Modern software tools for researching and teaching fuzzy logic incorporated into database systems

Authors:

Bożena Małysiak-Mrozek, Institute of Informatics, Silesian University of Technology, Gliwice, Poland, Bozena.Malysiak@polsl.pl
Stanisław Kozielski, Institute of Informatics, Silesian University of Technology, Gliwice, Poland, Stanislaw.Kozielski@polsl.pl
Dariusz Mrozek, Institute of Informatics, Silesian University of Technology, Gliwice, Poland, Dariusz.Mrozek@polsl.pl

Abstract *— In the paper, we show novel software tools developed at the Silesian University of Technology used in researching and teaching different forms of data processing with the use of fuzzy logic. Tools that we have developed allow us to create basic types, functions and operators used in our research. Moreover, they can be used in teaching students the foundations and principles of approximate data processing. These tools are available for the broad community of users and we want to make them popular over the entire world. We also present various implementations of real software applications that have been developed for the last five years based on the courses of data processing with the use of fuzzy logic. They confirm the transfer of the knowledge given at the courses towards the industry and commercial purposes.*

Index Terms *— databases, fuzzy sets, problem based teaching, novel software tools*

INTRODUCTION

Information systems for collecting and storing large amounts of data are now an integral part of our reality. A critical problem in these types of systems is the efficient search for the desired information. Analysis of the new trends in development of databases has shown that instead of the traditional search, which returns objects that strictly fulfill the given conditions, we can frequently use methods of approximate information retrieval.

Fuzzy logic is a form of approximate reasoning derived from the fuzzy sets theory proposed by Lotfi Zadeh in 1965. It bases on the assumption that everything is true with an appropriate degree of truth [1], [2]. In this way, it exceeds the classical, binary logic, which admits only two values 0 and 1 (false and true). In fuzzy logic the degree of truth of an expression or statement can vary between 0 and 1. For this reason, fuzzy logic is increasingly used in database systems for these cases of data processing, which need imprecise criteria of data filtering, grouping and aggregation [3]-[7]. Incorporation of fuzzy logic into database systems gives two possibilities regarding processing of data: (1) we can process data in a fuzzy manner, which usually extends the final result set showing or grouping similar data; (2) we can process fuzzy data stored in appropriately designed database, and the processing can also be crisp or fuzzy.

Due to the increasing popularity of methods of the approximate information retrieval, it is advisable to introduce into the teaching program appropriate systems using such an approximate search.

This paper presents the FuzzyQ system used to education purposes, which allows the construction of queries in the extended language FuzzySQL and their correct interpretation. The main task of the FuzzyQ system is to study the extended SQL by a graphical representation of the location of fuzzy elements built in the SQL syntax. Additionally, the system supports building a skeleton of queries containing fuzzy elements, sending them to a database server, and presenting an output result set.

We assume that user of the FuzzyQ system knows the syntax of SQL commands [8], has a minimum knowledge of fuzzy logic [1], [2], and how to create fuzzy queries (user should know implemented additional features and fuzzy operators [9]-[11]). The FuzzyQ system allows to connect to any database, for which fuzzy functions and fuzzy operators are defined.

The system is currently used in teaching fuzzy information retrieval in databases in the Institute of Informatics, at the Silesian University of Technology. During laboratories students make use of databases extended by appropriate fuzzy types, functions and operators.

FUZZYSQL QUERY LANGUAGE

FuzzySQL [11] is an extension of the SQL language, developed in 2003 in the Institute of Informatics at the Silesian University of Technology, which allows approximate retrieval of objects in database systems. Approximate nature of the search is achieved by incorporating elements of fuzzy set theory to the classical SQL syntax, and therefore enabling the construction of queries containing fuzzy expressions.

Fuzzy Terms in WHERE Clause of SELECT Statement

In this section we present the structure of queries containing fuzzy expressions. In fuzzy queries we apply elements of the fuzzy sets theory. The characteristic function determining the degree, in which rows from the database meets the search criteria specified in a query, is the membership function, which takes values from the interval [0, 1].

Let's consider a simple query in the SQL language:

```
SELECT A1, ..., Ak
FROM T
WHERE W;
```

where: W represents a filtering condition: X is A (X – column of the table T, A – given value).

The process of searching rows satisfying the condition can be treated as the reasoning process: for each row of the table T, by comparing values of column X with the given value A, it is proposed to return the row in the result set or not.

