hop-Itemset, $i = \{0.IS, 1.IS, \cdots, h.IS\}$ where $y.$ is the node position of the items.

Definition 2. Boring Item:
An item, i, is boring if for a hop-itemset there are $x.i$ and $y.j$, any two items of the hop-itemset where $x \neq y$ and $i = j$. Item j, being the same as i is also boring.

Definition 3. Out-of-scope Itemset:
A hop-itemset, g, is out-of-scope if there is $0 \leq x \leq h$ so that $x.g = \emptyset$.

UNIC-ARM Algorithm:
Hop-length= h
Prejoined Database= D
If Undirected ud =true

UNIC(h,*minconf*,*minsup*,D,ud)
1. D'=UNIC_PreProcess(h,D,ud)
2. FreqSet=Apriori:FreqItemset_Gen(D',*minsup*)
3. UNIC_Rule(*FreqSet*, h, *minconf*)

UNIC_Elimination(h,D,ud) Return→ D'
4. **Foreach** transaction, t in D
5. **If**(ud is true)
6. RemoveSymmetry:
7. **If** 0-positionID of t < h-positionID
8. Break to next t
9. **Foreach** Itemset1 in t
10. Itemset'=Itemset1
11. **Foreach** Itemset2 in t ≠ Itemset1
12. Itemset'=Itemset'-(Itemset1 ⋂ Itemset2)
13. **If** Itemset'=∅
14. Break to next t
15. t' = t'+Itemset'
16. D'= D'+t'

UNIC_Rule(*FreqSet*, h, *minconf*)
17. **Foreach** Hop-itemset in *FreqSet*
18. **If**((#positions in Hop-itemset) ≥ (h+1))
19. **Foreach** Position from 0 to h in Hop-itemset
20. **If** Position = ∅
21. Break to next Hop-itemset
22. ARM_RuleGen(Hop-itemset,*minconf*)

UNIC eliminates the effects of boring and out-of-scope rules finding all such confident and frequent rules from the relational interaction setting and is explained as follows. The prejoined database is pre-processed and then given to a standard Apriori frequent itemset generation algorithm. Frequent itemsets are given to the rule generation algorithm.

Preprocessing of the database (4.-16.) eliminates much unneeded data from the processing portion of ARM. Our database has already been joined where facts of how we want to view the data are accounted for. In our case the join was an inner-join, cycles were removed and we only join for linear paths. The first preprocessing step (5.) checks if we are working with an undirected interaction relation or not.

Transactions extracted from undirected interactions will always come in symmetric pairs where the path can traveled from any direction. Trivial redundancy due to undirected relations and their symmetry is removed by comparing the start and ending node ids of a transaction (7.). If the start id is less than the end id the transaction will not be used in rule generation since its symmetric compliment will already be represented. In this way instead of having two redundant rules say $\{0.TRANS\} \rightarrow \{1.nucl\}$ and $\{0.nucl\} \rightarrow \{1.TRANS\}$ we would have one non-redundant rule.

Second step of elimination processing (9.-16.) are transactions which contain the expected and uninteresting pairs of the same item at more than one position as in Definition 2. They are processed by removing the offending items from the itemset (11.-12.). This produces transactions that have node items that are unique, which means they appear in only one node position, Unique Node-Items. Out-of-scope rules Definition 3 do not have items covering every position in the path of the join. All transactions are in-scope before preprocessing so while removing boring items if a position becomes empty we break and do not use the transaction to preserve the scope (14.). This removal occurs since the transactions can *never* contribute to in-scope rules.

Once the database has been preprocessed, standard Apriori can take place (2.). The results are the typical set of frequent itemsets with a few modifications to how they are translated. Rules are then produced as in standard ARM by processing the frequent hop-itemsets. Out-of-scope itemsets Definition 3 are filtered from rule generation at this point to avoid producing rules which would also be out-of-scope. First, we completely ignore hop-itemsets which are smaller than the hop-length being considered (18). Then, we evaluate if the itemset indeed has at least one item for each path position (19.-20.). If it does not, the itemset can be skipped. Itemsets in scope have rules generated in the standard form (22.).

