Draft of paper In: International Journal of Human-Computer Studies, Vol 58 (2003) 363-391

Fuzzy Query Interface for a Business Database

Rita A. Ribeiro
Universidade Lusíada
Department of Management
Rua da Junqueira, 192
1349-001 Lisboa, Portugal
Email: rar@uninova.pt

Ana M. Moreira
Departamento de Informática
Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa
2825 Monte Caparica, Portugal
Email: amm@di.fct.unl.pt

Abstract: Managers, in today's corporations, rely increasingly on the use of databases to obtain insights and updated information to make their decisions. This paper describes a flexible query interface based on fuzzy logic. Hence, queries in natural language with pre-defined syntactical structures are performed, and the system uses a fuzzy natural language process to provide answers. This process uses the fuzzy translation rules of the meaning representation language PRUF, proposed by [Zadeh, 1978 #735]. The interface was built for a relational database of the 500 biggest non-financial Portuguese companies. The attributes considered are the economic and financial indicators. Examples of pseudo natural language queries, such as "*is company X very profitable?*" or "*are most private companies productive?*", are presented to show the capabilities of this human-oriented interface.

KEYWORDS: fuzzy logic, flexible queries, pseudo natural language, fuzzy adverbs, economic and financial indicators.

1. Introduction

1. 1. Why do we need flexible interfaces?

Many corporations have been storing enormous amounts of data, in databases, for a long time. More and more people, from experts to non-experts, are depending on information from databases to fulfill everyday tasks. These facts are leading to improvements on available languages to query databases so as to access information in a more human-oriented fashion. The objective is not to obtain more information but better information, in the sense of having filtered useful information instead of huge quantities of raw data. Managers, in today's corporations, rely increasingly on the use of databases to obtain insights and updated information to make their decisions.

The motivation for this work is to develop a flexible database query interface, for a relational database, that allows queries in pseudo-natural language, such as "*is IBM a dynamic company*?" or "*is Dan-Cake very profitable?*". Specifically, we describe a flexible human-oriented fuzzy query interface, developed to help users obtain intelligent information about the 500 biggest non-financial Portuguese companies. The Portuguese magazine [EXAME, 1996 #843] provided a table with information about the financial and economic indicators about those companies. The indicators are ordered by net sales volume and those selected are the ones with net sales above 27.4 million dollars. The magazine also described other possible financial and economic aggregated indicators for those firms. We used a description of aggregated indicators (i.e. associations of attributes) to group the given indicators into four main categories: dynamism, financial health, profitability and economic contribution. These categories reflect different perspectives of managers, bankers, stockholders and government. Our idea is to offer a broader view, to general users, of the best Portuguese firms. Obviously, the user can query either each indicator (henceforth denoted attribute) or a group of them (henceforth denoted association).

1. 2. How to deal with imprecision in query interfaces

An important issue in developing human-oriented interfaces has been how to allow non-expert users to query databases in a natural way. The main problem is that natural language includes many kinds of imprecise statements that are not compatible with the strict structure of relational databases. Many times when querying a database the users do not wish to define the precise limits of acceptance or rejection for a condition, that is, they want to be allowed some imprecision in the query. In other words, the satisfaction of a condition is a matter of degree [Bosc, 1993 #856] and a flexible query should provide answers that would have had an empty response on a classical relational SQL-type language. Moreover, it can easily rank-order the best answers, rather than showing a long list of answers.

Fuzzy set theory is a useful tool to handle imprecision [Zadeh, 1965 #722]. The application of this theory in the area of fuzzy databases, to deal with imprecision and vagueness, has been widely addressed in the literature (see for example [Baldwin, 1985 #788], [Buckles, 1985 #120], [Bosc, 1995 #833], [Bosc, 1997 #852], [Galindo, 2000 #929], [Prade, 1984 #854], [Kacprzyk, 1995 #797], [Kacprzyk, 1986 #796], [Kacprzyk, 2001 #935], [Takahashi, 1995 #803]). According to Dubois and Prade [Dubois, 1997 #853], fuzzy database research can be divided in three main areas of research: flexible querying; handling imprecise or fuzzy data; handling fuzzy dependencies. Further, [Takahashi, 1991 #801] categorizes fuzzy querying research in two classes of approaches: (a) one that extends relational database query languages, to deal with fuzzy propositions while the database model remains relational; and (b) another that extends relational databases to develop fuzzy databases as well as query languages.

Here we only address the problematic of building a human-oriented interface capable of handling flexible queries, since our focus is on the representation of attributes by means of fuzzy sets to allow pseudo-natural language queries. The approach we follow is to represent vagueness, involved in the requests, by fuzzy sets, and when the flexible queries are done the answers are processed by the fuzzy translation rules of PRUF (Possibilistic Relational Universal Fuzzy Language) [Zadeh, 1978 #735]. We follow this approach because, as recognized by [Takahashi, 1995 #803] [Takahashi, 1991 #801], flexible interfaces allows the use of vast amounts of useful information, already stored in relational databases all over the world. He also proposed a fuzzy query language ([Takahashi, 1991 #801], [Takahashi, 1995 #803]) based on the formal language PRUF (Possibilistic Relational Universal Fuzzy) of [Zadeh, 1978 #735]. His language contains a set of fuzzy propositions that can be easily incorporated in the relational database queries. Further, [Takahashi, 1995 #803] recognized that PRUF includes most types of propositions usually encountered in the condition descriptions of query languages, as can be seen in most of the literature on fuzzy query languages (see, for example, [Bosc, 1995 #833], [Bosc, 2001 #925], [Gonçalves, 2001 #930], [Kacprzyk, 1995 #797], [Kacprzyk, 2001 #935], [Tahini, 1977 #855], [Takahashi, 1995 #803]).

Since many fuzzy query languages include, implicitly or explicitly, the four types of propositions classified by Zadeh and incorporated in his formal language PRUF, we decided to implement directly the original translation rules to achieve a user-friendly interface that can handle pseudo-natural language queries on a business database. The questions are posed in natural language with a pre-defined structure (called pseudo-natural language). The answers are obtained using the translation rules to process the tuples and then they are mapped with a display language to provide qualitative and quantitative answers.

Our proposal includes the simple fuzzy calculus of PRUF and does not addresses more complex operations, as for instance, set-difference of fuzzy relational algebra. Different proposals for set-operations of relational algebra and calculus can be seen in ([Bosc, 1995 #833] [Bosc, 2002 #926], [Buckles, 1985 #120], [Gonçalves, 2001 #930], [Galindo, 2000 #929], [Kacprzyk, 1995 #797], [Prade, 1984 #854], [Takahashi, 1995 #803]). Many of these authors proposed new concepts for fuzzy algebra and calculus to create new fuzzy query languages. However, some of these approaches require changes in the underlying relational database systems. Our aim is only to build an intelligent interface.

1. 3. Our contribution

As mentioned before, in the context of fuzzy query languages, many authors propose extensions to relational algebra in order to develop a fuzzy Structured Query Language (SQL) that provides the means for performing queries with some uncertain concepts and to obtain answers [Bosc, 1995 #833] [Kacprzyk, 1995 #797] [Takahashi, 1995 #803]. However, what most authors did not consider was just to have a fuzzy interface that will be used as a top layer, on an existing relational database, without any modification on its database management system (DBMS).

We developed an interface that allows us to make questions in (quasi) natural language and to obtain answers in the same style, without having to modify neither the structure of the database nor the DBMS query language. That is, we developed an intelligent interface and not a fuzzy query language. The main advantages of our work are:

- The existing implemented systems do not have to be modified;
- The fuzzy attributes are built from the raw data;

- The developers do not have to learn neither a new query language, such as a new extension to SQL, nor new set-operations to maintain legacy systems;
- The dialog with the system is done in a language very close to natural language;
- The answers are given in a linguistic form, as well as a numeric form, which helps the user to better understand the results obtained;
- The interface can be used for other relational databases, after an initial preprocessing to define the fuzzy components with eventual slight changes in the grammar.

In summary, our work shows how to obtain intelligent information, from a business database, using a pseudo-natural language. In order to ask flexible queries we first perform a pre-processing on a relational business database to build fuzzy attributes and adverbs. Second, we build a parser to obtain fuzzy pre-defined syntactical structures (action language) that will be used in the processing of the query. Third, we process the queries using a fuzzy natural language processor, based on the four translation rules of PRUF [Zadeh, 1978 #735]. Finally, we process the answers with a display language to provide the user with pseudo-natural language answers. The most important aim of this work is to show how fuzzy set theory and PRUF provide a good tool to build user-oriented interfaces, capable of dealing with flexible queries which involve imprecision, in a business environment.

It should be noted that this work is an empirical study that represents a “proof-of-concept” and not a final commercial software package, hence we did not perform an evaluation in a real environment. Nevertheless, we believe that the ideas presented in this paper provide the elements for a basic methodology, for business problems, that might be exploited in further and more focused studies.

