# Parallelism in Fuzzy Databases

**Liberios Vokorokos, Anton Baláž, Norbert Ádám**

Department of Computers and Informatics, Technical University of Košice
Letná 9, 042 00 Košice, Slovakia
E-mail: Liberios.Vokorokos@tuke.sk, Anton.Balaz@tuke.sk,
Norbert.Adam@tuke.sk

*Abstract: This paper introduces basic parallelism in fuzzy databases and its queries. Article describes basic characteristics of fuzzy databases and their properties, compares fuzzy logic with boolean one in connection with fuzzy databases. Deal with data clasification and form of storage fuzzy data in database systems. Paper describes fuzzy queries and parallelism in queries from view of today's systems and their connection to SQL language and its extension fuzzy SQL. Discuss about pipeline parallelism of query operators and its benefits.*

*Keywords: fuzzy databases, parallel DBMS, SQL, parallel querying*

## 1 Introduction

Information quantity today more than ever increase every day more and more. It is recommended that new information could by handled very quickly with futher optional manipulation. This leads us to the need, increasing flexibility of systems for data manipulation. To give user oportunity to get huge number of data, it is recommended to have tools, which alleviate his access to data. Today are those tools database systems (DBMS). Current classic database systems are well know, but in quick spread society its futher development is necessary. From view of speed, solution is main memory databases, which are real-time databases. One form of incresing DBMS performance is build in parallelism of DBMS. Another ambition is to get query language as near to human expression as possible. Because human expression is not exactly precise, often full of imprecise and vague terms, there is demand of connection human expression with database systems. Actual query systems work with boolean logic. Only allowed values are yes or no, item belongs or does not belong to set, there is nothing between. This kind of logic does not reflect human expression. Now but exists theory, which does not divide world into black or white. This theory is named fuzzy set and its fuzzy logic.

This paper deals about explaination basic properies of fuzzy databases, database parallelism,queries parallelism, storaging fuzzy data, fetching data, positives of joining fuzzy logic and fuzzy databases.

## 2 Fuzzy Set and Classification Function

Origin of fuzzy set theory is up to year 1965, when prof. Lotfi A. Zadeh published arcticle, in which defines basic term – fuzzy set. Fuzzy set is generalization of classic set, which accept absolute, nul or partially membership. Which means, that item could be member of set with some degree, it is named as level of classification. Higher level of classification means higher representation.

Function, which gives every item classification level is named classification function. Classification level could be accepted as value, which explains degree of truth statement, that item belongs to set. Mostly, classfication level means real number on interval of values <0,1>.

Practical this denotes, that instead of two values 0 and 1 to represent membership of item, there are infinity chances from interval <0,1> to explanation of membership to set. Based on this simply thesis, exists today huge and perspective mathematic theory, which allows explanation reality in imprecise and vague form. This leads us to term fuzzy, which means imprecise, vague, opague, indefinite.

## 3 Differences between Fuzzy Logic and Boolean Logic

Fuzzy logic is extension of normal boolean logic, which is extended to identifier partially truth. What means, something between absolute truth and absolute false.

Advantage of fuzzy logic is mathematical ability catch up information described by words. This gives us possibility to work with ambiguous term like “small”, “near”, “far”, “about”, “very” and with number of other words used in human language. Basic problem is, how to handle words, which meaning is hard to define. Is close 50m, 100m or 1 kilometer. If 50m is close, so 51m is not? Is it far? Exactly number of ambiguous in common language represents problem, which can not by solved by boolean logic. To learn computer systems human speech, it is necessary to solve this problem. Fuzzy sets are probably one of solution.

Boolean logic model become basement form of classic quering languages among which belong SQL (Structured Query Language). Boolean algebra brings logic operators and (conjuction), or (dijunction), and not (negation). By using these operators when searching with imprecise or not complete information, it is not ensured that we get requested data.

By using AND, some inaccuracy resides in problem of results, because information which do not match one of conditions or more than one conditions are the same. There is no differences between information, which meet different number of conditions in query.

Disadvantage of strict evaluations AND operator express in this situacion: User is looking for information and he is sure that information exist in database. Conditions, which exactly identify searching data are composed into query using and operator (e.g year, author, name). If user enters only one wrong condition, than result will not include requested information. User has to do corrections in his query. If there are no information meeting query conditions, than user will be pleased if he gets information which meet conditions with some degree. That is hardly handled by classic approach.