The algorithm concludes with a set of rules that characterize the influence of the relational interactions while excluding results that can be explained without the relation or by trivial facts. Our rules have the desirable attribute that they place a focus on situations that are due to interactions and do not trivially arise from strong trivial facts. For any standard rule that may have been $\{0.A\} \rightarrow \{1.B\}$ in the hop-length one setting would have meant that A in one node leads to B in a neighbor no matter that the neighbor has A also and A and B are strong without an interaction. UNIC changes the meaning to say that A leads to B in a neighbor *only* because of the interaction.

4. THEORETICAL FOUNDATIONS

We will now show that UNIC achieves the goal of eliminating contributions to support and confidence that are due to combinations of boring and out-of-scope rules, which can be compared to treatment in the standard ARM setting. Note that boring and out-of-scope rules themselves are trivially eliminated by our algorithm. This approach is different from other work on redundancy in association rules [5, 24] in which some rules are discarded in favor of others. In the presence of rules that are explicitly irrelevant we can, in principle, go a step further in standard ARM and correct support and confidence of all other rules to eliminate the contributions of the irrelevant ones. Taking such a probabilistic approach is not efficient computationally but can serve us as a model against which we can compare the performance of the UNIC algorithm. The concept used to do this is that of transitivity between rules. In the strict sense, transitivity can be applied to two 100% confident rules to form a third rule. In the model we present we do not use strict transitivity in since our rules will not be 100% confident. Our discussion will focus on rules of the type $\{0.A\} \rightarrow \{1.B\}$ in the 1-hop setting, which are the simplest rules to pass all trivial relevance tests. Generalizing the concept to more complex rules is straight forward.

The confidence of a rule can be written as conditional probability $\text{Conf}(\{0.A\} \rightarrow \{1.B\}) = \text{P}(1.B|0.A)$. Boring rules lead to high conditional probabilities $\text{P}(1.A|0.A)$ and $\text{P}(1.B|0.B)$ as the majority of the relevant contributions to a rule. The relevant out-of-scope rule $\{A\} \rightarrow \{B\}$ is defined within a node and does thereby not depend on the node identifier ($\text{P}(0.B|0.A) = \text{P}(1.B|1.A)$). Here we assume an undirected setting for the interactions; in the directed case this fact is no longer true because the table no longer has symmetric transaction pairs. By realizing that the trivial facts of boring and out-of-scope rules contribute to other results through transitivity we can try to correct standard ARM for such situations. In that case the combined probability can be calculated as the product of individual probabilities. Two cases have to be considered, one with property $0.B$ and one with property $1.A$. These cases can be viewed as the component of a rule that occurs due to boring and out-of-scope conditions. The overlap between both cases has to be subtracted leading to the following model for the irrelevant part of the conditional probability of a standard ARM rule

$$\begin{aligned} \text{P}_{\text{irrel}}(1.B|0.A) &= \text{P}(1.B|1.A)\text{P}(1.A|0.A) \\ &+ \text{P}(1.B|0.B)\text{P}(0.B|0.A) \\ &- \text{P}(1.B|1.A, 0.B)\text{P}(1.A, 0.B|0.A) \end{aligned} \quad (1)$$

The joint probability, which corresponds to rule support is

$$\begin{aligned} \text{P}_{\text{irrel}}(0.A, 1.B) &= \frac{\text{P}(0.A, 1.A)\text{P}(1.A, 1.B)}{\text{P}(1.A)} \\ &+ \frac{\text{P}(0.A, 0.B)\text{P}(0.B, 1.B)}{\text{P}(0.B)} \\ &- \frac{\text{P}(1.A, 0.B, 1.B)\text{P}(0.A, 1.A, 0.B)}{\text{P}(1.A, 0.B)} \end{aligned} \quad (2)$$

We will now compare this model with the results of the UNIC algorithm. Support of the rule $\{0.A\} \rightarrow \{1.B\}$ can be written as the following probabilities

$$\begin{aligned} &\text{Support}_{\text{UNIC}}(0.A \rightarrow 1.B) \\ &= P(0.A, \neg 1.A, \neg 0.B, 1.B) \\ &= P(0.A, 1.B) - P(0.A, 1.A, 1.B) \\ &- P(0.A, 0.B, 1.B) + P(0.A, 1.A, 0.B, 1.B) \end{aligned} \quad (3)$$