This paper is organized in five sections. Section 1 is this introduction. Section 2 gives an overview of the basic concepts involved in PRUF. Section 3 discusses in detail the fuzzy querying model implemented, including: the fuzzy components, the database model, the fuzzy natural language processor and the dialog component. Section 4 shows illustrative examples of different types of queries and their answers, as generated by the proposed model. Section 5 provides the concluding remarks of this study.

2. Brief overview of Fuzzy Set Theory

In this Section, we first introduce some basic concepts of fuzzy set theory that form the basis of PRUF and then we describe its main characteristics. PRUF (Possibilistic Relational Universal Fuzzy Language) [Zadeh, 1978 #735] is a general meaning representation language that proposes four main types of translations rules to allow the treatment of fuzzy propositions that could be used in intelligent queries.

2. 1. Basic concepts

Fuzzy set theory was introduced in 1965 by [Zadeh, 1965 #722]. A fuzzy set, or more appropriately a fuzzy subset, is composed of elements that belong, with different degrees of membership to the subset, i.e. it is a subset without a precise boundary. Here we will use the example of Zadeh, *small integers* [Zadeh, 1978 #735], to describe the type of subset used in the PRUF translation rules. The fuzzy subset *small integers* belonging to the subset of integers can be represented by a set of pairs, separated by the plus sign, where the first component of the pair represents the element itself and the second represents the membership value of the element. For example the subset of "small integers" can be represented by the fuzzy concept,

small integer = {0/1 + 1/1 + 2/0.8 + 3/0.4 + 4/0.1}

This concept expresses that the integers {0, 1} fully belong to the subset (membership value of 1), integer {2} belongs with a membership value of 0.8 (belongs strongly), integer {3} belongs with 0.4 (belongs more or less) and integer {4} belongs with 0.1 (belongs very weakly).

Fuzzy Set. Formally, considering the universe U where u is the general element, U = {u}, a fuzzy subset Ã is defined as:

$$\tilde{A} = \left\{(u / \mu_{\tilde{A}}(u)) \mid u \in U\right\} \tag{1}$$

where $\mu_{\tilde{A}}(u)$ is the membership value of u in Ã. The membership function associates each element u of U with a real number $\mu_{\tilde{A}}(u)$, in the interval [0,1]. The main difference of fuzzy set theory from classical set theory is that different "degrees of membership" are allowed. In classical set theory, any element u of U either belongs to the subset (membership value 1) or does not belong to the subset (membership value 0).

Basic Operations. The basic operations on Fuzzy Sets are complement, union and intersection. The complement of *A* is a fuzzy set $\overline{A}$ in U whose membership function can be defined as,

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x) \tag{2}$$

The union of A and B is a fuzzy set in U, whose membership is defined as,

$$\mu_{A \cup B}(x) = \cup\left\lfloor \mu_A(x), \mu_B(x) \right\rfloor \text{ where } \cup \text{ is the max operator.} \tag{3}$$

The intersection of A and B is a fuzzy set in U, whose membership is defined as,

$$\mu_{A \cap B}(x) = \cap\left\lfloor \mu_A(x), \mu_B(x) \right\rfloor \text{, where } \cap \text{ is the min operator.} \tag{4}$$

There are many operators proposed in the literature to perform the operations of intersection, union and complement (for other proposed operators see [Klir, 1988 #378]).

Fuzzy relation. An important concept of fuzzy set theory is the existence, or not, of relations between fuzzy sets. A fuzzy relation, defined in the Cartesian product of the crisp sets $U_1, U_2, ..., U_n$, is a fuzzy set R such that

$$R = \{((u_1, u_2, ..., u_n), \mu_R(u_1, u_2, ..., u_n) \mid (u_1, u_2, ..., u_n) \in U_1 \times U_2 \times ... \times U_n\} \quad (5)$$

where $\mu_R : U_1 \times U_2 \times ... \times U_n \rightarrow [0,1]$. and the crisp Cartesian product of U and V is,

$$UxV = \{(u, v) \mid u \in U \ and \ v \in V\} \quad (6)$$

An example could be the fuzzy relation "more or less close" between two sets of cities, U={Lisbon, Paris} and V={Paris, London} with the fuzzy set R={(Lisbon, Paris)/0.6+ (Lisbon,London)/0.2+ (Paris, Paris)/1+ (Paris, London)/0.8}.

Projection and cylindrical extension. There are two important concepts related with fuzzy relations, projections and cylindrical extensions. Let Q be a fuzzy relation in $U_1 \times U_2 \times ... \times U_n$ and $\{i_1, ..., i_k\}$ be a subsequence of {1,2,…,n}, then the projection of Q_p on $U_{i1} \times U_{i2} \times ... \times U_{ik}$ is a fuzzy relation defined by the membership function,

$$\mu_{Qp}(u_{i1}, ..., \mu_{ik}) = \max_{u_{j1 \in U_{j1}}, ..., u_{j(n-k)} \in U_{j(n-k)}} \mu_Q(u_1, ..., \mu_n) \quad (7)$$

where $\{\mu_{j1}, ..., \mu_{j(n-k)}\}$ is the complement of $\{u_{i1}, ..., u_{ik}\}$ with respect to $\{u_1, ..., u_n\}$.

Using the previous example, the projection on U and V are the fuzzy sets,

$$Q1 = \{\text{Lisbon/0.6} + \text{Paris/1}\}, \quad Q2 = \{\text{Paris/1} + \text{London/0.8}\}$$

Now, considering again that Q_p be a fuzzy relation in $U_{i1} \times U_{i2} \times ... \times U_{ik}$ and that $\{i_1, ..., i_k\}$ is the subsequence of {1,2,…,n}, then the cylindric extension of Q_p to $U_1 \times U_2 \times ... \times U_n$ is a fuzzy relation Q_{PE} defined by,

$$\mu_{Q_{PE}}(u_1, ..., u_n) = \mu_{Q_P}(u_{i1}, ..., u_{ik}) \quad (8)$$

Using the two projections above, Q_1 and Q_2, their cylindrical extensions to $U \times V$ for the example are, Q_{1E}={(Lisbon,Paris)/0.6+(Lisbon, London)/0.6}and Q_{2E}={(Lisbon, Paris)/1+ (Paris,London)/0.8}.

Possibility distribution. Another rather important concept for natural language queries is the definition of possibility distribution and its relation with fuzzy sets. Returning to the *small integers* example and considering a non-fuzzy proposition, W= X is an integer in the interval [0,5]. If we do not have any information about X, we can say that W induces a possibility distribution Poss(X) which associates with each integer *u* in [0,5] the possibility that *u* can be a value of X. Thus,

$$\text{Poss}\{X = u\} = 1 \qquad \text{for } 0 \leq u \leq 5$$
$$\text{Poss}\{X = u\} = 0 \qquad \text{for } u < 0 \text{ or } u > 5$$

where Poss{X=u} means the possibility that X has value u.

Using a fuzzified version of proposition W, P= "X is a *small integer*", the fuzzy set "small integer" is defined as above.

Now we are in the position to show the equivalence between possibility and fuzzy set. Using the possibility postulate [Zadeh, 1978 #735] it can be said that if X is a variable that takes values in U and F is a fuzzy subset of U, then the proposition P= "X is F" induces a possibility distribution Poss (X) which is equal to F, implying that

$$\text{Poss}\{X=u\}=\mu_F(u) \text{ for all u in U} \qquad (9)$$

In summary, the membership of the possibility distribution of X is a fuzzy set that could be used to define the possibility that X could assume any specified value u in U. Moreover, we can say that proposition "X is F" translates into the assignment of a fuzzy set F, to the possibility distribution of X. This postulate enables us to use the translation rules proposed by [Zadeh, 1978 #735]. For more details and examples about possibility distributions vs. fuzzy sets see [Bosc, 1993 #856] [Zadeh, 1978 #735]. This type of approach is denoted the fuzzy set approach by Bosc and Prade [Bosc, 1993 #856].

There are many other concepts about fuzzy set theory that are important for dealing with imprecision in information systems. A good overview about main concepts, operators and properties of fuzzy set theory can be found in [Klir, 1988 #378] and [Ross, 1995 #809].

2. 2. PRUF translation rules

As mentioned, PRUF is a meaning representation language for natural languages [Zadeh, 1978 #735]. The basic assumption of PRUF is that imprecision, which is a common feature in natural languages, has a possibilistic rather than probabilistic nature. Typically, a proposition such as 'X is tall' translates in PRUF into the possibility assignment equation, $\text{Poss}_{\text{height}}(X)$=tall (henceforth we use either Poss or $\prod$). In 1987 Zadeh extended the PRUF language with a computational interpretation called test-score semantics [Zadeh, 1987 #748]. The main idea of test-score semantics is that a proposition in natural language may be viewed as a system of elastic constraints which obtains a final test-score for the proposition by scoring and aggregating the constraints in the proposition. Here the focus is on the translation rules of PRUF which provide the basis for querying and inferencing with fuzzy premises.