Due to the nature of the X and A, filtering conditions can be grouped into:

- classic, where X and A take crisp values
- X takes crisp values and A takes fuzzy values, represented by membership functions
- X takes fuzzy values and A takes crisp values
- both, X and A take their values fuzzy (represented by membership functions)

The analysis of each case is somewhat different, but the result has always the same general form: for each row (describing the value of X) it is determined the compatibility degree between values X and A (or more generally, the compatibility degree, with which the value of X satisfies the given condition).

The cases mentioned above concern a simple condition for comparing one value to another. However, the WHERE clause of the classic SELECT statement may contain many simple filtering conditions joined by conjunction operators AND, disjunction operators OR, and these conditions may also be preceded by a negation operator NOT. In the WHERE clause containing fuzzy conditions, values of the fuzzy conjunction and disjunction can be calculated in various ways, the easiest is to apply the Zadeh norms [1] (t-norm for conjunction operations, calculated as the minimum of compared compatibility degrees, s-norm for disjunction operations, calculated as the maximum of compared compatibility degrees). When the condition is preceded by a negation operator NOT, the value of the compatibility degree is determined, for example, by subtracting from 1 an initial value of the compatibility degree.

Fuzzy Aggregation

One of the key features of the SQL language is the possibility to aggregate data [8]. Calculation of aggregate functions, like sum, avg, min, max, operating on fuzzy data requires the knowledge of arithmetic of fuzzy numbers, i.e. operations, such as: addition, division, comparison of fuzzy numbers. In the FuzzySQL language, we assume that fuzzy numbers are of the LR-type [2] and we have implemented appropriate operations and functions.

Implemented aggregate functions that operate on fuzzy data have the same names as classic aggregate functions, but differ in the type of argument – an appropriate aggregate function is run on this basis of the argument, performing operations on the fuzzy or crisp data.

Fuzzy terms can also occur in filtering criteria in the HAVING phrase, where filtering conditions are imposed on aggregate functions. In Fuzzy SQL, we have identified and implemented the following cases, which differ from each other with the processing procedure:

- aggregation of crisp data with a fuzzy condition in the HAVING clause
- aggregation of fuzzy data with a crisp condition in the HAVING clause
- aggregation of fuzzy data with a fuzzy condition in the HAVING clause
- usage of fuzzy quantifiers that operate on the selected group of rows

The process of determining the compatibility degree with the criteria of the filtering condition is the same as for the WHERE clause. A detailed description of these issues is presented in [9], [10].

Data Grouping

Data grouping is often a required step in data aggregation process. In SQL language notation, the GROUP BY phrase is responsible for the grouping process.

```
GROUP BY <column list>
```

The process groups rows with identical values in columns specified in the GROUP BY phrase, so each unique combination of values in specified columns constitutes a separate group [8]. This is the classical grouping.

In FuzzySQL, we also allow to group similar (not identical) data in grouping columns, and therefore, we use methods of fuzzy grouping. Since there is a possibility to store fuzzy data in a database, these data can be grouped with the use of classical or fuzzy grouping methods. There are different approaches to the data grouping that we have implemented:

- fuzzy grouping of crisp values
- classical grouping of fuzzy values
- fuzzy grouping of fuzzy values

In our work, we have extended the SQL language by implementing the following algorithms of fuzzy grouping:

- grouping with respect to linguistic values determined for the attribute domain
- grouping with respect to the arbitral division of the attribute domain
- fuzzy grouping based on the hierarchical clustering method
- fuzzy grouping based on the author's algorithm

A detailed description of these algorithms is presented in [12].

Analyzing possibilities of grouping according to attributes that store fuzzy values, we can first consider the same grouping methods that operate on crisp data. In the case, we assume the grouping process is performed on LR-type fuzzy numbers [2]. In the most rigorous solution, the grouping allows for all parameters describing fuzzy values. Therefore, selected fuzzy values are joined into one group, if they have the same set of parameters. More elastic solutions allow grouping of fuzzy data regarding their modal values. We can distinguish two approaches:

- grouping of LR-type fuzzy numbers – all fuzzy numbers with the same modal value form a group
- grouping of LR-type fuzzy intervals – all fuzzy intervals with the same range of modal values become a group

In the FuzzySQL, we have implemented the first approach.

