Multi-Relational secured in Data Mining

R.Sudha
Department of Computer Science
Adhiparasakthi College of arts and science
Prof.J.Srinivasan
Department of Computer Science
Adhiparasakthi College of arts and science
Dr.G.Arutchelvan
Director and Head, Department of Computer Science and Applications

***Abstract*— Multi-Relational Data Mining (MRDM).Building on relational database theory is an obvious choice, as most data-intensive applications of industrial scale employ a relational database for storage and retrieval. But apart from this pragmatic motivation, there are more substantial reasons for having a relational database view on Structured Data Mining. Relational database theory has a long and rich history of ideas and developments concerning the efficient storage and processing of structured data, which should be exploited in successful Multi-Relational Data mining technology. Concepts such as data modeling and database normalization may help to properly approach an MRDM project, and guide the effective and efficient search for interesting knowledge in the data. Recent developments in dealing with extremely large databases and managing query-intensive analytical processing will aid the application of MRDM in larger and more complex domains.**

Keywords — Multi-Relational Data Mining, relational database, Structured Data Mining, MRDM, data modeling.

I INTRODUCTION

This thesis is concerned with Data Mining: extracting useful insights from large and detailed collections of data. With the increased possibilities in modern society for companies and institutions to gather data cheaply and efficiently, this subject has become of increasing importance. This interest has inspired a rapidly maturing research field with developments both on a theoretical, as well as on a practical level with the availability of a range of commercial tools. Unfortunately, the widespread application of this technology has been limited by an important assumption in mainstream Data Mining approaches. This assumption – all data resides, or can be made to reside, in a single table – prevents the use of these Data Mining tools in certain important domains, or requires considerable massaging and altering of the data as a pre-processing step. This limitation has spawned a relatively recent interest in richer Data Mining paradigms that do allow structured data as opposed to the traditional flat representation.

1.1 Structured Data in Relational Form

Structured data will always be represented in a relational database by multiple tables. The information concerning the different parts of an individual will be distributed over these tables. There is one particular table, which we will refer to as the *target table*, that has a special role. Each record in this table corresponds to exactly one individual. The target table will be connected to other tables through foreign key relations. By following these keys the remaining data concerning individuals may be looked up. The target table will always be the starting point for searching interesting patterns, as patterns represent sets of individuals. The target table will often contain attributes that describe individuals as a whole, but may also just contain a single key-attribute that points to the structural parts in the remaining tables. Each table contains parts belonging to one particular class. All parts of this class, regardless of the individual, appear in the same table. In order to determine the individual that a part belongs to, or to collect all parts belonging to a given individual, one will have to join over the foreign key relations. Figure1 and Figure 2 give an example of how molecules may be stored in a relational database for multi-relational analysis. Figure 1 demonstrates how carbon dioxide consists of six parts in three classes: one molecule, three atoms and two bonds. In Figure 2 these parts are distributed over three tables that also contain data for other molecules, e.g. water.

Figure 1 : Relational representation of CO_2 and H_2O.