The main constituents of PRUF are a collection of translation rules and a collection of inference rules [Zadeh, 1978 #735]. The collection of PRUF translation rules translate expressions in natural language (fuzzy propositions) into an expression in PRUF (with an assigned support) and the rules can be applied singly or in combination. These translation rules are:

a) Type I - modification rules. Example: 'X is very small';

b) Type II - composition rules. Example: 'X is small and Y is large';

c) Type III - quantification rules. Example: 'most Portuguese are short';

d) Type IV - qualification rules. Example: 'X is small is very possible'.

Translation rule of Type I is the modifier rule that, given the proposition P: "X is F" is transformed into a possibility assignment equation of the form, $\text{Poss}_{(x1,x2,\ldots,Xn)}$=F and then the modified translation rule translates this into the proposition p•: X is **m**F, where **m** is a modifier such as *not*, *very*, *quite* is given by,

$$X \text{ is } mF \rightarrow Poss_{(X1,X2,...Xn)} = F^{\bullet} \text{ where } \mathrm{F}^{\bullet} \text{ is a modification of fuzzy set } F \text{ induced by } m. \quad (10)$$

For example, if **m** is 'very' Zadeh suggests the use of the squaring function (from [Zadeh, 1972 #727]), mF=F^2 , thus for F={a/1 + b/0.5 + c/0.1}, the modified proposition mF is F^2= {a/1 + b/0.25 + c/0.01}.

Translation rules of Type II, pertain to the translation of propositions of the form p=q*r where * denotes any operation of composition such as conjunction, disjunction, implication, etc. Assuming that q: X is F-> $Poss_{(x1,x2,...,Xn)}$=F, r: Y is G-> $Poss_{(Y1,Y2,...,Yn)}$=G and F and G are fuzzy sets over U and V, then the compositional rules could take the following forms:

$$\begin{array}{ll}
(a)\ q \text{ and } r \rightarrow \prod{}_{(X_1,...X_n,Y_1,...Y_n)} = F \times G & \\
(b)\ q \text{ or } r \rightarrow \prod{}_{(X_1,...X_n,Y_1,...Y_n)} = \hat{F} + \hat{G} & \\
(c) \text{ If } q \text{ then } r \rightarrow \prod{}_{(X_1,...X_n,Y_1,...Y_n)} = \overline{\hat{F}} \oplus \overline{G} = F \times G + \overline{F} \times V & \\
(d) \text{ If } X \text{is F then } Y \text{ is } G \text{ else } Y \text{ is } H \rightarrow \prod{}_{(X_1,...X_n,Y_1,...Y_n)} = (\overline{\hat{F}} \oplus \hat{G}) \cap (\hat{F} \oplus \hat{H}) & (11) \\
(e) \text{ for relations, } R = X_1 \text{is } F_{11} \text{A}ND....X_n \text{ is } F_{1n} \text{ OR } X_1 \text{is } F_{21} \text{A}ND.... X_n \text{ is } F_{2n} & \\
\quad \text{OR.... } X_1 \text{is } F_{m1} \text{A}ND....X_n \text{ is } F_{mn} \text{ it follows that} & \\
R \rightarrow F_{11} \times ..... \times F_{1n} + ... + F_{m1} \times ..... \times F_{mn} &
\end{array}$$

where, Π is the possibility assignment equation ($\approx$ Poss); $\hat{F}, \hat{G}, \hat{H}$ are cylindrical extensions and $\overline{F}$ is the negation of *F*; $\mathrm{F} \times \mathrm{G}$ is the Cartesian product; + is the union; $\overline{\hat{F}} \oplus G$ is the Lukasiewicz implication (i.e. $a \rightarrow b = \min(1,(1-a+b))$) and F_{ij} are fuzzy subsets of $U_1, U_2, ..., U_j$, respectively.

Lets consider an example adapted from [Zadeh, 1978 #735]. Let F: small numbers= {1/1 + 2/0.6 + 3/0.1} and G: large numbers= {1/0.2 + 2/1}. Thus, the proposition 'If X is small or Y is large' translates into $Poss_{(x,y)}$ = {(1,1)/ 1+ (1,2)/1 + (2,1)/0.6 + (2,2)/1+ (3,1)/0.2 + (3,2)/1}. It must be noted that instead of the operations proposed for the composed translation rule our model uses different operations, as will be described in Section 3.

Translation rules of Type III, translate propositions of the form P: "QN are F", where Q is a fuzzy quantifier (e.g. *most, many*), F is a fuzzy set of U and N is a descriptor. A descriptor could be a class (tall Portuguese), a label of a fuzzy set (cheap) or a fuzzy subset (small integers). One way to express a fuzzy quantifier is by defining a function of X where X is a fuzzy attribute in the unit interval. In order to use the quantifier translation rule, the notion of cardinality of a fuzzy set must be introduced. Cardinality of a non-fuzzy set is the number or proportion of elements of U, which are in F [Zadeh, 1978 #735]. In order to extend the cardinality concept to fuzzy sets, a sigma-count must be formed [Zadeh, 1978 #735], which is the arithmetic sum of the grades of membership in F. For example for

A= {1/a + 0.5/b} the cardinality is 1.5. The terminology used is usually count(F) or sigma-count for the power-set definition or fuzzy cardinality and rcard(F/G) or relative sigma-count for the weighted proportion. Formally, the relative sigma-count or relative fuzzy cardinality is:

$$\text{rcard}(F/G) = \frac{Count(F \cap G)}{Count(G)} \text{ where } Count(X) = \sum_i \mu_X(u_i) \qquad (12)$$

which specifically corresponds to,

$$\text{rcard}\{F/G\} = \frac{\sum_i \left(\mu_F(u_i) \cap \mu_G(u_i)\right)}{\sum_j \mu_G(u_j)} \quad \text{and} \quad \text{rcard}(F) = \frac{1}{N}\sum_{i=1}^{N} \mu_F(u_i) \qquad (13)$$

where rcard(F) corresponds to G=U, N is a finite number of elements (e.g. the class of Portuguese) and F is a fuzzy set (e.g. tall).

An example of the relative sigma count might be to determine the proportion of men who are tall (F) and fat (G). If only one feature (e.g. tall) is used the proportion is obtained with the elements that have the property over the number of elements in the fuzzy set.

The relative sigma-count provides the basis for the following translation quantifier rule:

$$\text{'QN are F'} \rightarrow \mu_Q(\sum_i p(u)*\mu_F(u)) \qquad (14)$$

where p(u) is the proportion of elements of U which are in F and therefore whose values lie in the range of values of the fuzzy set F. For example, in the proposition 'most Portuguese are short', the relative sigma-count is determined by the sum of Portuguese that are short (sum of the membership degrees) over the Portuguese population (N), and then, obtain the classification of this relative sigma-count in the quantifier 'most' (represented by a fuzzy set).

Translation rules of Type IV, qualification rules, are concerned with the translation of propositions of the form p = 'N is F is τ ', where τ might be a truth-value, a probability value or a possibility value. Hence, the principal modes of qualification of a proposition are:

a) truth qualification, e.g. 'John is tall is true';

b) probability qualification, e.g. 'It is probable that John lost weight';

c) possibility qualification, e.g. 'It is possible that John is fat'.

Here we only describe the truth qualification because it is the only one used in our fuzzy query model. A truth qualified proposition is shown to be semantically equivalent to a reference proposition [Zadeh, 1978 #735], such as 'N is F is τ ' ⇔ 'N is G' where:

$$\tau = \mu_F(G) \text{ and then } \mu_G(u) = \mu_\tau(\mu_F(u)) \qquad (15)$$

Specifically, the translation rule for truth qualification is,

$$N \text{ is } F \text{ is } \tau \rightarrow \prod_X = F^\circ \text{ where } \mu_{\text{F}^\circ} = \mu_\tau(\mu_F(u)) \qquad (16)$$

Illustrating, consider the proposition ‘X is tall is very true’ with the fuzzy set definition μ_{tall} = {0.1/1.50 + 0.7/1.70 + 1/1.80}. Thus, using the square function to express ‘very’, as mentioned above: μ_{tall} = {0.01/1.50 + 0.49/1.70 + 1/1.80}. This truth qualification rule with, or without qualifications such as ‘very true’, ‘not true’ etc., is used in our fuzzy natural language processor.

A combination of the translation rules of PRUF is possible and usually needed. For example, the proposition ‘most Portuguese are tall and fat is not very true’ includes translation rules of Type III (‘most Portuguese’), Type II (composed ‘tall and fat’) and Type IV (truth qualification ‘not very true’).