Fuzzy Terms in Nested FuzzySQL Queries

In the preceding paragraphs of this paper we presented an analysis of the interpretation process for non-nested SQL queries containing fuzzy values. In the section, we focus on the nested fuzzy queries.

Fuzzy values in nested queries can appear in several places:

- in all phrases of a inner subquery
- in all phrases of the outer subquery
- in the binding condition of inner and outer subqueries

In nested queries, the process of searching rows that satisfy fuzzy filtering criteria, in both phrases – WHERE and HAVING, is implemented similarly to non-nested queries. However, there are some differences:

- nested queries often take a very complicated form
- there is a problem with placing the condition on the compatibility degree in case of fuzzy condition binding two subqueries

For these reasons, we had to consider different places where specific fuzzy values can occur in nested queries. Due to the presence of fuzzy values in binding condition of two subqueries, we have identified and implemented the following cases:

- comparison of the crisp value of the outer subquery with fuzzy value (or a set of such values) returned as a result of the execution of the internal subquery
- comparison of the fuzzy value of the outer subquery with crisp value (or a set of such values) returned as a result of the execution of the internal subquery
- comparison of the fuzzy value of the outer subquery with fuzzy value (or a set of such values) returned as a result of the execution of the internal subquery
- location of fuzzy values in the correlation condition binding correlated subqueries

FUZZYQ SYSTEM

Architecture of FuzzyQ System

The FuzzyQ [13] system allows building queries in the FuzzySQL language, submitting them to the database server and interpretation of results. The general architecture of the system is shown in Figure 1. In the FuzzyQ system we can distinguish two main modules. The first module is Query Analyzer, which provides the following functions:

- communication with a database
- submitting queries in SQL/FuzzySQL to the database
- receiving result sets for submitted queries
- presentation of a result set
- saving results to a file
- cooperation with the Query Wizard

The Query Analyzer module has been implemented in the Visual Studio.NET in the C# programming language.

The second module is the FuzzySQL Query Wizard, whose primary objective is to assist the user in the construction of fuzzy queries and such impact on his memory, that he/she could alone indicate the location of fuzzy elements in simple and complex FuzzySQL query, and he/she was able to construct such a query. This module implements a typical feature of teaching. It has been implemented in the Macromedia Flash MX environment, which allows programming in Flash language.

Query Wizard cooperates with Query Analyzer allowing users to keep checking validity of individual queries. A query constructed using the wizard is only a skeleton – wizard presents only a general form of the query. Complete FuzzySQL query is then transferred to the Query Analyzer. Before submitting a query to the database a user intervention is required – a user will adjust the general form of a query to a query, which wants to achieve. After the query is executed, the user obtains appropriate results or an error message.