In case of operator OR, there is no chance to differentiate information matching different number of conditions. Result consists of data, which meet one condition or two and more conditions. This can leads to huge number of tuples among which is wanted tuple.

Empirical research shows, that strict form of evaluation boolean operators do not match human thinking, appreciation and decision. Human tolerance, displayed by do not eliminating solutions, which do not meet all his conditions is in contrast to strict evaluation and operator.

Basic difference between fuzzy approach and boolean one is ability to get rated result, which can by ordered by real number from <0,1> interval. User has chance to deal only by the most valuable information, what is not possible to handle by boolean logic. Next differece is direct in query, where fuzzy statement can be used. It is allowed to use words like average, big, far, near.

## 4 Fuzzy Databases and Data Classification

As the application of database technology moves outside the realm of a crisp mathematical world to the realm of the real world, the need to handle imprecise information becomes important, because a database that can handle imprecise information shall store not only raw data but also related information that shall allow us to interpret the data in a much deeper context, e.g. a query "Which student is young and has sufficiently good grades?" captures the real intention of the user's query than a crisp query as

```
SELECT * FROM STUDENT
WHERE AGE < 19 AND GPA > 3.5
```

Such a technology has wide applications in areas such as medical diagnosis, employment, investment etc. because in such areas subjective and uncertain information is not only common but also very important.

One of the major concerns in the design and implementation of fuzzy databases is efficiency i.e. these systems must be fast enough to make interaction with the human users feasible. In general, there are two feasible ways to incorporate fuzziness in databases:

1 Making fuzzy queries to the classical databases

2 Adding fuzzy information to the system.

Storaged data in database can be classified as following:

**Crisp** :There is no vagueness in the information.

e.g., X = 13

Temperature = 90°

**Fuzzy** :There is vagueness in the information and this can be further divided into two types as

**Approximate Value:** The information data is not totally vague and there is some approximate value, which is known and the data, lies near that value.

**Linguistic Variable:** A linguistic variable is a variable that apart from representing a fuzzy number also represents linguistic concepts interpreted in a particular context. Each linguistic variable is defined in terms of a variable which either has a physical interpretation (speed, weight etc.) or any other numerical variable (salary, absences, gpa etc.)

A linguistic variable is fully characterized by a quintuple <v,T,X,g,m> where,

v - is the name of the linguistic variable.

T - is the set of linguistic terms that apply to this variable.

X - is the universal set of the values of X.

g - is a grammar for generating the linguistic terms.

m - is a semantic rule that assigns to each term t $\varepsilon$ T, a fuzzy set on X.

The information in this case is totally vague and we associate a fuzzy set with the information. A linguistic term is the name given to the fuzzy set.

e.g., X is SMALL

Temperature is HOT

These are considered have a trapezoidal shaped possibility distribution as shown below

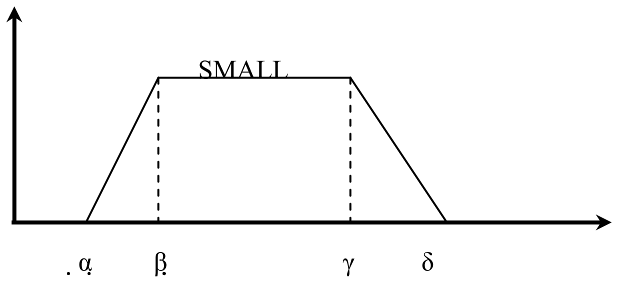

Figure 1
Possibility Distribution for a Linguistic Term SMALL for the Linguistic Variable HEIGHT

There are four parameters associated with a linguistic term as α, β, γ and δ as shown in the Fig. 3. For the range [β, γ] the membership value is 1.0, while for the range [α, β]and [γ,.δ] the membership value remains between [0.0, 1.0].

## 5 Query Language and its Parallelism

The most of quering languages are based on classic two value logic and does not allow to trasform vague conditions into query. That's why result is again classic set of data, which match strict conditions in query. This means, strictness in input leads to strictness on output. User is not able to different information on output in case of, basement request form in native language. There is no difference between more relevant or significant information.

Ability of fuzzy set theory to work with vagueness information, enable effective application in quering data. If the information is accepted as value, which human gives to data, than object of our interest are data storaged in database, ordered in user's way. Main user's deal is to find relevant information. User, who form query in native language trasform his query into query language of software he use. Today is the most interesting think, using classic database systems, which do not storage vague data directly. Based on query form, speech is about two category.