In summary, as stated by Zadeh: “PRUF may be regarded as a relation-manipulating language which serves the purposes of (a) precision of expressions in a natural language; (b) exhibiting their logical structure; and (c) providing a system for the characterisation of the meaning of a proposition by a procedure which acts on a collection of fuzzy relations in a database and returns a possibility distribution” [Zadeh, 1978 #735].

3. Fuzzy Querying model

Our querying model includes four main modules: database, fuzzy components, the fuzzy natural language processor, and the dialog component. Figure 1 depicts the general architecture of the model and inter-relations of its main components.

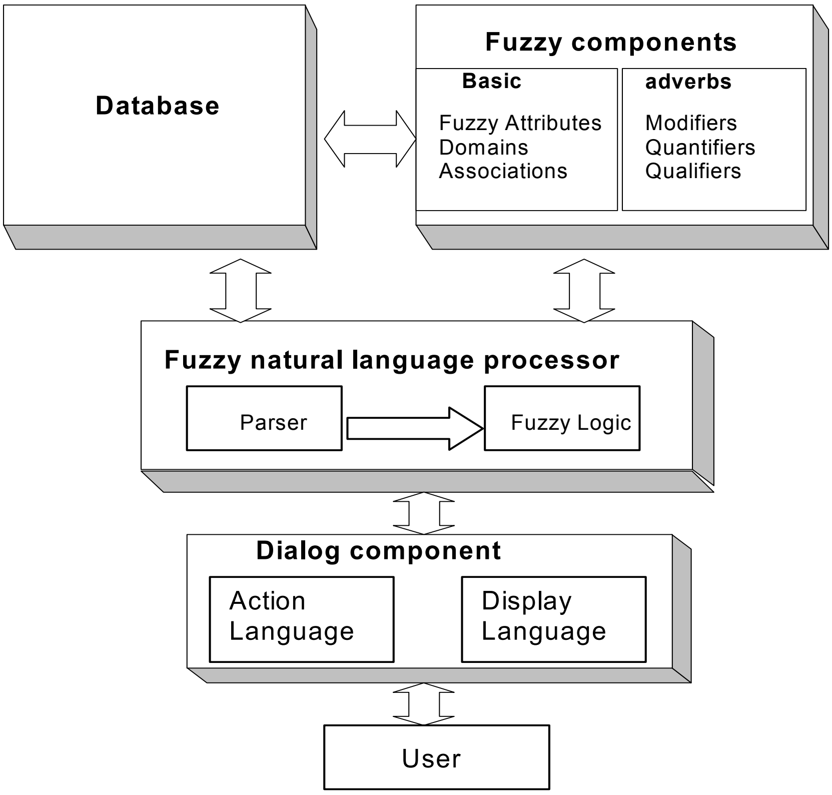


Figure 1. Querying Model

The model was implemented using two different environments: VBA (Visual Basic for Applications) and Prolog. The fuzzy engine was implemented in VBA using the DBMS Access and the parser was implemented using LPA Win-Prolog. The communication between the two implementation' environments was accomplished by using DDEs (Dynamic Data Exchange Extension).

3. 1. Economic indicators

Our prototype was built to handle the information about the 500 biggest non-financial Portuguese companies, published in [EXAME, 1996 #843]. This information includes 15 economic and financial indicators about these companies (see Table 1).

1. Sales growth, given by the ratio sales_95/sales_94;
2. Net profits growth, given by the ratio net_profits_95/net_profits_94. It measures the dynamism and the capacity to maintain or increase the market quota;
3. Assets_turnover, given by the ratio net_sales/assets. It represents the degree of efficiency of available resources;
4. Productivity, given by the ratio gross_added_value/number_workers. It measures the degree of efficiency of human resources;
5. Return_on_investment, giving the profit per unit of capital invested in the company;
6. Return_on_equity, given by the ratio of net_profits/owners_equity. It measures the profitability of the owner's capital;
7. Profit_margins, given by the ratio net_profits/ sales;
8. Sales_profitability, given by the ratio current_net_profits/sales;
9. Gross_added_value, given by the sum of the net sales, production fluctuations, subsidies and net extraordinary profits;
10. Gross_added_value/net_sales, measuring how much a company contributes to the national economy per escudo (Portuguese currency) of sales;
11. Indebtedness, given by the ratio liabilities/net_assets. It measures the capacity of the firm to contract loans (the bigger the worse);
12. Solvency, given by the ratio owners_equity/liabilities. It measures the long-term capacity to fulfil commitments;
13. Financial_autonomy, given by the ratio owners_equity/net_assets. It measures the participation of the owner's equity in financing the company activities (complement of Indebtedness);
14. General_liquidity, given by the ratio assets/current_liabilities. It measures the capacity to fulfil the short-term commitments;
15. Cash_flow, measuring the auto-financing capacity of the company.

Table 1. Information about companies and its economic indicators

These economic and financial indicators will be fuzzified and, henceforth, will be denoted fuzzy attributes.

3. 2. Fuzzy components

The fuzzy module includes two components, the basic ones and the adverbs (see Figure 1). In order to define these two components a pre-processing over the information on Table 1 was performed. This pre-processing allowed us to define the fuzzy attributes, associations and adverbs, which are then used in the fuzzy natural language processor.

We use triangular and trapezoidal functions to represent the fuzzy attributes and fuzzy adverbs (i.e. modifiers, quantifiers and qualifiers). These functions' intervals hold values (a1, b1, b2, a2) for the trapezoidal and (a1, b1, a2) for the triangular. Both are linearly increasing and usually have open-end intervals. Details about the construction and meaning of fuzzy attributes and fuzzy adverbs will be presented next.

3.2.1. Basic components

The basic components of the fuzzy model are the domains of the fuzzy attributes, the fuzzy attributes and relations (associations) between attributes.

Domain of attributes. The domain is retrieved from the initial values given in [EXAME, 1996 #843]. The domain limits for each attribute are two points, one with the minimum value and another with the maximum value. These will be used to defined the fuzzy attribute.

Fuzzy attributes. We denote fuzzy attributes the fuzzification of the fifteen economic and financial attributes (indicators) described in Table 1. To construct the fuzzy attributes we plotted all the values for each attribute (ordered) and by visual observation of the graphics, defined their membership functions. Further, we used open interval trapezoidal membership functions to define the fuzzy attributes, because, when we plotted the values for each attribute (increasingly or decreasingly ordered) it allowed a good approximation to represent the concepts involved. This is an empirical method but it takes into account the real values of each concept (i.e. attribute). The membership function is defined by using three points (a1, b1, a2) corresponding to the lower, inflexion point and higher value of the attributes (economic and financial indicators).

The steps followed to define each fuzzy set were:

- Sort the companies (private or public) by increasing (or decreasing) order for attribute values.
- Plot the values into a graphic.
- From the graphic determine points (a1, b1, a2) to build the trapezoidal membership function with an open interval.
- Build the trapezoidal function (if it contains an open interval either a1=b1 or b2=a2).

The increasing or decreasing ordering depends on the attribute type. For instance, net sales will be ordered increasingly, while extraordinary costs will be ordered decreasingly. The rational is that for some economic and financial indicators the higher the values the better the membership should be, while for others the bigger the values the worse the membership value should be. Figure 2 shows an example of the plotted values for the ratio gross_added_value/net_sales of the private companies, where the X-axis contains the ordering number of private companies and the Y-axis contains the actual indicator (attribute) values increasingly ordered. Further, X-axis refers to the 448 private companies (ordered increasingly by attribute real values) and the Y-axis refers to the values of the ratio gross-added-value/net sales.

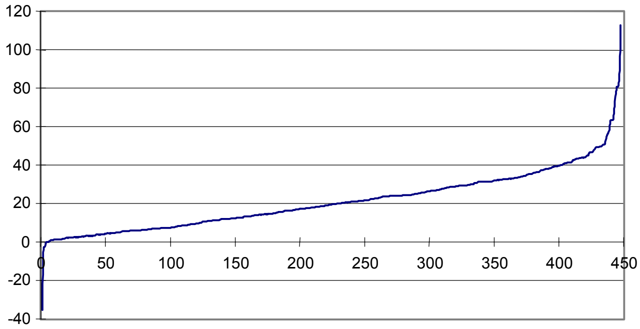


Figure 2. Gross added value / net sales

It should be stressed that public companies (a total of 52) were handled separately in terms of membership definitions because the values scale is completely different from the private ones (e.g. Public electric company serves everybody in Portugal).

Observing Figure 2, we see that the Y-axis minimum value is –35 and that from –35 to 62 the function increases in an approximate linear way. From 62 onwards the values are more or less constant. It should be noted that the vales of the Y-axis are the ones used to define the fuzzy set membership function. From the visual observation we can get the inflexion point and domain, i.e. the points (-35, 0, 62,120). With these points we can build a trapezoidal membership function (with an open right interval) considering that below 0 the membership value is always 0, from 0 to 62 there is a linear increasing line and after 62 the membership value is always 1. It should be noted that we used for lower value zero, instead of –35, because from a business point of view any negative value for the ratio is always bad. Figure 3 depicts the trapezoidal membership function that was defined for the ratio gross-added-value/net-sales.

