Control and Cybernetics

vol. **25** (1996) No. 2

A fuzzy relaxation mechanism for relational dbms querying based on the ordered weighted averaging (OWA) operators

by

Dariusz A. Łazar

Intelligent Systems Laboratory, Roskilde University,
DK-4000 Roskilde, Denmark
Email: lazar@ibspan.waw.pl

Abstract: A query relaxation technique for a relational DBMS is considered. Fuzzy sets and ordered weighted averaging (OWA) operators are employed. An approach to the inclusion of importances in the OWA operators is discussed. An interactive method for the refinement of weights of the OWA operators is proposed.

Keywords: query relaxation, relational DBMS, fuzzy sets, database querying, importance, linguistic quantifiers, aggregation, OWA operator.

1. Introduction

In the traditional quering for relational DBMSs it is often impossible to adequately express the users' "real" intentions and needs. One of the reasons is an inherent incompatibility between a "precise" DBMS and an "imprecise" user. The traditional query formalism considerably limited the possibility of a proper expression of the user's intentions. For many users, who in everyday life use natural language as their (only) natural communication means, this has become an obstacle in an efficient use of data from the DBMS, and more generally from any information system. Therefore, there is an urgent need for constructing more natural and human-consistent systems that would be in line, as much as possible, with the users' natural behaviour.

The inconsistency between what the user wants to express and what is expressed in the query may lead to several critical situations in query processsing which may result in undesired answers (many of them may be empty).

While working with a standard relational DBMS the user often faces the problem that a given output does not satisfy him or her in general. However, the user would still clearly want to get some information from the database. To formulate a valuable answer he may use some relaxation (generalization)

technique (e.g. Guyomard and Siroux, 1989, Cuppens and Demolombe, 1991, Chu, Chen, 1992, Gaasterland, Godfrey and Minker, 1992, Kacprzyk, Zadrożny and Ziółkowski, 1989, Motro, 1988). The relaxation slightly softens (relaxes) initial query elements in order to obtain additional answers (if they exist) in the neighbourhood of the initial query.

Among various aproaches to this relaxation, there are a few that employ fuzzy sets. Fuzzy sets theory, Zadeh (1965), implanted into a query language allows us to define imprecise linguistic labels inside a query. A fuzzy-logic-based approach makes the system flexible which may help formulate a better query, and hence yield a better answer at the same time.

Basically, there are two general lines of research when discussing the application of fuzzy sets to the relational DBMSs. In the first one, which may be termed fuzzy querying (e.g. Bosc, Galibourg, Hamon, 1989, Kacprzyk, Zadrożny and Ziółkowski, 1989, Bookstein, 1980, Larsen, Yager, 1993, Tahani, 1977), it is assumed that the database is traditional (nonfuzzy) and only the query language is extended, through an "add-on" module employing some fuzzy tools, in order to handle imprecision in the query. The second one, which may be termed the fuzzy database approach (e.g. Zemankowa and Kandel, 1985, Anvari, Rose, 1987, Buckles and Petry, 1982) is concentrated on a modification of existing data models so as to store imprecise data. Ordinary databases are there extended with fuzzy and possibilistic elements. Having vague data inside the database, the query language must also be fuzzified.

Here, we only consider relaxation based on query language modification, while the database is not extended. Hence it appears that, using the above classification, our approach to the relaxation technique fits in the former class.

It should be noted that a basic inclusion of fuzzy sets in query language, corresponding to a fuzzification of nonfuzzy elements in the query, can be seen as some kind of initial relaxation. We refer to such a relaxation as a *primary fuzzy relaxation.*

However, in our work, we do not consider primary relaxation. We assume, that a given query is already fuzzfied, but nevertheless responds with empty answer. At that position, our aim is to relax this fuzzy query further in order to obtain a statisfactory answer. We call this *secondary fuzzy relaxation.* In principle, this problem was not discussed in literature before.

The fuzzy relaxations (primary and secondary) can be included in the query language in several ways using various types of fuzzy elements. Here, we consider the inclusion of the so-called ordered weighted averaging (OWA) operators, Yager (1988), which are viewed here as a universal tool for relaxing the aggregation mechanisms in the relational DBMS querying. Moreover, we extend the traditional OWA-based aggregation with an interactive refinement of the OWA's weights.

