USING dmFSQL FOR FINANCIAL CLUSTERING

Ramón Alberto Carrasco
Dpto. de Lenguajes y Sistemas Informáticos, Universidad de Granada, Granada, Spain
Email: rcarrasco@caja-granada.es

Maria Amparo Vila
Dpto. de Ciencias de la Computación e I.A., Universidad de Granada, Granada, Spain
Email: vila@decsai.ugr.es

José Galindo
Dpto. de Lenguajes y Ciencias de la Computación, Universidad de Málaga, Spain
Email: ppgg@lcc.uma.es

Keywords: Clustering, Flexible Queries, Data Mining, Fuzzy SQL, Fuzzy Databases.

Abstract: At present, we have a dmFSQL server available for Oracle© Databases, programmed in PL/SQL. This server allows us to query a Fuzzy or Classical Database with the dmFSQL (data mining Fuzzy SQL) language for any data type. The dmFSQL language is an extension of the SQL language, which permits us to write flexible (or fuzzy) conditions in our queries to a fuzzy or traditional database. In this paper, we propose the use of the dmFSQL language for fuzzy queries as one of the techniques of Data Mining, which can be used to obtain the clustering results in real time. This enables us to evaluate the process of extraction of information (Data Mining) at both a practical and a theoretical level. We present a new version of the prototype, called DAPHNE, for clustering which use dmFSQL. We consider that this model satisfies the requirements of Data Mining systems (handling of different types of data, high-level language, efficiency, certainty, interactivity, etc) and this new level of personal configuration makes the system very useful and flexible.

1 INTRODUCTION

We can define Data Mining as the process of extraction of interesting information from the data in databases. According to (Frawley 1991) a discovered knowledge (pattern) is interesting when it is novel, potentially useful and non-trivial to compute. A serie of new functionalities exist in Data Mining, which reaffirms that it is an independent area (Frawley 1991):

- High-Level Language. This representation is desirable for discovered knowledge and for showing the results of the user's requests for information (e.g. queries).
- Certainty. The discovered knowledge should accurately reflect the content of the database. The imperfectness (noise and exceptional data) should be expressed with measures of certainty.
- Efficiency. The process of extraction of knowledge should be efficient, i.e. the running time should be predictable and acceptable when dealing with very large amounts of data.
- Handling of Different Types of Data. There are different kinds of data and databases used in diverse applications (relational data, objects, hypertext, etc.) so it would be desirable that a Data Mining system would carry out its work in an effective way.
- Interactive Mining Knowledge at Multiple Abstraction Levels. The interactive discovery of knowledge allows the user to refine a Data Mining request on line, dynamically change data focusing, progressively deepen a Data Mining process, and flexibly view the data and Data Mining results at multiple abstraction levels and from different angles.
- Mining Information from Different Sources of Data. Currently the knowledge mining from different sources of formatted or unformatted data with diverse data semantic is perceived to be a difficult challenge.
In this paper we discuss the implementation of two prototypes for Data Mining purposes: we have used a combination of DAPHNE which was initially designed for clustering on numeric data types (Carrasco, 1999) and dmFSQL which was designed for fuzzy (or flexible) queries (Galindo 1998, Galindo 1998b, Galindo 1999). At this point, we would like to point out that Data Mining is an autonomous and self-interesting field of research, in which techniques from other fields could be applied. Among these techniques are the use of dmFSQL (data mining Fuzzy SQL), which is a database query language which incorporates fuzzy logic. In particular, we use dmFSQL to solve, in real time, queries, which obtain objects (tuples) with similar characteristics, i.e. objects of a specific group through a process of clustering. Often, the clustering is carried out on a set of examples from the database and not on the entire database. We present some experimental results with this alternative solution in the context of a bank. This area needs a Data Mining system tailored to its needs, because this area manages very large databases and these data has a very concrete meaning. Thus, data must be treated according to this meaning. Finally, as conclusions we consider that this model satisfies the requirements of Data Mining systems (Chen 1996, Frawley 2001) (handling of different types of data, high-level language, efficiency, certainty, interactivity, etc.) and this new level of personal configuration makes the system very useful and flexible.