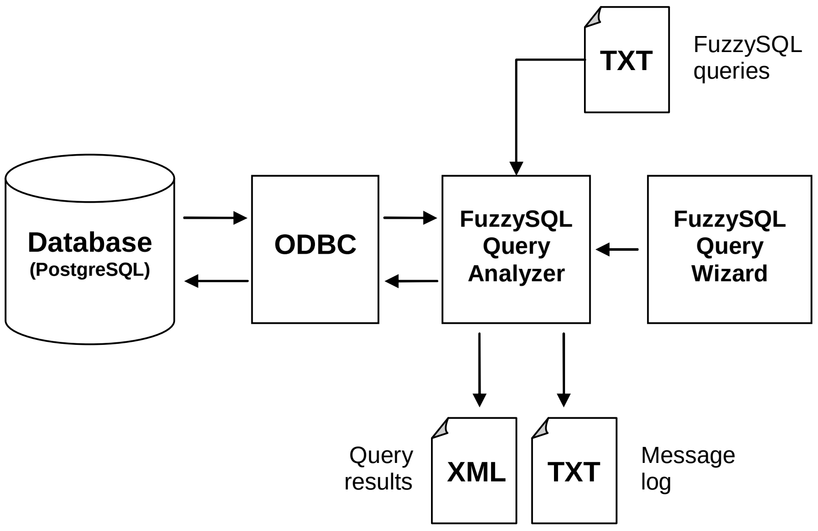


FIGURE 1
ARCHITECTURE OF FUZZYQ SYSTEM

FuzzyQ Query Wizard

Query Wizard is the educational part of the whole FuzzyQ system. With the use of the Query Wizard users (usually students) learn how to build a query using fuzzy terms. The main task of the wizard is to indicate the possible location of fuzzy values in the SQL SELECT statement. Query Wizard module is written in the Flash language using Action Script 2.0. This is an interactive movie, where the user plays the main role. From the user's decision depends how the construction of a query will proceed. The query skeleton is visible at every stage of the design (Figure 3). The Query Wizard begins from the SELECT clause. The user then decides whether the fuzzy element should occur after the particular keyword or not. The decision is done by pushing an appropriate button – FUZZY or NORMAL. Choosing one of them will move to the next clause of the SELECT statement. At the same time the second option, which was not chosen, is removed. If the phrase can not be followed by a fuzzy element, the user will see only one button (NORMAL). Such a case occurs after the FROM phrase. There is also the additional option that can be chosen. This is the green button

END. Choosing it causes the completion of the query design. The END button appears wherever the end of query is possible. Pressing the END button causes the query skeleton to be sent to the Query Analyzer module of the FuzzyQ system. The END button is also the only possible choice after the ORDER BY phrase, since this is the last possible phrase in the SELECT statement. After the WHERE and HAVING phrases users may also write a subquery on one of the sides of the equality/inequality operator. This is the user, who decides whether he/she wants to create a complex query by selecting one of the options: YES or NO. Figure 2 shows the skeleton of the query, which consists of fuzzy terms in the SELECT, WHERE and HAVING clauses.