- Using fuzzy predicats, fuzzy sets.
- Adhoc extension of actual database queries.

First named option is based on using fuzzy sets and their properties. Every selection of information (selection, projection, join) form again data set. This property is heart of fetching data through fuzzy predicates, when the request is in boolean algebra imprecise. Result of fetched operations is collection, where by fuzzy set properties (join, conjuction, disjuction, union) is possible to give classification level to each item in set, and evaluates input query of fuzzy predicates.

Second option shows, how are the queries allowed in ad'hoc extensions of relational systems, expressed in terms of fuzzy sets. Basically, any such query involves two aspects: a selection part (S) and an ordering part (O) and can be seen as: "select the tuples satisfying S, then rank them according to O'. In the framework of fuzzy sets, we will have one component for S (S') valued over {0,1} (generally S itself) and one for O (O') expressing the ordering behaviour of the system as a membership degree over [0,1]. It is of course of importance that the effect of O' is exactly the same as that of the initial mechanism expressed by O. One major problem is then to find out an appropriate fuzzy set expression in each case. Authors [3] use three main classification function, complement order, distance from ideal object, criteria of weight and priority. Details of named methods can be found in [3]. It is hard to make decision, which method is better or more precise. Each of this method, regarding to actual database systems need to do additional operations to retrieve information.

In previous paragraph was described fuzzy databases according to information and its classifcation. Another interesting view is query paralellism in fuzzy databases and its queries. Relational queries are ideally suited to parallel execution; they consist of uniform operations applied to uniform streams of data. Each operator produces a new relation, so the operators can be composed into highly parallel dataflow graphs. By streaming the output of one operator into the input of another operator, the two operators can work in series giving pipelined parallelism. If one operator sends its output to another, the two operators can execute in parallel giving potential speedup of two. By partitioning the input data among multiple processors and memories, an operator can often be split into many independent operators each working on a part of the data. This partitioned data and execution gives partitioned parallelism.

The benefits of pipeline parallelism are limited because of three factors:

- Relational pipelines are rarely very long - a chain of length ten is unusual.
- Some relational operators do not emit their first output until they have consumed all their inputs. Aggregate and sort operators have this property. One cannot pipeline these operators.
- Often, the execution cost of one operator is much greater than the others. In such cases, the speedup obtained by pipelining will be very limited.

The parallel database system is based on the top-down evaluation of logic programs. Parallelism is provided at the rule level, by transforming the fuzzy query AND/OR tree into Disjunctive Normal Form. The clauses of the transformed formula are executed independently in parallel, on a transputer multi-processor machine.

## 6 Realization of Fuzzy SQL

To get information from database it is needed to have tools, query languages to do this. Today the most used query language represent SQL. SQLf is based on SQL with additional properties of imprecises, fuzziness. The objective is to introduce some fuzziness in the base block of SQL. This can be achieved at two principal levels: in the predicates and in the way they are combined (connectors). First of all we assume that a fuzzy condition fc delivers a fuzzy relation fr (where each tuple has a degree expressing its membership to the relation) and that the result of a query must be a usual relation more precisely the "best" elements of fr. So, it becomes necessary to provide the user with an output regulation mechanism that can be either the number n of desired responses or t ¢ [0,1] for the t-cut of fc. In so doing, the new formulation for a simple base block is: select <n/t> <attr> from <relations> where <fuzzy cond>. Basically, a fuzzy condition applying to individual tuples is composed of boolean and fuzzy predicates and connectors (and, or, means, etc). Just like in an ordinary query a predicate can express a join between two relations, it is possible to connect two relations by means of a fuzzy predicate, like in : select ... from R, S where ... more or less equal (R.A, S.B).

Fyzzy relations can be queried by unfocused members, which keep classification level in memory and which special cases 0 and 1 represent normal relation. Classification level is given by classification function. Integrated fuzziness (classification level) are not evaluated during inserting data, but classication level are calculated during quering data. Values are unique for each query and exists only during one request.

That proces can be divided into following steps. At the beginning there is normal relation, which quering through unfocused conditions become fuzzy semi-relation. Doing this, each tuples in result gets classfication level in connection with fuzzy conditions. Through fuzzy set operators, semi-relation become definite fuzzy relation, which is translated into normal relation by result definition.