One central approximation has to be made to identify parts of this expression with equation (2).

$$\begin{aligned} \text{P}(0.A, 1.B|1.A) &= \text{P}(0.A|1.A)\text{P}(1.B|1.A) \\ \text{P}(0.A, 1.A, 1.B) &= \frac{\text{P}(0.A, 1.A)\text{P}(1.A, 1.B)}{1.A} \end{aligned} \quad (4)$$

Table 2: UNIC vs. ARM Model

Association Rule	Support %		Confidence %	
	UNIC	ARM	UNIC	ARM
{0.TRANS}→{1.nucl}	0.40	1.20	0.08	0.07
{0.META}→{1.TRANS}	0.78	0.86	0.19	0.12
{0.META}→{1.nucl}	0.30	0.16	0.07	0.02
{0.TRANS}→{1.CF}	0.46	0.27	0.08	0.02
{0.nucl}→{1.CF}	0.58	0.17	0.08	0.01

By following equation 4 we replance pieces in equation 3 like P(0.A, 1.A, 1.B). This allows equation (3) to be written equivalently to equation (2). It is assumed that if the propery $1.A$ ($0.B$ and $1.A, 0.B$ respectively) exists, - and only in that case - the properties $0.A$ and $1.B$ are independent. In the absence of the property $1.A$ one would not necessarily expect these conditional probabilities to be independent.

We have now shown that as far as support is concerned the standard ARM model matches the UNIC algorithm to the extent that could be expected. When comparing confidence we do, however, have to make an additional assumption.

$$\begin{aligned} &\mathrm{Conf}_{\mathrm{UNIC}}(0.A \rightarrow 1.B) \\ &= \frac{P(0.A, \neg 1.A, \neg 0.B, 1.B)}{\mathrm{P}(0.A, \neg 1.A)} \\ &= \frac{\mathrm{Support}_{\mathrm{UNIC}}}{\mathrm{P}(0.A) - \mathrm{P}(0.A, 1.A)} \end{aligned} \quad (5)$$

If the denominator was P($0.A$) we could rewrite confidence based on equation (4) to match the standard ARM model. Calculating P($0.A$) would, however have some serious drawbacks from the implementation point of view. A separate table from which no elements are deleted would have to be constructed, and many support calculations duplicated. No standard ARM algorithm could be used.

If we were to change the divisor to match the ARM model it would lower the confidence unfairly in the UNIC setting since we have apriori eliminated the possibility of the itemset {*0.A, 1.A*}. To completely remove the influence of boring and out-of-scope rules we must also consider the effect on the condition component of confidence. In the standard setting we would use {*0.A*} since all occurrences of that item have the potential to lead to the completion of the rule. Once we decide that the boring property is undesirable then we can determine that {*0.A, 1.A*} will not lead to a satisfactory rule. Comparing the UNIC confidence and the standard confidence of irrelevant we can see that this fact influences the difference in confidence under the two settings.

By looking at some sample rule calculations from our data in Table 2, we can see that as stated the standard ARM model comes close to UNIC by small factors from 2 to 3. If we look at confidence it is evident that this is where standard ARM does not account for the same boring and out-of-scope influences since results now range in difference from close the same confidence to off by a factor of 10. The computational benefits of UNIC outweigh this uncertainty. In using UNIC we will avoid the difference between large numbers which provides better results.

5. IMPLEMENTATION & RESULTS

Table 3: YEASTP-P Statistics

Stat.	Node/Item	Int./Item	Int./Node
Avg.	177	288	1
Std.	95	135	2

For implementation of our algorithm, we used the Apriori algorithm from Christian Borgelt [6] as the core. The preprocessing algorithm was performed on pre-joined data and the resulting output was sent to Apriori. A filter was applied to the Apriori frequent itemsets to keep the rules produced in-scope. We used one basic dataset, YEASTP-P, over four experiment settings.

5.1 Modularized Design

Before we describe the details and results of using UNIC it is beneficial to consider the issues that were addressed in the algorithm design. One aspect of the UNIC design that is important to note is the fact that each major component is independent of the others. For example, the UNIC_Elimination steps are the most important part of the process but they take place in isolation to the Apriori itemset and rule generation portion. This choice is not coincidental. Such work could very well take place during the Apriori by adding processing "on-the-fly". An approach like this has the potential to do worse for processing than the current approach. Second, this would make the implementation of the algorithm dependent on the Apriori method.