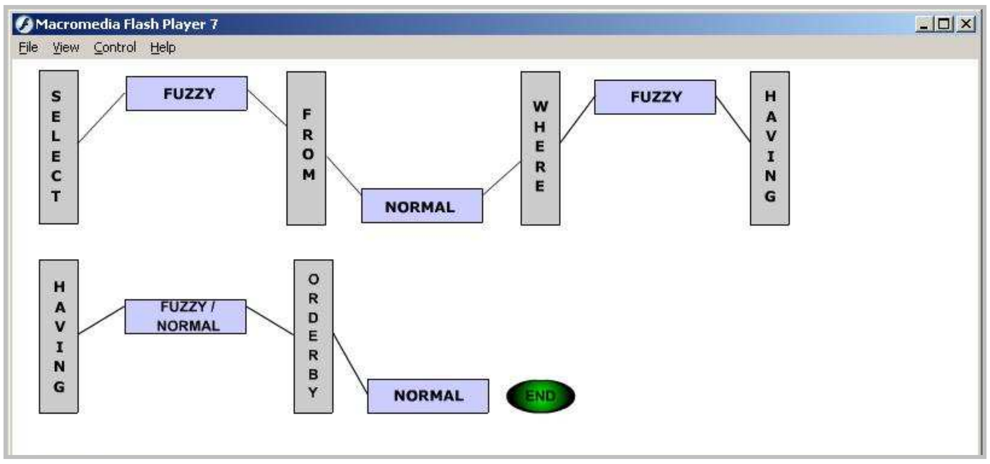


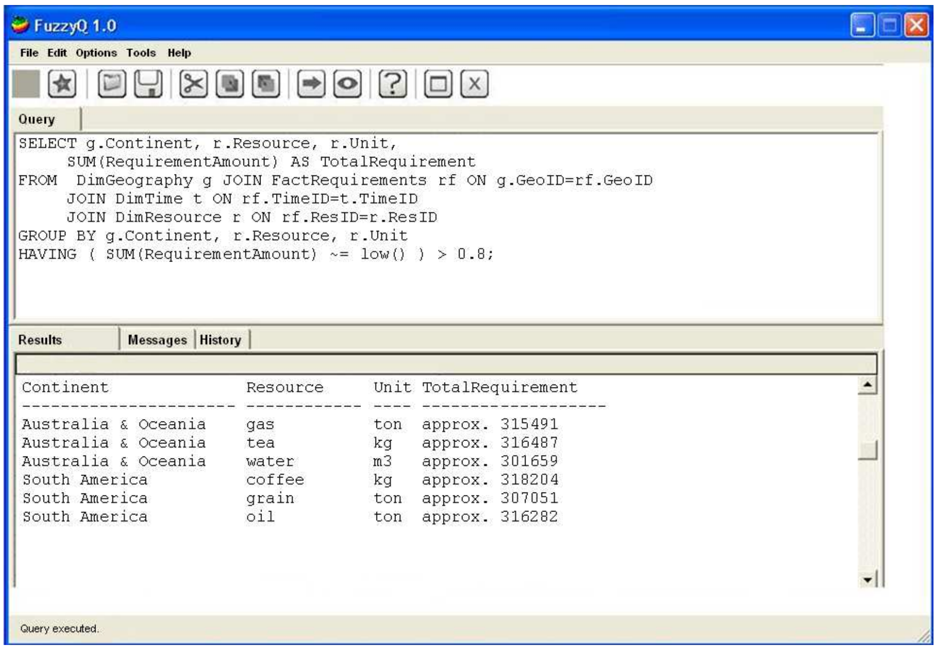
FIGURE 2
SKELETON OF QUERY WITH FUZZY TERMS IN SELECT, WHERE AND HAVING CLAUSES

FuzzyQ Query Analyzer

In Query Analyzer a user may construct a FuzzySQL query from the scratch or he/she should complete the skeleton returned from the Query Wizard. When the query is ready, it can be submitted to a database. If the syntax of the query is correct, the user obtains results. Otherwise, the user obtains an error message and has to improve the query.

In Figure 3, we can observe the window of the Query Analyzer with sample query and execution results. The example implements the following report:

Display total requirements for resources in particular continents. Show these continents, where total requirement is low, with a compatibility degree greater than 0.8.



```
SELECT g.Continent, r.Resource, r.Unit,
     SUM(RequirementAmount) AS TotalRequirement
FROM  DimGeography g JOIN FactRequirements rf ON g.GeoID=rf.GeoID
     JOIN DimTime t ON rf.TimeID=t.TimeID
     JOIN DimResource r ON rf.ResID=r.ResID
GROUP BY g.Continent, r.Resource, r.Unit
HAVING ( SUM(RequirementAmount) ~= low() ) > 0.8;
```

```
Continent              Resource     Unit TotalRequirement
---------------------- ------------ ---- -------------------
Australia & Oceania    gas          ton  approx. 315491
Australia & Oceania    tea          kg   approx. 316487
Australia & Oceania    water        m3   approx. 301659
South America          coffee       kg   approx. 318204
South America          grain        ton  approx. 307051
South America          oil          ton  approx. 316282
```

FIGURE 3
QUERY ANALYZER WITH SAMPLE FUZZYSQL QUERY AND EXECUTION RESULTS

In the example, we present the use of the sum aggregate function (SUM) operating on fuzzy data. We also show the fuzzy filtering condition in the HAVING clause, which consists of two fuzzy numbers. The result of the SUM function (in the HAVING clause), which is a fuzzy number, is compared to the fuzzy value *low requirement*. The comparison is implemented by the ~= equality operator, which returns the compatibility degree between two fuzzy numbers (Figure 4). In the final result we obtain rows having the degree greater than 0.8.

In Figure 4 we can observe different possible cases of comparison of two fuzzy numbers represented by trapezoidal membership functions. If two fuzzy numbers have only one common point (Figure 4a), this point determines the compatibility degree of these two fuzzy numbers. If there is more than one common point (Figure 4b) and we obtain more compatibility degrees, we choose the highest one.