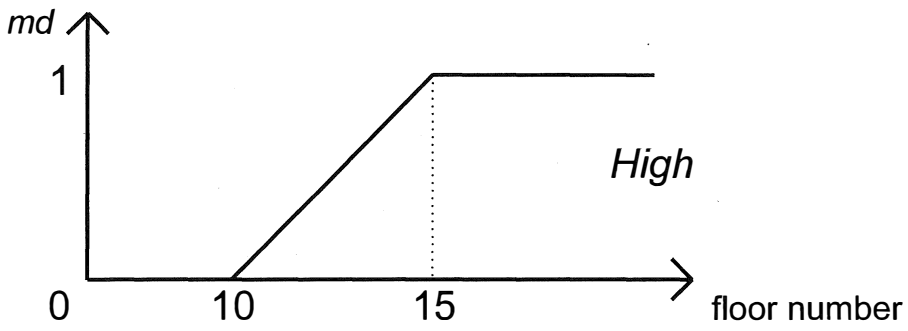


Figure 1. A fuzzy set representing an imprecise linguistic label High

2. Basic elements of fuzzy sets

Before we turn to the main subject we sketch basic concepts of fuzzy sets theory. Assume X is a set, then a fuzzy subset A of X is defined by a membership function $\mu_A : X \rightarrow [0,1]$ such that for any element $x \in X$, $\mu_A(x)$ indicates the degree to which the concept represented by A is satisfied by the element x.

Fuzzy sets theory may provide a description of imprecise linguistic labels by their corresponding fuzzy set. A fuzzy set includes only crisp (precise) values. Each of these crisp values has its own membership degree (to be denoted *md*) which indicates to what degree this particular precise value complies with the user's requirements. In Fig.1, we present a domain dependent (floor number in an apartament house) linguistic label "High". Thus, the floors below 10 are certainly not high, the floors above 15 are certainly high, and those in-between are high to an intermediate degree. Notice that such a representation is, first, context dependent, and second, subjective.

The basic operations on fuzzy sets are:

- the intersection

$$\mu_{A \cap B}(x) = T(\mu_A(x), \mu_B(x)) \tag{1}$$

where $\mu(x)$ is the membership function and T is a t-norm, i.e. a function $T : [0,1] \times [0,1] \rightarrow [0,1]$ which satisfies the following conditions:

1. $T(a, T(b,c)) = T(T(a,b),c)$ *associativity*
2. $T(a,b) = T(b,a)$ *commutativity*
3. $T(a,b) \geq T(c,d)$ if $a \geq c$ and $b \geq d$ *monotonicity*
4. $T(a,1) = a$ *boundary*

The most popular examples of t-norms are:

- $T_1(a,b) = \min(a,b)$
- $T_2(a,b) = a \cdot b$
- $T_3(a,b) = \max(0, a+b-1)$

- the union

$$\mu_{A \cup B}(x) = S(\mu_A(x), \mu_B(x)) \tag{2}$$

where $\mu(x)$ is the membership function and S is a t-conorm, i.e. a function $S : [0,1] \times [0,1] \to [0,1]$ which satisfies the following conditions:

1. $S(a, S(b,c)) = S(S(a,b),c)$ *associativity*
2. $S(a,b) = S(b,a)$ *commutativity*
3. $S(a,b) \geq S(c,d)$ if $a \geq c$ and $b \geq d$ *monotonicity*
4. $S(a,0) = a$ *boundary*

The most popular examples of t-conorms are:

- $S_1(a,b) = \max(a,b)$
- $S_2(a,b) = a + b - a \cdot b$
- $S_3 = \min(1, a+b)$

For more information of fuzzy sets and related topics, see, e.g. Kacprzyk (1986).

3. A fuzzy-logic-based relaxation in a query language

The relaxation of a query in a relational DBMS is based on the process that "softens" (relaxes) elements of the initial query by transforming them into other elements that represent a higher level of abstraction, or in other words are softer, i.e. less constraining and more flexible. As a result of this relaxation a new (relaxed) query is created. In comparison with the initial query, the relaxed query has less restrictive (wider) elements. Thanks to wider elements, the answer from the relaxed query relates to a "wider" class of elements in comparison with the initial query answer. Therefore, the relaxed query may result in additional answers (if they exist) that are in the neighbourhood of those of the initial query (possibly empty).