The consideration to modularize the algorithm makes it possible for us to apply any Apriori method as better implementations and algorithms are developed. For example, we could choose to use an ARM core that uses closed itemsets for rule generation if we wanted to reduce redundant results. This also makes any choice of changes easier to make, such as the case of switching to a Unique Neighbor-Item setting.

5.2 Datasets

YEASTP-P consists of two data table which were gathered from the Comprehensive Yeast Genome Database at MIPS [13, 1]. One table is yeast physical protein-protein interactions or the interaction table. These interactions are undirected and the information used is the gene ORF ids for every interacting pair. The second table consists of gene annotation data. Annotations are hierarchically structured, with hierarchies for function, localization, protein class, complexes, pathways, and phenotypes. In any category attributes are multi-valued and we pick the highest level in each hierarchy as items. Some items were added from other knowledge and interests. This annotation data table in a binary representation is called the entity table.

We used four datasets derived from YEASTP-P. The first, called 0-hop, is the entity table itself. Then we generated joined tables as previously described of 1-hop, 2-hop and 3-hop linear neighborhoods. Notice that since the table 0-hop is the entity table itself it serves as a control to ensure algorithm correctness and also serves for baseline measures.

Table 3 describes how entities and items interact through the yeast protein-protein interaction relation. The basic entity table has 6,480 gene ORF entries in it, 2,282 which have no

item annotations associated with them. Based on this information we can see that the dataset is sparse. Interactions seem to be high on average for items and are low for the node to node case. Most items and nodes have few interactions while some are "hubs" with many interactions. Of the 4,198 basic nodes used in the entity table, a little less than 50%(1,868) had any interactions which changes the values from the table for average number of interactions to about 2.5 and the standard deviation to 2.8.

Each table was run with the standard ARM implementation (ARM), a modified ARM that removed out-of-scope rules in post-processing (ARM -oos), the UNIC ARM implementation (UNIC) and a modified UNIC that does not remove out-of-scope rules (UNIC +oos). In each case the support and confidence were fixed at 3% and 20% respectively. For ease of interpretation and comparison, the tables were modified for both standard ARM algorithms so that the trivial symmetry of the undirected interaction was removed. Also for uniformity, support and confidence calculations are based on the full number of initial tuples rather then the number resulting from elimination steps. Therefore numbers of total transaction for hops greater than zero are half the true value and any support or confidence measures will differ by a factor of two, in the case of UNIC additional factors can be seen due to the other tuple elimination steps. While we could present UNIC results under the new total tuple count, we choose to present rules based on the same count an unmodified ARM algorithm would produce in order to draw comparisons.

5.3 Performance and Complexity

Complexity is an important issue in datamining algorithms. UNIC produces good results from a readability and user application view. We must also produce these results in practical time. UNIC has three contained components to consider in complexity: UNIC_Elimination, Apriori and UNIC_Rule. Since we have not worked with optimizing the standard Apriori process, the complexity of that component depends on the core used and is the same as that proven by the theory behind that core.

Post-processing takes place during the rule generation phase of ARM which has a relatively low complexity compared to the frequent itemset generation in Apriori. Simple operations can accomplish the post-processing and complete the task in linear time to the number of frequent itemsets generated. Preprocessing after the pre-joins can be done linear to the number of transactions. Its main complexity comes from set operations which scale to the number of nodes per transaction. For a basic data set, the average scaling of data fed to Apriori is expected to be linear to a fractional factor of the non-preprocessed size.