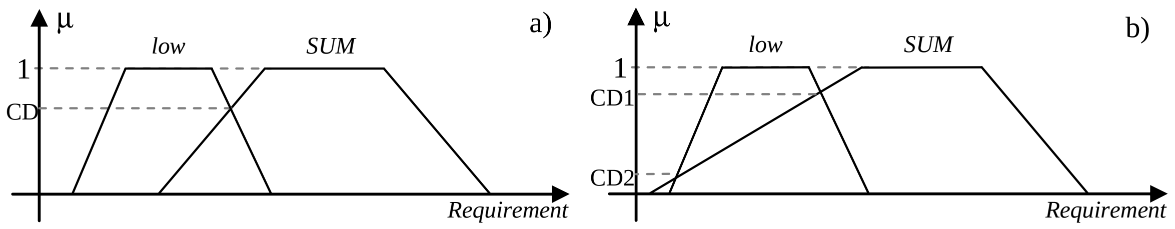


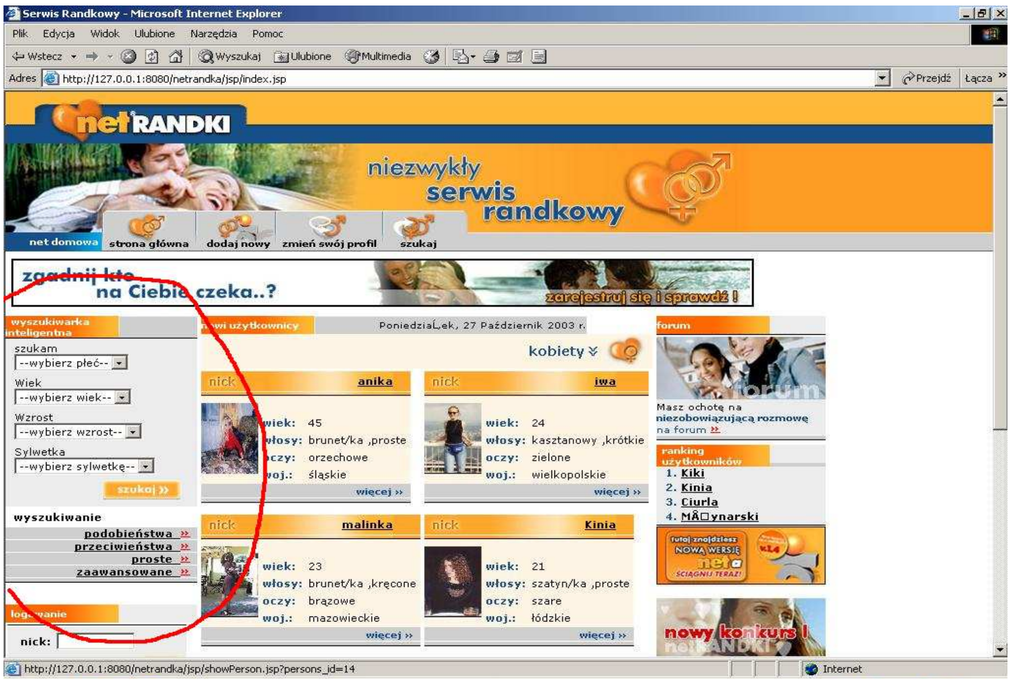
FIGURE 4
COMPARISON OF TWO FUZZY NUMBERS: A) ONE COMMON POINT, B) MANY COMMON POINTS

TRANSFER OF KNOWLEDGE GIVEN AT COURSES TOWARDS INDUSTRY AND COMMERCIAL PURPOSES

In the section, we present examples of various implementations of real software applications that have been developed for the last five years based on the courses of data processing with the use of fuzzy logic.

Rendez-Vous Agency Web Service

The purpose of this system is to connect people in pairs. The web service (Figure 5) is entirely developed in Polish language and it uses fuzzy terms in order to filter potential candidates for a date. In the filtering, a user can display candidates based on the fuzzy terms, like: *high height*, *long hair*, *slim physique*, *average age*, etc.



Serwis Randkowy - Microsoft Internet Explorer
Plik Edycja Widok Ulubione Narzędzia Pomoc
Wstecz Wyszukaj Ulubione Multimedia
Adres http://127.0.0.1:8080/netrandka/jsp/index.jsp
Przejdź Łącza
netRANDKI
niezwykły serwis randkowy
net domowa
strona główna
dodaj nowy
zmień swój profil
szukaj
zgadnij kto na Ciebie czeka..?
zarejestruj się i sprawdź !
wyszukiwarka inteligentna
nowi użytkownicy
forum
kobiety
szukaj »
wyszukiwanie
podobieństwa »
przeciwieństwa »
proste »
zaawansowane »
logowanie
nick:
tutaj znajdziesz NOWĄ WERSJĘ net ŚCIĄGNIJ TERAZ!
nowy konkurs ! netRANDKI
http://127.0.0.1:8080/netrandka/jsp/showPerson.jsp?persons_id=14
Internet

Poniedziałek, 27 Październik 2003 r.

szukam: --wybierz płeć--
Wiek: --wybierz wiek--
Wzrost: --wybierz wzrost--
Sylwetka: --wybierz sylwetkę--

nick: anika
wiek: 45
włosy: brunet/ka ,proste
oczy: orzechowe
woj.: śląskie
więcej »

nick: iwa
wiek: 24
włosy: kasztanowy ,krótkie
oczy: zielone
woj.: wielkopolskie
więcej »

nick: malinka
wiek: 23
włosy: brunet/ka ,kręcone
oczy: brązowe
woj.: mazowieckie
więcej »

nick: Kinia
wiek: 21
włosy: szatyn/ka ,proste
oczy: szare
woj.: łódzkie
więcej »