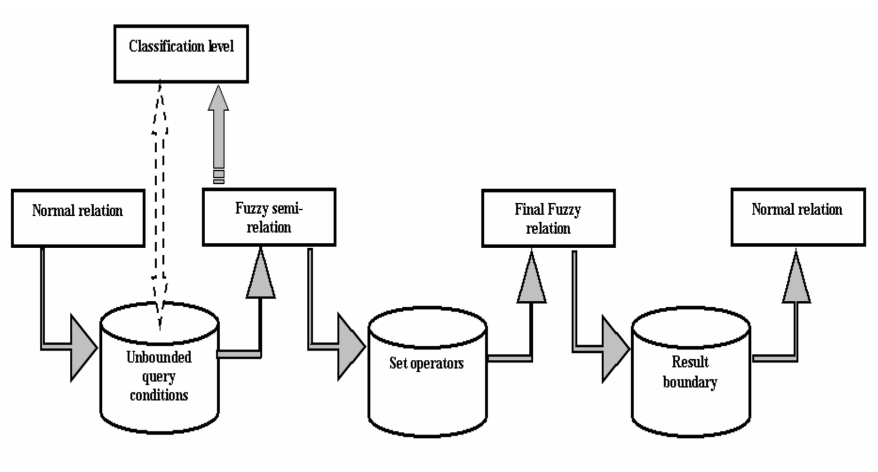

Figure 2
Fuzzy SQL executing.

Main point of method is definition unfocused conditions, which transform normal relation to fuzzy relation. By miscellaneous function, gets every tuple level of classification. Fuzzy conditions can include words like „very“ „very much“, „little“, „few or more“ or negation. Connection inside of unfocused conditions will be realized by fuzzy set operators, for conjuction (max operator) and for disjuction( min operator).

**SQL :** *select <attributes> from <relations> where <condition>*

**SQLf :** *select **n/t** <attributes> from <relations> where <**fuzzy** condition>*

Difference resides on new <fuzzy condition>, unfocused condition is used instead of simply boolean condition. Application of boundary parameter in result by n and t determined, which tuples will by in final relation.

***Boundary operators:***

- Integer **n** for quantity (e.g. find 50 books)
- Real number **t**, which determine boundary value for membership (e.g. find all books with membership 0.5)

After acceptance boundary values, result is sharp relation, because values of membership are removed. They are not needed again. That leads us to ability to request data in form:

*Select NAME, AGE, INCOME From EMPLOYEE Where*

*(AGE = middle) and (INCOME >= high)*

Final result is direct in query, where it is possible to use vague terms like (middle, high, very), which could be found in real, native language request.

## Conclusions

Today exists huge application of fuzzy sets and their properties. One of those are databases. Fuzzy sets represent basement of fuzzy database systems, which lead to futher step of joinnig computer systems and human. They bring oportunity to query data by language close to human speech. This paper focuses only on some fuzzy databases properties. We explain fuzzy sets, fuzzy logic and its comparation to boolean one, form of storage fuzzy data, parallelism of query operators and fuzzy SQL (SQLf). Nowadays we can not talk about genuine fuzzy database systems. Current fuzzy database systems are extension to actual database systems, which allow apply fuzzy terms into the systems using boolean logic.

## References

[1] Anna Kolesárová, Monika Kováčová, *Fuzzy sets and their applications* STU Bratislava 2004, ISBN 8022720364

[2] Juan Eduardo, Marlene Goncalves, Leonid Tineo, *A Fuzzy Querying System based on SQLf2 and SQLf3*

[3] P. Bosc, O. Pivert, *Fuzzy queries and relational databases*

[4] B. P. Buckles, F. E. Petry*, Fuzzy databases in the New ERA*, Tulane University, New Orleans

[5] Marek Kalman, *Fuzzy databases*, TUKE Košice 2004

[6] Bosc P. and Pivert O. SQLf: *A Relational Database Language for Fuzzy Querying,* IEEE Transactions on Fuzzy Systems, Vol. 3, No. 1, Feb 1995

[7] Annita N. Wilschut, Stephan A. van Execution*, A model of pipelined query execution*, University of Twente, Netherlands

[8] Annita N. Wilschut, Peter M. G. Apers, *Dataflow query execution in parallel main-memory environment*, University of Twente, Netherlands

Supported by VEGA project No. 1/1064/04