5.4 Data Reduction and Processing Time

Figure 3 shows the processing time of the Apriori algorithm. Recorded is the time to generate frequent itemsets. We did not include time to load the database or print the rules since it is less of a reflection on actual CPU time. It is noted that the reduction in data size and rule output gives great increases in speed of the overall process when we do include loading and printing times that become a factor in practical applications. As seen, UNIC outperforms ARM by a factor

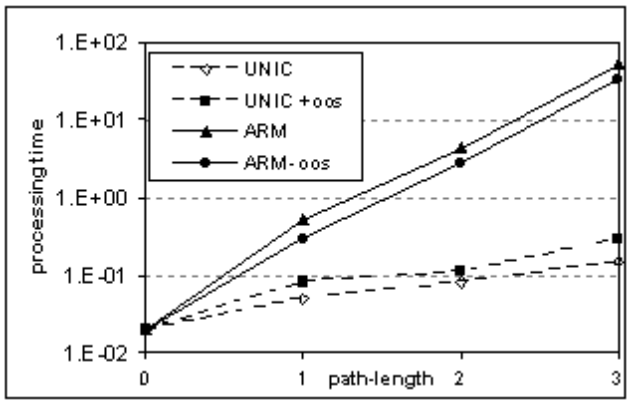


Figure 3: Reduction in Processing

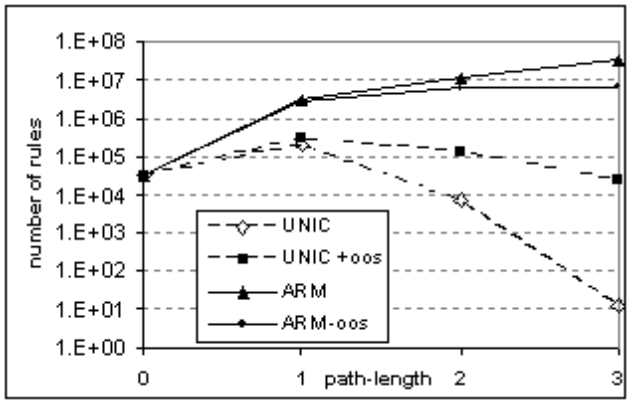


Figure 4: Reduction in Number of Rules

of 100 in the 3-hop setting. It is also worth noting that the improvement can be viewed in accumulative fashion since users are likely to want results from each hop setting.

Results which show a reduction in the width of each transaction in terms of average number of items in the dataset are reported in Figure 5. This is a result of the item removal preprocessing to avoid boring itemsets and the out-of-scope tuple pruning. As seen in 3-hop, the dataset can be reduced by a relative factor of 4 considering the reduction in items with reduction in transactions, which translates directly into lower computation. It can also be observed that this factor is related to the interaction graph and cannot be extrapolated from previous results alone since without the graph data we cannot predict the number of paths which branch or terminate. Interesting enough also is that the average width of transactions actually slightly drops in the 3-hop UNIC setting compared to the 2-hop setting.

5.5 Reduction in Rule Output

A comparison of the volume of rules between standard ARM and UNIC ARM is shown in Figure 4. We can see very significant reduction in the number of rules produced. It is a great improvement when considering an analyst would be faced with a rule set on the order of 30 million in order to discover the same order of ten rule set under our conditions,

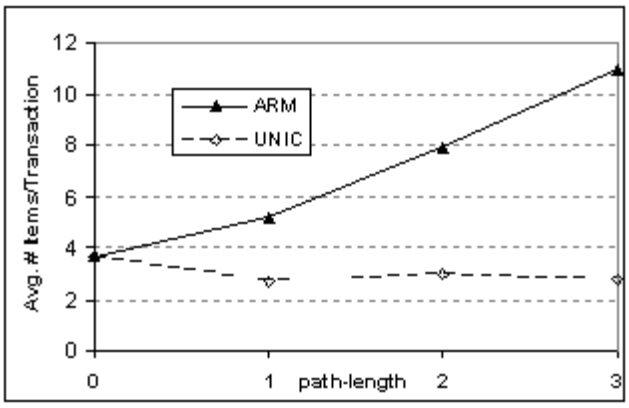


Figure 5: Reduction in Average Items

that's a factor of 10^5!

Even when we match processing by limiting the rule output of standard ARM to the correct scope as (ARM -oos), we reduce rule output by a factor of 10^3. Our algorithm includes more than just post-filtering. Where dealing with symmetry and out-of-scope cases has to be done in post-processing of standard ARM, UNIC does this in pre-processing at a reduced cost. Our algorithm is shown to produce the same results as standard ARM in a non-relational setting as indicated by the same results for the 0-hop table which is not the result of a relational join. The complexity of our method is also the same for this base level as standard ARM.