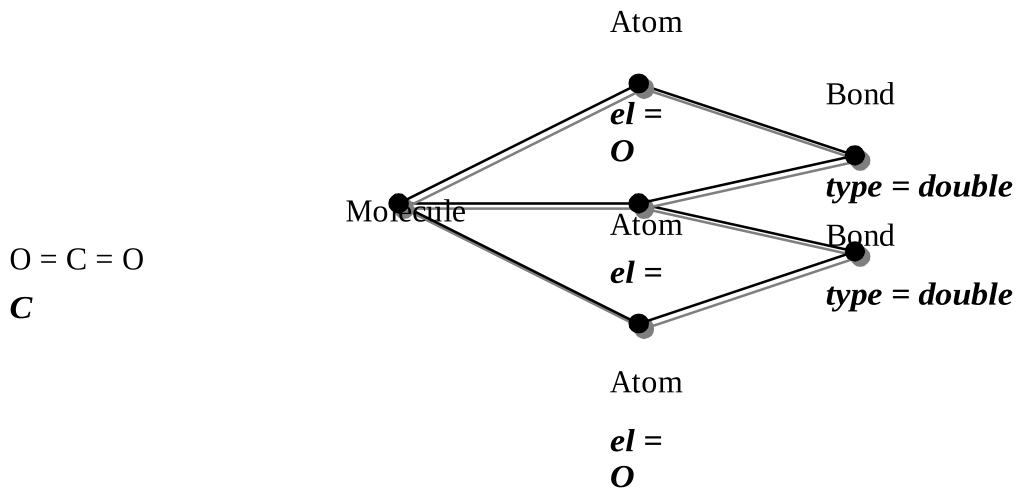

Figure 2 CO_2 and its graphical representation.

Molecule

id	name
1	CO_2
2	H_2O

Atom

id	mol_id	element
1	1	C
2	1	O
3	1	O
4	2	H
5	2	H
6	2	O

Bond

id	atom1	atom2	type
1	1	2	double
2	1	3	double
3	4	6	single
4	5	6	single

1.2 Propositionalisation

In this section we describe the basic concepts involved in propositionalisation, and provide some definitions. We define propositionalisation as the process of transforming a multi-relational dataset into a propositional dataset with derived attribute-value features, describing specific structural properties of the individuals. The process can thus be thought of as summarising data stored in multiple tables in a single table (the *target table*) containing one record per individual. The aim of this process, of course, is to pre-process multi-relational data for subsequent analysis by attribute-value learners.

2. The RollUp Algorithm

The algorithm will traverse the data model graph and repeatedly use the aggregation operation to project data from one table onto another, until all information has been aggregated at the target table. Although this repeated summarisation can be done in several ways, we will describe a basic algorithm, called RollUp. The RollUp algorithm performs a depth-first search (DFS) through the data model, up to a specified depth. Whenever the recursive algorithm reaches its maximum depth or a leaf in the graph, it will "roll up" the relevant table by aggregating it onto the parent in the DFS tree. Internal nodes in the tree will again be aggregated after all its children have been aggregated. This means that attributes considered deep in the tree may be aggregated multiple times. The process continues until all tables are summarised onto the target table. Combined with a propositional learner we obtain an instance of Polka. The following pseudo code describes RollUp more formally:

```
RollUp (table T, data model M, integer d)

        V = T
        if d
        □□0
                for all associations A from T in M
                        W = RollUp(T.getTable(A), M, d-
                        1) S = Aggregate(W, A)
                        V.add(s
        ) return V
```

The effect of RollUp is that each attribute appearing in a table other than the target table will appear several times in aggregated form in the resulting view. This multiple occurrence happens for two reasons. The first reason is that tables may occur multiple times in the DFS tree because they can be reached through multiple paths in the data model. Each path will produce a different aggregation of the available attributes. The second reason is related to the choices of aggregate function class at each aggregation along a path in the data model. This choice, and the fact that aggregate functions may be combined in longer paths, produces multiple occurrences of an attribute per path. The association-depth of the deepest feature is equal to the parameter *d*. Each feature corresponds to at most one attribute aggregated along a path of depth d_a. The refinement-depth is therefore a linear function of the association-depth. As each feature involves at most one attribute, and is aggregated along a path with no branches, the association-width will always be either 0 or 1. This produces the following characteristics for RollUp. Use Lemma to characterise Polka instantiated with RollUp.

Lemma

1. d_a(RollUp) = *d*
2. d_r(RollUp) = d_a(RollUp) +

1 3. w_a(RollUp) = 1

2.1 Related Work

Two alternative MRDM approaches implement propositionalisation by means of aggregate functions. The RELAGGS system [68, 69] uses a slightly larger collection of aggregate function classes to summarise groups of records. The essential difference with RollUp however, is where aggregation is applied. Rather than recursively joining and aggregating pairs of tables towards the target table, RELAGGS propagates the key of the individual to each table, effectively turning the data model into a star schema (association-depth 1). It then applies aggregate functions to all tables, as if directly connected to the target table. This process produces the same result (given the same collection of aggregate function classes) whenever a database of association-depth 1 or less is processed. On association-deeper data models however, RollUp will nest aggregate functions (e.g. average of the count), whereas RELAGGS will aggregate over the transitive association.