The softening of query elements by relaxation requires some efficient tools. In the literature several attempts have been made to construct a relaxation system for the querying languages. Among the systems that deal with the semantic transformation (relaxation) of the query, there are those that apply a hierarchy of concepts (e.g. Guyomard and Siroux, 1989, Chu and Chen, 1992), a collection of rules (e.g. Cuppens and Demolombe, 1991, Gaasterland, Godfrey and Minker, 1992), and finally imprecise linguistic labels (e.g. Kacprzyk, Zadrożny and Ziółkowski, 1989, Motro, 1988).

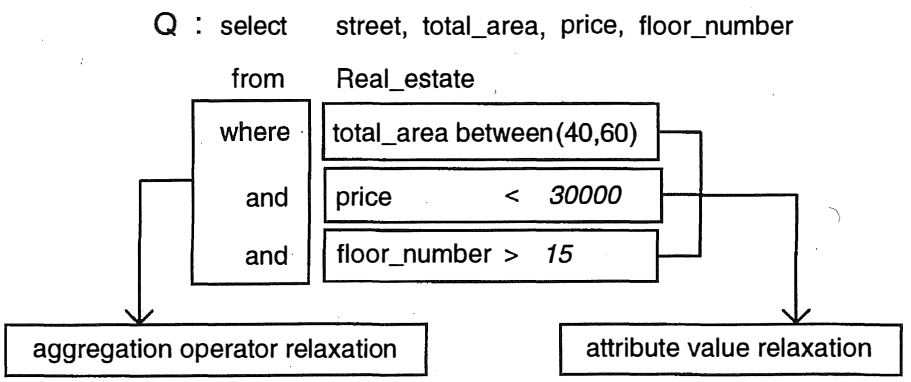


Figure 2. Elements of query Q which may be subject to fuzzy query relaxation

The relaxation approach that we propose in our work, called fuzzy querying relaxation, basically softens (relaxes) initial query elements by employing imprecise linguistic labels defined through fuzzy sets, and some "soft" aggregation techniques. If for some reasons (e.g. an empty answer), the user wants to relax the query, he replaces the initial query elements with new softer elements which are expressed in terms of imprecise linguistic labels. We can distinguish here two types of relaxation: the attribute value relaxation and the relaxation of the aggregation operator. Both the attribute value relaxation and relaxation of the aggregation operator use linguistic labels for relaxing specific elements of the query (cf. Fig.2).

Assume that we have typical real estate data concerning Warsaw, Poland. This database is described by several common attributes like "name_of_the_street", "total_area", "number_of_rooms", "floor_number", etc. The user constructs a query Q exemplified by one given in Fig.2 to be understood as: find an ideal flat that must have a moderate total area (between$(40, 60)$) and the price must be inexpensive (< 30000) and the floor number must be high (> 15).

Relaxation of the attribute value is implemented through the fuzzification of selection requirements (atomic conditions, criteria, ...). Each precise, initial attribute value in the query (e.g. floor number > 15) may be fuzzified with a related imprecise linguistic label (e.g. High, that is floor number is High). A fuzzy description of imprecise linguistic labels, besides the initial attribute value, comprises other precise attribute values that are in some way related to the initial one. Therefore, this kind of relaxation gives a possibility to search for other precise values that are in the neighbourhood of the initial precise values.

The relaxation of aggregation operators proceeds basically through the use of fuzzy linguistic quantifiers in line with the original ideas proposed in Kacprzyk