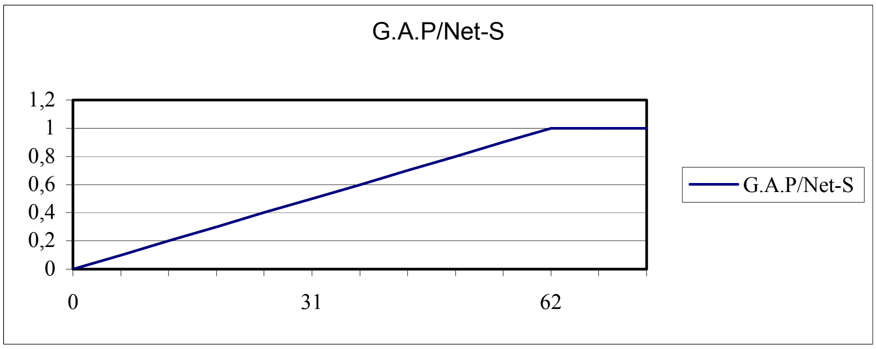


Figure 3. Fuzzy set for gross-added-value/net-sales

Observing figure 3 we see that the specific trapezoidal function is defined by 2 points [0, 62], that correspond to membership value zero for values below zero, membership value of 1 for values above the point 62, and membership values in the interval [0, 1] for elements between 0 and 62. All other fuzzified attributes were constructed in a similar fashion.

This approach allows us to build all fuzzified attributes automatically and without changing the initial database. Obviously, we could have considered other types of membership functions (such as sigmoid, Gaussian, triangular), but in this business context, trapezoidal functions proved a simple and good representation since they are similar to the original data histograms.

Associations. We considered four different types of associations in our model: dynamism, profitability, economic-contribution and financial-health. Here we assume that associations represent relations between attributes and they were created to reflect the different perspectives of managers, stockholders, government and banks. Managers are mainly interested in dynamism and productivity. Stockholders are mainly interested in the profitability of companies. The government is interested in the economic contribution of companies to the national economy. Finally, banks are interested in the financial health of companies. The four associations include the following attributes (see Table 1):

- Dynamism: it includes sales_growth, net profits growth, assets_turnover, productivity;
- Profitability: it includes return_on_investment, return_on_equity, profit_margins, sales_profitability;
- Economic-contribution: it includes gross_added_value, gross_added_value/net_sales;
- Financial-health: it includes indebtedness, solvency, cash_flow, financial_autonomy, general_liquidity.

In our model the membership value for any association is obtained by:

$$\mu_{associationX}(Company_i) = \left(\frac{1}{n} \sum_{j=1}^{n} attribute_{ij} \right) \tag{17}$$

which is a simple arithmetic average of all associated attributes pertaining to an association. Other operators could have been used to calculate the association membership value (for an overview on operators, see [Klir, 1988 #378]). It should be pointed out that an association is interpreted as an aggregation of composing elements and this is the reason for using arithmetic operators.

3.2.2. Fuzzy adverbs

The fuzzy adverbs contained in our model can be classified into: modifiers, quantifiers, and qualifiers. Instead of constructing them as either a function concentrator or a function dilation as in the PRUF approach [Zadeh, 1978 #735], we used the notion of “filter” from the evidential logic rule of Baldwin et al. [Baldwin, 1995 #67] and the interpretation for quantifiers from Kacprzyk et al. [Kacprzyk, 1986 #796], Zadeh [Zadeh, 1983 #740] and Yager [Yager, 1994 #849]. The fuzzy set membership value is “passed” through an S-funtion to determine the interpretation of the body, as

S(x): [0,1]→[0,1]. This type of "filtering" process allows more flexibility in defining modifiers and quantifiers since we can cover the spectrum of the [0, 1] interval of the attribute membership values and it is easier to interpret. For example the query "*very* profitable?" considers that, for membership values below 0.7 of attribute "profitable", the membership value of intensifying it with modifier *very* is zero (see Figure 4). Below we depict the triangular and trapezoidal functions used for modifiers and quantifiers, which were borrowed from [Ribeiro, 1993 #546], with some slight adaptations.

Modifiers. These are adverbs that modify the fuzzy attribute in a way to intensify its meaning. As mentioned above we used membership functions to represent intensification of attribute membership values (X-axis), instead of the proposal in PRUF. An example can be "is IBM *very* dynamic?". The modifiers available in our model are depicted in Figure 4.

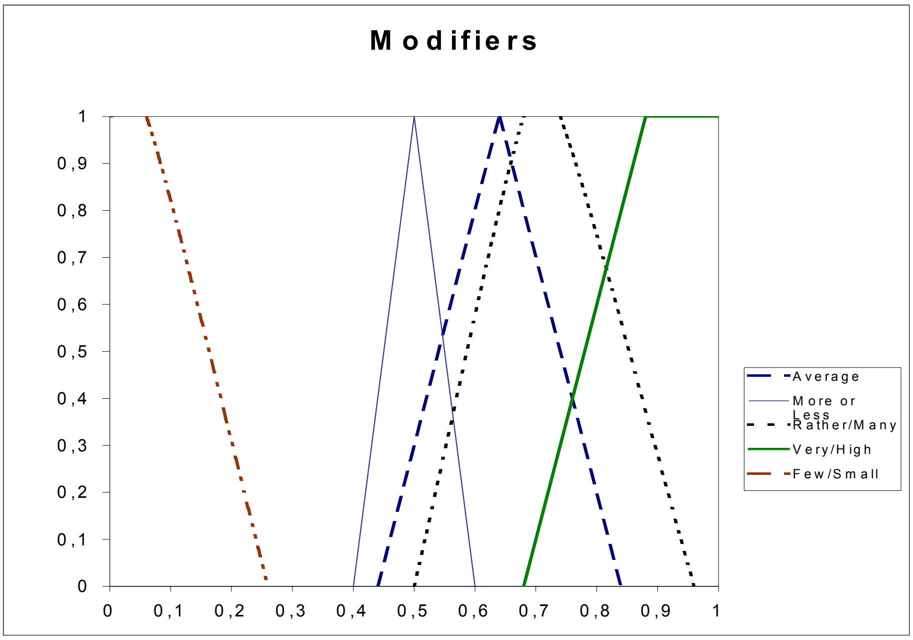


Figure 4. Membership functions of modifiers

The modifiers considered in our model are 8 {*few, small, average, more or less, rather ,many, very, high*}. Considering that some of them have similar interpretations (e.g. *very* and *high*) we use the same membership function for some of them. Although not listed in figure 4, our model also accepts the modifier *not*, which is represented by (1-μ_M(F)).

Quantifiers. These are linguistic expressions that limit the number of cases to be queried (in the sense of not having precise limits of acceptance or rejections for conditions [Bosc, 1993 #856]) such as *all, most, approximately-half.* An example is "Are most private companies dynamic?". Like the modifiers, quantifiers were defined as triangular and trapezoidal functions, but in this case we follow the interpretation of quantifiers defined by [Zadeh, 1983 #740]. The membership functions are depicted in Figure 5.

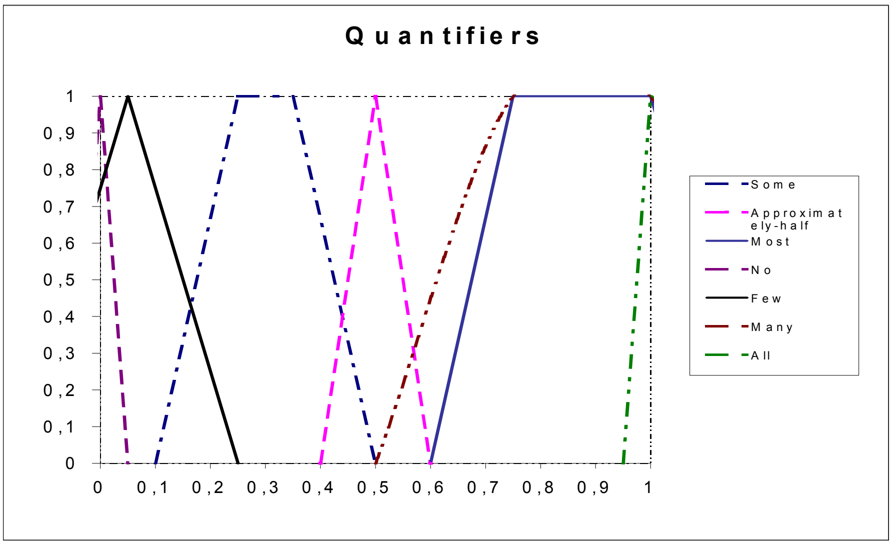


Figure 5. Membership functions of quantifiers

The quantifiers considered are seven {*no, few, some, approximately-half, many, most, all*} and we believe they are quite enough to cover most of the queries about the business indicators.