Further analysis of Figure 4 reveals that the number of out-of-scope rules which can be viewed as irrelevant is large at any level as seen by the difference in trials run with out-of-scope rules included or excluded. UNIC's number of rules decreases as the hop neighborhood increases and support and confidence are held constant. This fact offers insight into the characteristics of the graph itself and causes beneficial scaling. We were able to run UNIC until no results were shown and because this reflects the diameter of the graph we are mining.

5.6 Readability and Information Content

As we presented before, rules mined from a standard ARM setting can be difficult to read and interpret. Here are some examples of the transformation that occurs in a rule set as we apply UNIC to it. As you can see in the first case Table 4, there are many rules that are redundant and we cannot determine what influences are due to the network itself or only due to node characteristics alone. Usually there is an elevated amount of repeated data which is very hard to sort through even with the aid of additional post-processing which might attempt to cluster rules. The rules here all involve *TRANS* and *nucl* to such extent that we cannot judge the difference. We can see an instance of "transitive" rule $\{0.TRANS\}\rightarrow\{1.nucl\}$ since the rules $\{0.TRANS\}\rightarrow\{1.TRANS\}$ and $\{1.TRANS\}\rightarrow\{1.nucl\}$ can be seen as strong.

Even after performing trivial operations to remove obvious problems Table 5 we are left with two situations which can

Table 4: Standard ARM Rules

RHS		LHS	Supp	Conf
1.nucl	→	0.nucl	14.73%	69.45%
0.nucl	→	1.nucl	14.73%	67.21%
0.TRANS	→	0.nucl	13.51%	80.54%
0.nucl	→	0.TRANS	13.51%	61.66%
1.TRANS	→	1.nucl	13.49%	83.41%
1.nucl	→	1.TRANS	13.49%	63.60%
1.TRANS	→	0.TRANS	11.41%	70.58%
0.TRANS	→	1.TRANS	11.41%	68.02%
0.TRANS	→	1.nucl	11.23%	66.97%
0.nucl	→	0.letha	11.61%	57.23%

Table 5: Modified ARM Rules

RHS		LHS	Supp	Conf
1.TRANS, 0.nucl	→	1.nucl	9.79%	88.56%
1.TRANS, 1.nucl	→	0.nucl	9.79%	72.61%
1.nucl, 0.nucl	→	1.TRANS	9.79%	66.49%
0.Tcmplx, 1.TRANS	→	0.TRANS	5.11%	96.19%
0.Tcmplx, 0.TRANS	→	1.TRANS	5.11%	87.83%
1.Tcmplx, 0.TRANS	→	0.nucl	5.06%	87.72%
0.Tcmplx, 0.TRANS	→	1.nucl	5.06%	86.96%
0.Tcmplx, 1.nucl	→	0.TRANS	5.06%	84.39%
1.Tcmplx, 0.nucl	→	0.TRANS	5.06%	81.30%

be misleading or at the least minimally informative. Here the results presented are better than the standard ARM case. The only effect modified ARM offers is slight reduction in redundancy and irrelevant rules. We still have the case were a rule such as identified in Table 4 in this setting is strong and in the UNIC setting is weak. After analysis the difference was found to be because the items involved were very strong on a non-relational level.

When we look at the rule set which comes from UNIC (Table 6) we notice that trivial redundancy has been removed and the information content of each particular rule is higher and more intuitive. The first thing to notice is that the variety of results has changed even from a comparable standard approach that was modified. The top mixed transitive rule in standard ARM $\{0.TRANS\} \rightarrow \{1.nucl\}$ ranks very low in UNIC at support of 0.4% and confidence of 7.5%. The reason for this change comes from the fact that in most cases where the rule occurs in standard ARM it is trivial. Scores reported in UNIC reflect the non-trivial influence of the rule, which in this case is not significant.