2 dmFSQL A LANGUAGE FOR FLEXIBLE QUERIES

The dmFSQL language (Galindo 1998, Galindo 1998b, Galindo 1999) extends the SQL language to allow flexible queries. We have extended the SELECT command to express flexible queries and, due to its complex format, we only show an abstract with the main extensions added to this command:

- **Linguistic Labels**: If an attribute is capable of undergoing fuzzy treatment then linguistic labels can be defined on it. These labels will be preceded with the symbol $ to distinguish them easily. They represent a concrete value of the attribute. dmFSQL works with any kind of attributes (see 2.1.1 section) therefore, by example, a label can have associated: a trapezoidal possibility (Figure 1), a scalar (if there is a similarity relationship defined between each two labels in the same domain), a text, a XML document, etc.

- **Fuzzy Comparators**: In addition to common comparators (=, >, etc.), dmFSQL includes fuzzy comparators in Table 1. There are some different kinds of fuzzy comparators. By example a fuzzy comparator is used to compare two trapezoidal possibility distributions $A, B$ with $A=\{\alpha_A, \beta_A, \gamma_A, \delta_A\} B=\{\alpha_B, \beta_B, \gamma_B, \delta_B\}$ (see Figure 1). In the same way as in SQL, fuzzy comparators can compare one column with one constant or two columns of the same type. More information can be found in (Galindo 1998b, Galindo 1999). These definitions can are based in fuzzy set theory, classical distance functions and other type of similarity functions.

- **Fulfilment Thresholds $\gamma$**: For each simple condition a Fulfilment threshold may be established with the format <condition> THOLD $\gamma$, indicating that the condition must be satisfied with a minimum degree $\gamma$ in [0,1] fulfilled.

- **CDEG(<attribute>) function**: This function shows a column with the Fulfilment degree of the condition of the query for a specific attribute, which is expressed in brackets as the argument.

- **Fuzzy Constants**: We can use and store all of the fuzzy constants (which appear in Table 2) in dmFSQL.

![Figure 1: Trapezoidal possibility distributions: A, B.](114)

<table>
<thead>
<tr>
<th>Fuzzy Comparator (fcomp) for:</th>
<th>Possibility</th>
<th>Necessity</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEQ</td>
<td>NF EQ</td>
<td>Fuzzy EQual</td>
<td></td>
</tr>
<tr>
<td>FGT</td>
<td>NF GT</td>
<td>Fuzzy Greater Than</td>
<td></td>
</tr>
<tr>
<td>FGEQ</td>
<td>NF GEQ</td>
<td>Fuzzy Greater or Equal</td>
<td></td>
</tr>
<tr>
<td>FLT</td>
<td>NF LT</td>
<td>Fuzzy Less Than</td>
<td></td>
</tr>
<tr>
<td>FLEQ</td>
<td>NF LEQ</td>
<td>Fuzzy Less or Equal</td>
<td></td>
</tr>
<tr>
<td>MGT</td>
<td>NM GT</td>
<td>Much Greater Than</td>
<td></td>
</tr>
<tr>
<td>MLT</td>
<td>NM LT</td>
<td>Much Less Than</td>
<td></td>
</tr>
</tbody>
</table>

- **Fulfilment Thresholds $\gamma$**: For each simple condition a Fulfilment threshold may be established with the format <condition> THOLD $\gamma$, indicating that the condition must be satisfied with a minimum degree $\gamma$ in [0,1] fulfilled.

- **CDEG(<attribute>) function**: This function shows a column with the Fulfilment degree of the condition of the query for a specific attribute, which is expressed in brackets as the argument.

- **Fuzzy Constants**: We can use and store all of the fuzzy constants (which appear in Table 2) in dmFSQL.