Qualifiers. These are adverbs that linguistically qualify a proposition to determine its degree of truth, probability or possibility. For example, the query "is it *very true* that IBM is *productive*?" clearly shows that some measure of qualification of the proposition is being asked. In our model we only use the truth qualification because it does not make sense to ask the probability or possibility of any company regarding a financial or economic indicator, as for example the query "Is it probable that IBM is dynamic" is incoherent in this business context. In this work we used the simple real line for the truth function, as shown in Figure 6. However, any other function could have been used to give more flexible degrees of truth.

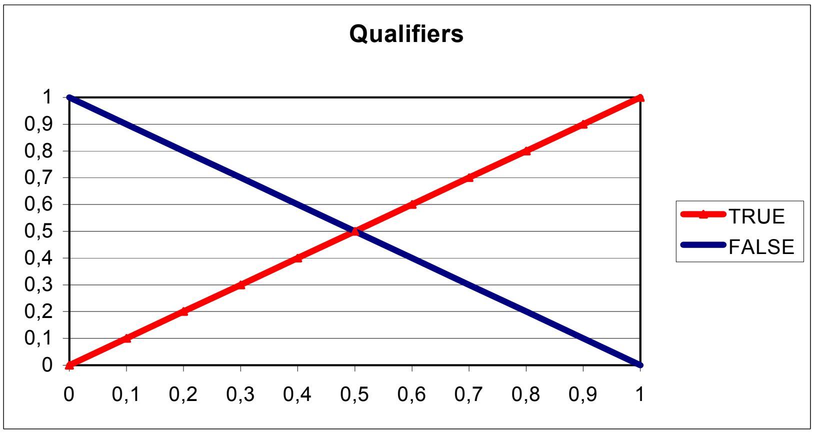


Figure 6. Membership Functions of Qualifiers

It should be noted that we use the qualifiers membership functions combined with modifiers and quantifiers to describe concepts such as "very true", instead of the linguistic hedges proposed by [Zadeh, 1972 #727].

In summary, modifiers act as intensifiers filters for attributes. For example (see Figure 4), considering a membership value of attribute F to be 0.8, if we say "*very* F", the final membership value will be 0.5 because value 0.8 has membership value of 0.5 on the modifier fuzzy set *very*. This shows it is more difficult to be, for instance, "*very* productive" than to be just productive. Quantifiers behave as a filtering process to the percentage of population on the universe that satisfies one or more attributes. For example, the query "are many companies profitable?" will trigger a counting of the percentage of companies that are profitable in the database and that value is filtered through the fuzzy set *many* in the same way as for a modifier. Qualifiers apply the filtering process to the whole proposition in the same fashion.

3. 3. Modelling the Data Base and the Fuzzy Components

By analysing the information contained in Table 1 (Section 3.1.), the following conceptual entities can be identified: Company, Public Company, Private Company, Economic Indicator, Country Controller and Region (see Figure 7). Public Company and Private Company can be defined as subtypes of the supertype Company; they were created due to the financial and economical scale differences between public and private companies, as mentioned in Section 3.2.1. Therefore, the two subtypes inherit Company attributes. Each entity is described by an identifier attribute (i.e. a key attribute) and by at least one descriptor attribute that define other characteristics the entity' occurrences.

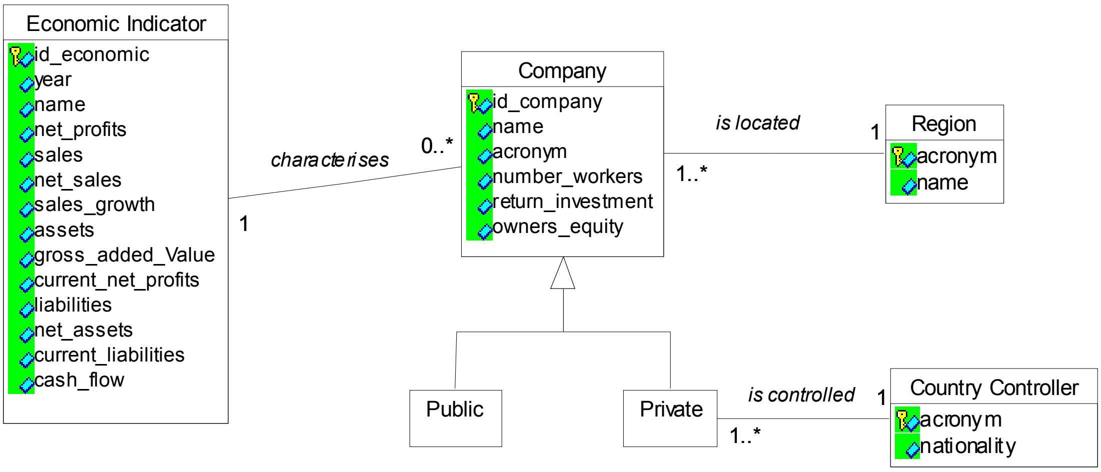


Figure 7. Entity relationship model for the company's related information

The entity "Region" was created to simplify and allow queries on companies within a certain region. "Country Controller" is only associated to private companies because public companies are government controlled.

The fuzzy components, discussed in the previous subsection, needed for the fuzzy natural language processor, can be modelled, as depicted in Figure 8.

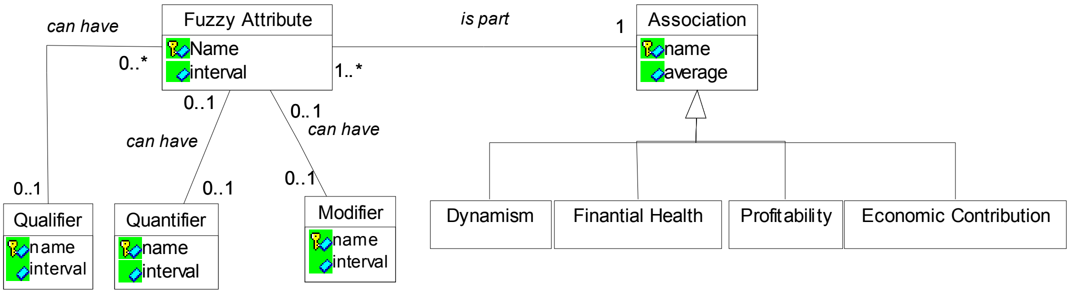


Figure 8. Fuzzy components model

As all the entities are in the Third Normal Form [Codd, 1970 #842], we can apply the cardinality rules for relational databases and obtain the final skeleton table. To accomplish this, we added a foreign key to each table in the many side of the association (i.e. cardinality '*'). The foreign keys will allow us to navigate in the database. For simplicity, Figure 9 only shows some of the tables that compose the final relational schema.

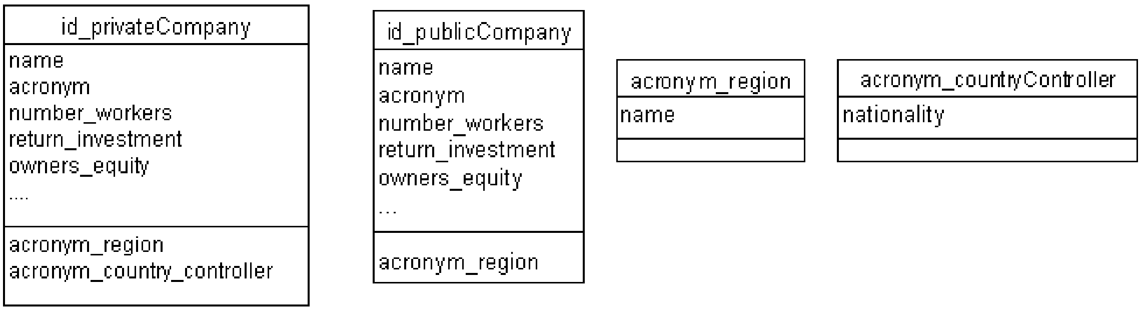


Figure 9. Tables that compose the database

Each table is divided into three compartments: the first displays the primary key, the second displays the description attributes and the third displays the foreign keys, i.e. the attributes that allow the navigation in the database. From a logical perspective, Figure 7 includes a super type and two subtypes. From an implementation perspective, and in order to increase the processing performance we decided to create only two tables for those three entities, eliminating the super type and therefore duplicating its attributes among Private and Public companies.

3. 4. Fuzzy natural language processor

This module is composed of two submodules: a parser and the PRUF fuzzy logic calculus.

3. 4.1. Parser

A natural language parser was built to translate a query into the syntactic structure accepted in our model and then to validate its semantic consistency. A first validation is done by checking the database for the existence of company names, modifiers, qualifiers, quantifiers and attributes. At the semantic level the parser validates the proposition grammar to detect invalid propositions, as for example, using a single subject with a plural verb or an incoherent question like "if X is *dynamic* then Y is *profitable*?" (The incoherence is due to the fact that either subject is the same and the attributes are different, or the subjects are different and the attribute is the same, in order to being comparable). Another semantic evaluation on type II rules is to check if the connector is *and* because then it requires that the subject to be the same.