Table 6: UNIC ARM Rules

RHS		LHS	Supp	Conf
1.META, 0.Ccd	→	0.CCaDP	0.66%	74.29%
1.cyto, 0.TRANS	→	0.nucl	0.66%	70.27%
1.cyto, 0.nucl	→	0.TRANS	0.66%	54.17%
1.META, 0.CCaDP	→	0.Ccd	0.66%	53.06%
1.cyto, 0.nucl	→	0.lethal	0.63%	52.08%
0.META, 1.TRANS	→	1.nucl	0.56%	70.97%
1.TRANS, 0.cyto	→	1.nucl	0.56%	68.75%
1.CF, 0.Ccd	→	0.lethal	0.56%	62.86%
0.META, 1.nucl	→	1.TRANS	0.56%	51.16%
0.cyto, 1.nucl	→	1.TRANS	0.56%	50.00%

6. CONCLUSIONS

In this paper we have shown that dealing with relational interaction data is difficult problem. There are many aspects which are computationally expensive or that lead to unreadable results. Through the introduction of the UNIC ARM algorithm we were able to reduce the complexity of the association rule search space, reduce the volume of rule output, and increase the information content of produced rules. UNIC can take advantage of many existing and developing ARM core strategies and is able to open a path towards exposing the hidden associations of relations.

Relational interaction and link data is a relatively new topic area and we feel that such advancements will make deeper understanding of the data possible in more effective and efficient ways. Future works will include adaptations to handle relations between different entities. Other extensions will include dealing with item hierarchies and other complex data in the data sets.

7. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. #01322899.

8. REFERENCES

[1] Cygd. *http://mips.gsf.de/genre/proj/yeast/index.jsp*, January 2003.

[2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. S. P. Chen, editors, *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press.

[4] A. L. Barabasi and E. Bonabeau. Scale-free networks. *Scientific American*, 288(5):60, 2003.

[5] Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal. Mining minimal non-redundant association rules using frequent closed itemsets. *Lecture Notes in Computer Science*, 1861:972–??, 2000.

[6] C. Borgelt. Apriori. *http://fuzzy.cs.uni-magdeburg.de/~borgelt/software.html*, August 2003.

[7] D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.

[8] L. Cristofor and D. Simovici. Mining association rules in entity-relationship modeled databases, 2001.

[9] L. Cristofor and D. Simovici. Generating an informative cover for association rules, 2002.

[10] L. Dehaspe and L. D. Raedt. Mining association rules in multiple relations. In S. Džeroski and N. Lavrač, editors, *Proceedings of the 7th International Workshop on Inductive Logic Programming*, volume 1297, pages 125–132. Springer-Verlag, 1997.

[11] A. H. Y. T. et al. Global mapping of the yeast genetic interaction network. *Science*, 303(5695):808–815, 2004.

[12] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *IJCAI*, pages 1300–1309, 1999.

[13] M. HW, F. D, G. U, M. G, M. K, M. M, M. B, M. M, R. S, and W. B. Mips: a database for genomes and protein sequences. *Nucleic Acids Research*, 30((1)):31–4, 2002.

[14] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Principles of Data Mining and Knowledge Discovery*, pages 13–23, 2000.

[15] D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *ICML2002*, pages 259–266, 2002.

[16] A. J. Knobbe, H. Blockeel, A. P. J. M. Siebes, and D. M. G. van der Wallen. Multi-relational data mining. Technical Report INS-R9908, Maastricht University, 9, 1999.

[17] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *ICDM*, pages 313–320, 2001.

[18] S. A. Macskassy and F. Provost. A simple relational classifier. *2nd Workshop on Multi-Relational Data Mining (MRDM-2003) at KDD-2003*, 2003.

[19] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan mining sequential patterns efficiently by prefix projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering*, pages 215–226, 2001.

[20] O. T., K. K., S. K., and I. T. Extraction of knowledge on protein-protein interaction by association rule discovery. *Bioinformatics*, 18(8):705–14, 2002.

[21] A. Wagner and D. A. Fell. The small world inside large metabolic networks. In *Proceedings of the Royal Society: Biological Sciences*, pages 1803–10, 9 2001.

[22] X. Yan and J. Han. gspan: Graph-based substructure pattern mining, 2002.

[23] X. Yan, J. Han, and R. Afshar. Clospan: Mining closed sequential patterns in large datasets. In *Proceedings 2003 SIAM Int.Conf. on Data Mining*, 2003.

[24] M. J. Zaki. Generating non-redundant association rules. In *Knowledge Discovery and Data Mining*, pages 34–43, 2000.

[25] M. J. Zaki. SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1/2):31–60, 2001.