2.2 Aggregate Functions & Rule Discovery

The experiments in the previous chapter demonstrate the power of aggregate functions. Particularly on databases that combine a high level of non-determinacy with large numbers of (numeric) attributes, these aggregate functions are a promising means of capturing local structure. Unfortunately, the propositionalisation method in which we embedded these functions has a disadvantage that reduces their potential. This has to do with the static nature of the features produced in the propositionalisation step. The small set of aggregate functions employed produces a moderate set of fixed features which cannot be modified when the actual mining takes place. The overall structure of the Rule Discovery method remains unchanged. The primary changes deal with the richer pattern language and the refinement operator that introduces aggregate functions into a given pattern. We will show that extra care needs to be taken in order to guarantee that the refinement operator is actually a specialisation operator. Catering to aggregation also requires more complex data mining primitives.

2.3 Experiments

In order to acquire empirical knowledge about the effectiveness of our approach, we have tested RollUp on three well-known multi-relational datasets. These datasets were chosen because they show a variety of data models that occur frequently in many multi-relational problems. They are Musk [27], Mutagenesis [95], and Financial [106].
Each dataset was loaded into the RDBMS Oracle. The data was modelled in MRML. Based on this declarative bias, the RollUp module produced one database view for each dataset, containing the propositionalised data. This was then taken as input for the common Machine Learning procedure C5.0. For quantitative comparison with other techniques, we have computed the average accuracy by leave-one-out cross-validation for Musk and Mutagenesis, and by 10-fold cross-validation for Financial.

3. Generalised Selection Graphs

In this section, we will show how aggregate functions are a natural generalisation of the existential conditions represented by edges in a selection graph. To support aggregate functions with selection graphs, we have to extend the language with a selection mechanism based on local structure. In particular, we add the possibility of *aggregate conditions*, resulting in *generalised selection graphs* (GSG).

Definition 3.1 An *aggregate condition* is a triple (*f*, □, *v*) where *f* is an aggregate function, □□ a comparison operator, and *v* a value of the codomain of *f*.

Definition 3.2 A *generalised selection graph* is a directed graph (*N*, *E*), where *N* is a set of triples (*t*, *C*, *s*), *t* is a table in the data model and *C* is a, possibly empty, set of conditions on attributes in *t* of type (*t.a* □□ *c*); □□ one of the following operators, =, □□, □. The flag *s* has the possible values *open* and *closed*. *E* is a set of tuples (*p*, *q*, *a*, F) where *p*, *q* □□ *N*, *a* is an association between *p.t* and *q.t* in the data model, and *F* is an aggregate condition. The generalised selection graph contains at least one node n_0 (the *root-node*) that corresponds to the target table t_0.We will use the same recursive construct in a translation procedure for SQL, as follows. If we start at the leaves of the graph and work back to the root, respecting all the selection conditions, we can compute the selection of records in the target table. This is achieved as follows. First we produce a list of groups of records in a table *Q* at a leaf node by testing on the aggregate condition. Each group is identified by the value of the foreign key:

```
SELECT foreign-
key FROM Q
WHERE  attribute-conditions
GROUP  BY foreign-key
HAVING aggregate-
condition
```

We then join the result *Q'* with the parent table *P* to obtain a list of records in *P* that satisfies the combined conditions in the edge and leaf node:

```
SELECT P. primary-
key FROM P, Q'
WHERE P. primary-key = Q'. foreign-key
```