Thus, the parser interprets the structure of the sentence in order to recognize the type of the question to be handled. For each type of question we have designed a grammar. A grammar defines the set of rules that can be used to build up a sentence. The formalisms used to define the grammars are simple:

- the non-terminal symbol appears in the left hand side of the rewrite-rule;
- this symbol is expanded in the right hand side of the rewrite-rule;
- the expansion of a symbol can be composed by terminal and non-terminal symbols;
- a non-terminal symbol always starts with uppercase;
- a terminal symbol always starts with lowercase (these are the words that appear in the original sentence);
- if the terminal symbol has to start with uppercase or if it is a set of words, then it appears between primes, such as ‘A’ or ‘economic contribution’;
- ‘|’ is a choice separator;
- ‘(‘ and ‘)’ are used to represent optional symbols.

As an example, the grammar used to define the structure of question of type I (modification) is:

Sentence → NounPhrase VerbPhrase

NounPhrase → Article Noun

VerbPhrase → (Neg) Verb Fuzzy

Fuzzy → (Modifier) Association

Article → the | ‘The’ | a | ‘A’ | an | ‘An’

Noun → subject

Verb → is | has

Neg → ‘not’

Modifier → high | low | few | ‘a lot’ | more | less | rather | many | very | small | average

Association → ‘economic contribution’ | ‘financial equilibrium’ | dynamism | profitability

The parser was implemented in Prolog. This language incorporates a mechanism called DCG (define-clause grammar) that allows us to easily transform our grammars into Prolog predicates. Moreover, we can easily: build and deal with complex and recursive data structures; represent knowledge using first order logic; implement depth search algorithms.

3.4.2. Fuzzy logic calculus

Our model includes the four types of translation rules of PRUF [Zadeh, 1978 #735]. The queries are posed in a pseudo natural language with a pre-defined structure and then they are treated according with the definitions of the PRUF translation rules, as presented in Section 2.

The calculus used in our dialog system include the "and", "or" and implication rules like "if … then" of fuzzy logic to express statements like "A and/or B are dynamic". In this work we selected the operators min and max (from the t-norms and t-conorms) for, respectively, the "and" and "or" and the implication rule of [Kleene, 1938 #377]. Many other operators, that have been proposed in the literature, could also have been used, but it is out of scope to discuss them here (see [Klir, 1988 #378], [Yager, 1991 #716]). Hence, the four translation rules are: modification, composition, quantification, qualification.

Type I: Modification ("X is m F")

This rule expresses that a simple fuzzy proposition P, P = X is F, where X is the subject and F is the fuzzy set corresponding to an attribute, as shown in the equation, $P = \mu_F(x)$, and the modified rule translation equation is,

$P^+ = \mu_m(\mu_F(x))$ where μ_m yields the membership value of $\mu_F(x)$ in a modifier function *m*. (18)

The syntactic structure of this rule for our model is:

<company_name> is <modifier><attribute>

This type of rule is the simplest one and an illustrative example can be "Is IBM *very productive*?", where F is the attribute *productive* and *very* is the modifier.

It is important to point out that when a query is made about an association (set of fuzzy attributes), the modifier is applied to the aggregated combination of attributes, given by the average of the membership values of the attributes belonging to the association.

Type II: Composition

This rule comprises three different types of compositions:

Rule II-1. It corresponds to two modification propositions "X is m_1F_1" and "Y is m_2F_2", connected by operator *and* or *or*. As mentioned, the operators used are *min* and *max* for *and* and *or* respectively. Thus,

X is m_1F_1 and Y is $m_2F_2 \Rightarrow P^+ = \min(\mu_{m1}(\mu_{F1}(x)), \mu_{m2}(\mu_{F2}(y)))$ (19)

X is m_1F_1 or Y is $m_2F_2 \Rightarrow P^+ = \max(\mu_{m1}(\mu_{F1}(x)), \mu_{m2}(\mu_{F2}(y)))$ (20)

This rule accepts more than two connected propositions, X and Y can be same subject, and m_1 and m_2 are optional. An example is "Is IBM profitable and Bayer dynamic?".

Rule II-2. It includes propositions of the type, "X is m F then Y", but the connector m can be *more* or *less*. The two cases for the connectors are:

X is more F then Y => if $\mu_{F1}(x) > \mu_{F2}(y) = 1$, else =0 (21)

X is less F then Y => if $\mu_{F1}(x) < \mu_{F2}(y) = 1$, else =0 (22)

An example could be, "Is IBM less profitable than Bayer?".

In this rule other connectors such as *much more* or *more* could also have been considered, but here we used the simplest cases in our experiment.

Rule II- 3. It is a condition rule of type If "X is m_1F_1" then "Y is m_2F_2". This connection is obtained using Kleene's implication ([Ribeiro, 1993 #546]), a => b ⇔ (~a *or* b), where the operator *or* corresponds to the *max* operator. Formally,

If X is m_1F_1 then Y is m_2F_2 => P^+= max [*not* $(\mu_{m1}(\mu_{F1}(x)))$, $\mu_{m2}(\mu_{F2}(y))$] (23)

where *not* is given by (1-μ). Like above, modifiers are optional. All types of composition rules accept negation of mF.

The respective accepted syntactic structures for Type II rules:

<company_name> is <modifier><attribute>{and/or}<company_name_2> is <modifier> <attribute>

or

<company_name> is {more/less} <attribute> then <company_name_2>

or

if <company_name X> is <modifier><attribute> then <company_name Y> is <modifier> <attribute>

Type III: Quantification

This rule corresponds to the proposition, P = qX are F, where X is a set of objects, F is a fuzzy attribute and q is a quantifier. Thus, the translation rule equation is:

QX are mF =>P^+= $\mu_q(\mathbf{rcard}_F(x))$ (24)

where $\mathbf{rcard}_F(x)$ is the averaged proportion of the membership degrees of the elements x in F (relative sigma-count in Zadeh´s terminology) and μ_q is the membership value of **rcard** in the fuzzy quantifier function Q.

The syntactic structure of rule III, accepted by our model is:

<quantifier> <companies/region/control> are <modifier><attribute>

We should point out that other interesting approaches to linguistically quantified propositions have been proposed in the literature, such as [Kacprzyk, 1984 #795] [Yager, 1983 #705]. However, since in this paper we follow Zadeh's proposal they will not be discussed further.

Type IV: Qualification

This rule measures the truthfulness or falsity of a modified proposition. Thus the translation rule is:

$$X \text{ is } mF \text{ is true} \Rightarrow P^{+}=\mu_{truth}(\mu_{m}(\mu_{F}(x))) \qquad (25)$$

where μ_{truth} is the membership value in the qualifier truth (or qualifier false) of the modified rule. An example could be "Is it true that IBM is dynamic?".

The syntactic structure of type IV rule, accepted in our model is:

<company_name> is <modifier><attribute> is {true/probable}

As could be observed in the rules' description, the attributes can have modifiers, qualifiers or quantifiers attached to them. Modifiers change the strength of an attribute, for example, the query "IBM is *very* productive" is more difficult to achieve a high membership value than the query "IBM is productive". Quantifiers change the strength of the proposition and qualifiers define the truth or falsehood of a proposition.

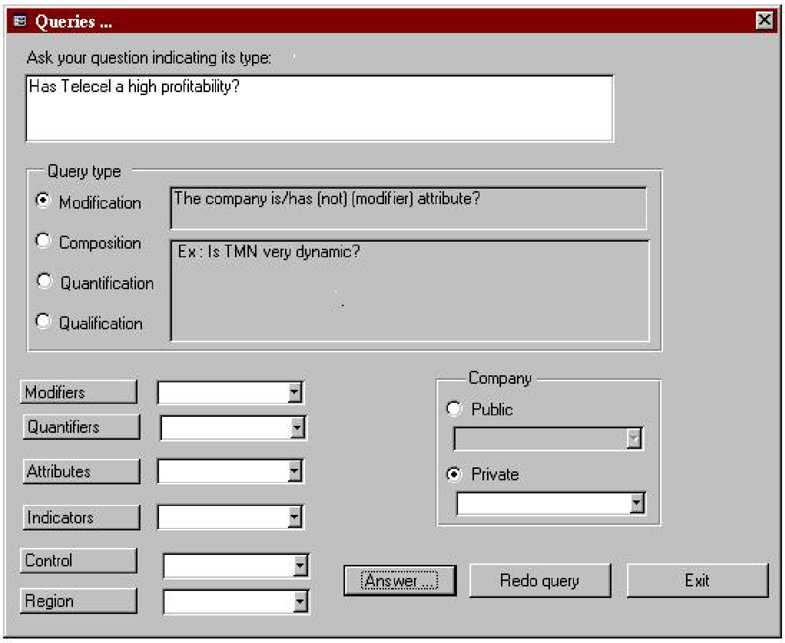
3. 5. Dialog component

The dialog component performs the dialog with the user and is composed of two items: the action language and the display language [Turban, 1993 #661]. The action language controls the communication between the user and the system in terms of the input, i.e. how the user states his/her queries. The display language is how the system communicates the answers to the user, i.e. the output answers.