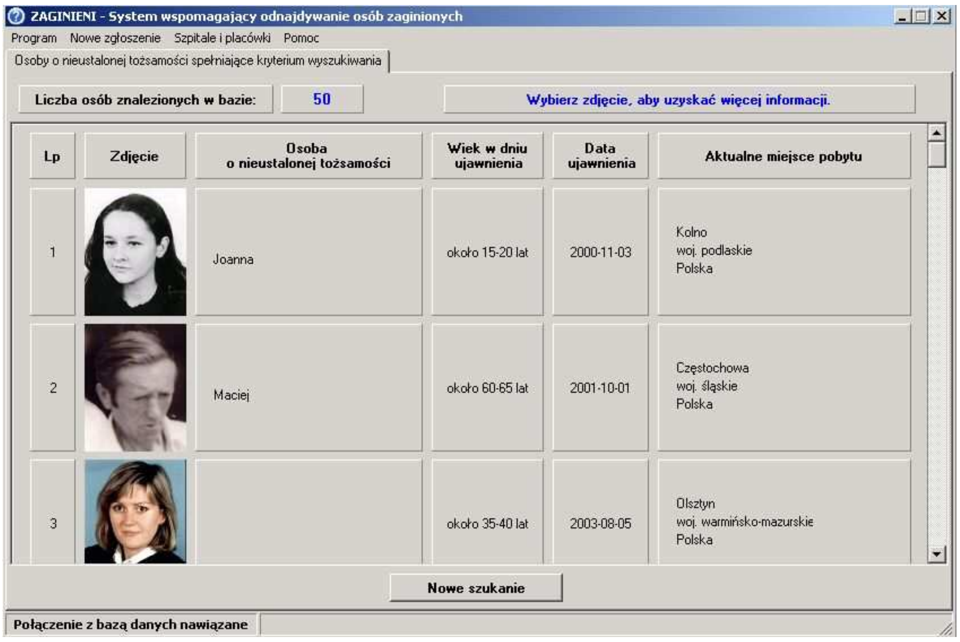
Masz ochotę na niezobowiązującą rozmowę na forum »

ranking użytkowników
1. Kiki
2. Kinia
3. Ciurla
4. MÅ□ynarski

FIGURE 5
RENDEZ-VOUS AGENCY WEB SERVICE

System for Searching Missing and Unidentified People

This system was developed with the cooperation with Police and Welfare Center. Its purpose was to support collecting and searching data describing missing and unidentified people. In this case, description of unidentified people is difficult to achieve in the standard way. Here is the place, where the fuzzy logic helps. Using the fuzzy logic we can describe people with the use of fuzzy values (like: *age about 15*, *bright hair*, etc.), store all this information in a database and improve possibilities of data (people) searching and retrieval. In Figure 6 we can see unidentified people retrieved from a database based on the given searching criteria.



ZAGINIENI - System wspomagający odnajdywanie osób zaginionych

Program
Nowe zgłoszenie
Szpitale i placówki
Pomoc
Osoby o nieustalonej tożsamości spełniające kryterium wyszukiwania

Liczba osób znalezionych w bazie: 50

Wybierz zdjęcie, aby uzyskać więcej informacji.

Lp	Zdjęcie	Osoba o nieustalonej tożsamości	Wiek w dniu ujawnienia	Data ujawnienia	Aktualne miejsce pobytu
1		Joanna	około 15-20 lat	2000-11-03	Kolno woj. podlaskie Polska
2		Maciej	około 60-65 lat	2001-10-01	Częstochowa woj. śląskie Polska
3			około 35-40 lat	2003-08-05	Olsztyn woj. warmińsko-mazurskie Polska

Połączenie z bazą danych nawiązane

FIGURE 6
SYSTEM FOR SEARCHING MISSING AND UNIDENTIFIED PEOPLE

Stock Exchange News Generator

The main goal of the Stock Exchange News Generator is an automatic production of exchange newses for newspaper readers as friendly as it is possible. The system assumes that readers of the financial information do not have to be financial experts. Therefore, by elimination of big amount of numerical information it is easier to interprete the given news for an ordinary reader. Compare two versions of the same news given below. The first version consists of big amount of numerical data and therefore, it is opaque. The second version was automatically generated by the Stock Exchange News Generator and it uses fuzzy logic and appropriate linguistic values in order to make the news more transparent.