This process continues recursively up to the root-node, resulting in a large query of nested SELECT statements. The second query can be extended with the grouping construct of the first query for the next edge in the sequence. This results in exactly one SELECT statement per node in the graph. This process is formalised in Figure 3.3

```
SelectSubGraph (node n)

        S = 'SELECT  ' + n.Name( ) +
        '.' if (n.IsRootNode( ))
                S.add (n.PrimaryKey(
        )) else
                S.add (n.ForeignKey(
        )) S.add (' FROM ' + n.Name(
        )) for each child i of n do
                        $S_i$ =
                  SelectSubGraph(i)
                S.add(', ' + $S_i$ + ' S' + i
                        )
        S.add(' WHERE ' + n.AttributeConditions(
        )) if (!n.IsLeaf( ))
                for each child i of n do
                        S.add (' AND ' + n.Name( ) + '.' + n.PrimaryKey( )
                                + ' = S' + i + '.' + i.ForeignKey( ))
        if (! n.IsRootNode( ))
                S.add(' GROUP BY ' + n.Name( ) + '.' + n.ForeignKey( ))
                S.add(' HAVING ' + n.ParentEdge( ).AggregateCondition(
                ))
        return S
```

Figure 9.1 The *SelectSubGraph* algorithm.

CONCLUSION

Multi-Relational Data Mining is inherently more powerful than Propositional Data Mining. There clearly is a large class of Data Mining problems that cannot be successfully approached using a single table as representational setting. These problems, which can be characterised by the presence of internal structure within the individuals they deal with, can successfully be approached by the multi-relational tools and techniques that are the subject of this thesis. MRDM techniques are not the only ones that deal with structured data. We have presented a genus of Structured Data Mining paradigms that each approach the representation of data, and consequently the manipulation and analysis of the database, from a unique 'tradition'. Although our main emphasis has been on MRDM, we recognise the value of approaching problems in the more abstract setting of Structured Data Mining. By combining achievements that have been made relatively independently of one another, a richer set of techniques becomes available, and redundant development can be prevented. Furthermore a unified approach aids the comparison of existing techniques, which are mainly representation-specific. We therefore see the generalization of techniques from the individual paradigms, and integration of common ideas in SDM, as an important direction for future research.

REFERENCES

1. Abe, K., Kawasoe, S., Asai, T., Arimura, H., Arikawa, S. *Optimized Substructure Discovery for Semi-Structured Data*, In Proceedings of PKDD 2002, LNAI 2431, 2002
2. Abiteboul, S., Buneman, P., and Suciu, D. *Data on the Web*, Morgan Kaufmann, 2000
3. Adriaans, P. *Adaptive System Management*, in Advances in Applied Ergonomics, In Proceedings of ICAE'96, Istanbul, 1996
4. Adriaans, P., Zantinge, R. *Data mining*. Addison-Wesley, 1996
5. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A. *Fast Discovery of Association Rules*, in Artificial Intelligence, 1997
6. Alphonse, É., Rouveirol, C. *Selective Propositionalization for Relational Learning*, In Proceedings of
7. Hahn, M. *Info Charger Data Sheet*, http://www.tektonic.de/ icdatash.htm, 1997
8. Hand, D., Mannila H., Smyth, P., *Principles of Data Mining*, MIT Press, 2001
9. Herlaar, L. *Diagnosing Performane of Relational Database Managament Systems*, technical report, Utrecht University, 1995
10. Holsheimer, M., Kersten, M., Mannila, H., Toivonen, H. *A Perspective on Databases and Data Mining*, In Proceedings of KDD '95, 1995
11. Holsheimer, M., Kersten, M., Siebes, A. *Data Surveyor: Searching the nuggets in parallel*, In [33]
12. Ibraheem, S., Kokar, M., Lewis, L. *Capturing a Qualitative Model of Network Performance and Predictive Behavior*, Journal of Network and System Management, vol 6, 2, 1997
13. Inokuchi, A., Washio, T. and Motoda, H. *An Apriori-Based Algorithm for Mining Frequent Substructures from Graph Data*, In Proceedings of PKDD 2000, LNAI 1910, 2000
14. Jensen, D., Neville, J., Hay, M. *Avoiding Bias when Aggregating Relational Data with Degree Disparity*, In Proceedings ICML-2003, 2003.
15. John, G.H., Lent, B. *SIPping from the Data Firehose*, In Proceedings of KDD '97, 1997