The user dialog is a very important component of our model because its flexibility and ease-of-use is what makes it a real human-oriented interface.

3.5.1. Action language

The action language dialog is built to minimize the possibility of introducing errors since the user can select with a click modifiers, qualifiers, attributes, indicators, select a region or a company from private to public ones. Further, examples of each of the four types of query are available to help non-expert users to formulate their queries. The action language dialog is depicted in Figure 10.


Queries ...

Ask your question indicating its type:

Has Telecel a high profitability?

Query type

● Modification
○ Composition
○ Quantification
○ Qualification

The company is/has (not) (modifier) attribute?

Ex : Is TMN very dynamic?

Company: ○ Public ● Private

Figure 10. Action dialog

All the pull-down menus contain a list of existing modifiers, quantifiers, attributes, indicators (referring to the associations of attributes), control (foreign owners), regions, public companies and private companies. The user does not need to know which modifier, quantifier, fuzzy attributes and so forth, exist in the system. He/she can formulate the query by just selecting from those menus. Further, there is a help box (the one that says “ex: is TMN very dynamic”) to help the user select the type of query desired (modification, composition, quantification or qualification) and to show how the query should be made.

This action language will allow the users to perform queries in a pseudo-natural language to retrieve knowledge instead of simple facts. In summary, our action language interface is clearly intended for expert and non-expert users that want an easy to use query system.

In Section 4 we present various types of queries and results obtained to show the capabilities of our model.

3.5.2. Display Language

The display or presentation language processes what the user sees as a response to his/her query. In our case the results have two distinct forms, quantitative and qualitative. First, as mentioned, it uses the translation rules calculus, presented in Section 3.3, to obtain a quantified result for the query. The quantified answer is then matched with a linguistic interval corresponding to the quantitative evaluation. The following correspondence table depicts these linguistic display values:

Answers Intervals	Modification/Quantification	Qualification
[0 – 0.02]	Null	null truth

]0.02 – 0.2]	very small	very low truth
]0.2 – 0.4]	Small	low truth
]0.4 – 0.6]	Average	more or less true
]0.6 – 0.7]	above average	quite true
]0.7-0.85]	High	rather true
]0.85 – 1]	very high	very high truth

Table 2: Linguistic display values

To determine the seven intervals displayed in the table we followed an empirical method. Many other intervals could have been determined, but these seven seem to cover rather well the nature of the results obtained from our database. It should also be noted that they are easily changed to match any other context database (using the same method).

Further, considering that our objective is to build a human-oriented interface, the answers from our model also need to reflect this user-friendliness. Hence, to provide a more human-oriented answer, the following four output templates, as part of the display language, were created:

a. modification answers => "<modifier+attribute> is linguistic value (quantitative value)"

b. composition answer => "the condition is true/false (quantitative value)"

c. quantification answer => "the set has linguistic value <modifier+attribute> (quantitative value)"

d. qualification answer =>"the question is/has linguistic value (quantitative value)"

Examples of each template are shown in Section 4.

4. Querying examples and answers

The queries presented in this section cover all types of rules accepted by our model. However, because the model was developed in Portuguese, the queries and answers correspond to a translated version. Besides the query and answer, at least one rule of each type (modification, composition, quantification and qualification) also show the respective calculus, to clarify the reasoning process involved.

4. 1. Modification queries

Query 1: Has Dan Cake high return on equity?

Answer: high return on equity is high (85%)

Calculus of answer:

Value from database: (return_on_equity)= 18.6

Membership of attribute (return_on_equity)= 0.82

Filtering with modifier high, $\mu_{high}(0.82)=0.85$

Query 2: Is DanCake rather profitable?

Answer: rather profitable is average (48%) because,

- return on investment is average (58%)
- return on equity is high (82%)
- profit margins is average (49%)
- sales profitability is average (45%)
- association of profitable, $\mu_{profitability}$(DanCake)=(0.58+0.82+0.49+0.45)/4 = 0.58
- filtering with rather, μ_{rather}(profitability)= 0.48

Query 3: Has IBM financial health?

Answer: financial health is average (43%) because,

- cash flow is above_average (61%)
- indebteness is average (42%)
- general liquidity is small (25%)
- solvency is very small (14%)
- financial autonomy is high (72%)
- Association of financial health, μ_{fin_health}(IBM)=(0.61+0.42+0.25+0.14+0.72)/5= 0.43

Query 4: Has IBM a high financial health?

Answer: high financial health is null (0%) because,

- cash flow is average (61%)
- indebteness is average (42%)
- general liquidity is small (25%)
- solvency is very small (14%)
- financial autonomy is high (72%).
- Association of financial health, μ_{fin_health}(IBM)=(0.61+0.42+0.25+0.14+0.72)/5 = 0.43
- Filtering with high, μ_{high}(0.43)= 0

4. 2. Composition queries

Query 5: Has Dan Cake more return on equity than Bayer?

Answer: the condition is false(0%)

Calculus of answer:

database value: Dan Cake(return_on_equity)= 11.7

database value: Bayer(return_on_equity)= 18.6

membership: Dan Cake (return_on_equity) = 0.8225

membership: Bayer(return_on_equity)= 0.8678

Comparative composition: If 0.8225 > 0.8678 μ=1 else μ=0

4. 3. Quantification queries

Query 6: Do most companies in central Portugal have sales_profitability?

Answer: the set has very high sales_profitability (100%)

Calculus of answers:

$\text{rcard}_{\text{sales\_profit}}$=0.90 (rel. cardinality of all companies with sales_profit.)

$\mu_{\text{most}}(0.90)=1$

Query 7: Do some companies have financial_health?

Answer: the set has very high financial health (100%) because

- cash_flow is (35%)
- indebteness is (36%)
- general_liquidity is (20%)
- solvency is (21%)
- financial autonomy is (75%)
- Association for fin. Health, $\mu_{\text{fin\_health}}=(0.35+0.36+0.20+0.21+0.75)/5 = 0.37$
- Filtering with some, $\mu_{\text{some}}(0.37)= 1$

4. 4. Qualification queries

Query 8: Is it true that IBM is productive?

Answer: the question has low truth (21%) because:

Calculus of answer:

value from database: 11.3

membership of productive: 0.21

$\mu_{\text{true}}(0.21)= 0.21$

Query 9: Is it true that TMN has financial health?

Answer: the question is more or less true (41%) because:

- cash flow has low truth (35%)
- indebtedness is more or less true (59%)
- general liquidity (no data available!)
- solvency has low truth (26%)
- financial autonomy is more or less true (42%)
- Association for financial health (35+59+26+42)/4 = 41%

This small set of questions illustrates the behaviour of our querying model, both at the query and answer levels. It clearly displays how a human-oriented query model can help a non-expert user to ask natural language questions and obtain not only raw data, but also real information about the companies.

5. Conclusions

We presented a fuzzy querying model capable of handling various types of questions in a natural language form. The query system allows questions on different market perspectives, such as from managers, bankers, stockholders and government, as well as a general overview about the main economic and financial data of the largest 500 non-financial Portuguese firms.

The main advantages of our model are: existing implemented systems do not have to be modified; the developers do not have to learn neither a new query language, such as a new extension to SQL, nor new set-operations to maintain legacy systems; the dialog with the system is done in a language very close to natural language; the answers are given in a linguistic form, as well as a numeric form, which helps the user to better understand the results obtained;

Further, our model is human-oriented and the four query-types used cover a large spectrum of possible queries to be made in the non-financial Portuguese companies' database. The interface developed is user-friendly and does not require prior knowledge about the existing modifiers, quantifiers, qualifiers, attributes and so forth, because they are listed on the screen by clicking on the respective pull-down menu. However, many improvements could be considered both in the query and answer formats. Specifically, at the level of relational operators and fuzzy algebraic operations this interface could be expanded to consider other proposals on fuzzy query languages considered in the literature.

To adapt this model to other databases we only need to define new syntactic structures for the fuzzy attributes, because these are context-dependent. However, since the proposed approach is quite general it seems that the adaptation is not difficult.

We believe this type of human oriented interfaces can be very useful for companies that wish to provide a really user-friendly service to the community. More human-oriented query models should be developed to learn from experience and, hence, improve their capabilities to allow easy and fast access to non-expert users.

Acknowledgements

We are indebt to the students Claudia S. Castelhano and Susana R. Barros for the implementation of the proposed interface.

References