Version 1
*During the last day, the main Polish stock index - WIG has **lost 330,61 points**, falling to the level of 25524,02 points. This is **a 1,28% decrease**. Today's exchange value for this index was 19154,00 thousands PLN. This means **a 30,14% decrease**. WIG20 has gained 7,68 points, and reached the level of 1891,90 points. This is **a 0,41% increase**. Exchange value for today was **40185,00 thousands PLN**, which means **a 110,36 percent increase**.*

Version 2
*During the last day, the main Polish stock index - WIG has noted a **high decrease**, falling to the level of 25524,02 points. Today's exchange value for this index **was little**. This means **a low decrease**. WIG20 has reached the level of 1891,90 points. This is **a low increase**. Exchange value for today was **above average**, which means **a high increase**.*

CONCLUDING REMARKS

Teaching in the areas that still remain under research requires new tools for better understanding of the analyzed field. FuzzySQL language and FuzzyQ system developed in the Institute of Informatics at the Silesian University of Technology in Gliwice, Poland and presented in the paper are examples of such tools. Our experiences show the functionality of the FuzzyQ system is so good that students become familiar with the theory of fuzzy sets applied in databases very fast and know how to use and implement the knowledge in real applications. Moreover, examples of these applications given in the paper confirm the transfer of the knowledge given at the courses towards the industry and commercial purposes becomes fluent.

REFERENCES

[1] Zadeh, L., A., “Fuzzy sets”, *Information and Control*, 8, 1965, pp. 338-353.

[2] Dubois, D., Prade, H., “Fundamentals of Fuzzy Sets”, *Kluwer Academic Publisher*, 2000.

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

[4] Kacprzyk, J., Szmidt, E., “Intiutionistic Fuzzy Sets in Intelligent Data Analysis for Medical Diagnosis”, *Springer-Verlag* Berlin Heidelberg, 2001.

[5] Rasmussen, D., Yager, R., “Summary SQL – A Fuzzy Tool For Data Mining”, *Intelligent Data Analysis*, Vol. 1, 1997.

[6] Yu, C.,T., Meng, W., “Principles of Database Query Processing for Advanced Applications”, *Morgan Kaufmann Publishers, Inc.*, 1998.

[7] Fasel, D., Zumstein, D., “A Fuzzy Data Warehouse Approach for Web Analytics”, *LNCS*, vol. 5736, *Springer Heidelberg*, pp. 276-285, 2009.

[8] Elmasri, R., Navathe, S., B., “Fundamentals of Database Systems”, *Addison-Wesley Publishing Company, World Student Series*, 2000.

[9] Małysiak, B., Mrozek, D., Kozielski, S., “Processing Fuzzy SQL Queries with Flat, Context-Dependent and Multidimensional Membership Functions”, *Proc. of 4th IASTED International Conference on Computational Intelligence*, Calgary, Canada. *ACTA Press*, pp. 36-41, 2005.

[10] Małysiak-Mrozek, B., Mrozek, D., Kozielski, S., “Processing of Crisp and Fuzzy Measures in the Fuzzy Data Warehouse for Global Natural Resources”, *IEA/AIE 2010, Part III, LNAI*, Vol. 6098, *Springer-Verlag Berlin Heidelberg*, pp. 616-625, 2010.

[11] Małysiak, B., “Methods of approximate object searching in database systems”, *PhD thesis*, Silesian University of Technology, Institute of Informatics, Gliwice, Poland, 2003.

[12] Małysiak-Mrozek, B., Mrozek, D., Kozielski, S., “Data Grouping Process in Extended SQL Language Containing Fuzzy Elements”, *Advances in Intelligent and Soft Computing*, Vol. 59, *Springer Verlag GmBH*, pp. 247-256, 2009.

[13] Dziedzic, B., “Fuzzy terms in simple and complex SQL queries”, *MSc thesis* supervised by B. Małysiak-Mrozek, PhD, Silesian University of Technology, Institute of Informatics, Gliwice, Poland, 2005.