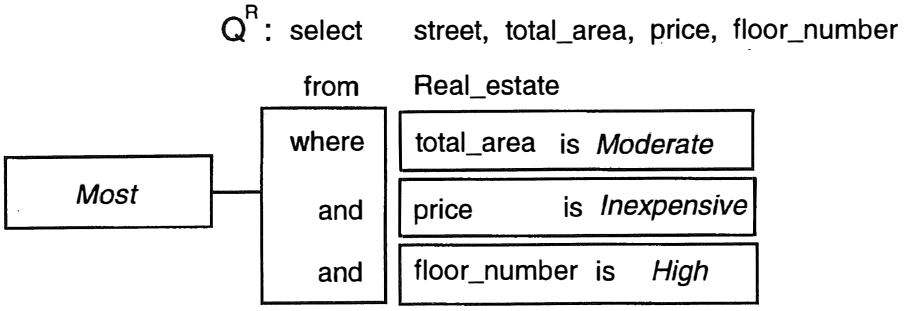


Figure 3. The relaxed query Q^R

and Ziółkowski (1986), Kacprzyk, Zadrożny and Ziółkowski (1989). The fuzzy linguistic quantifiers that we employ in the query language allow to perform not only the aggregation, where all (AND-like aggregation) or at least one (OR-like aggregation) query criteria are satisfied, but also to perfom an intermediate type aggregation between the two extremes where "most of", "at least half", "more or less", etc. query criteria are to be satisfied. Relaxing the query with linguistic quantifiers is equivalent to the replacement of the initial (traditional) quantifier (e.g. all) with a less restrictive quantifier (e.g. most of). In this paper we use novel means for dealing with a wide class of fuzzy linguistic quantifiers by representing them via the so-called ordered weighted averaging (OWA) operators, Yager (1988).

An example of the query Q (Fig.2) which was relaxed using the above mentioned aggregation-operator-related relaxation to the query Q^R is presented in the Fig.3 to be meant as: find a (quasi) ideal flat where most of conditions: a flat has moderate total area and the price is inexpensive and the floor number is High, are satisfied.

In the database systems verification of individual atomic conditions against data (records) included in the database may result in an evaluation of a so-called individual matching degree (imd_{kj}). The value of imd_{kj} indicates to what degree individual atomic condition k is satisfied by record j in the database.

In most of ordinary database systems, where we consider only precise atomic conditions, there are two possible situation, a record matches with an atomic condition or it does not. In case when a record j entirely matches an atomic condition k the individual matching degree $imd_{kj} = 1$ is delivered, otherwise, even if a record is very close to match, it receives $imd_{kj} = 0$. There are no intermediate values for imd_{kj}. Records that match all atomic conditions in the

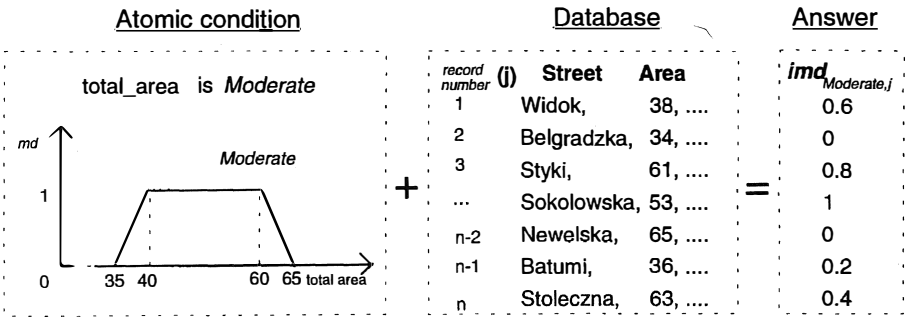


Figure 4. Determination of the individual matching degree for atomic condition: total area is Moderate

query ($imd_{kj} = 1$, for each k) are then selected for further aggregation while the other ones are rejected.

In the fuzzy querying system, along with the inclusion of imprecise linguistic labels inside a query, the algorithm for computing the individual matching degree was changed. Records may yield individual matching degrees from the unit interval $[0, 1]$: from the mismatched records with $imd_{kj} = 0$ through intermediate values for $imd_{kj} \in (0, 1)$ to full match with $imd_{kj} = 1$.

In Fig.4, we sketch the idea of determination of the individual matching degree for the atomic condition involving "total area" attribute that was relaxed with the linguistic label Moderate.

Now, we turn to the relaxation of the aggregation operator. At this level of relaxation, we may determine how many, or which, atomic conditions have to be taken into account in defining an overall satisfaction (overall matching degree) for the aggregated criteria. In standard query languages, the OR-like aggregation is supported by the max operator which chooses the highest value from the set of existing individual matching degrees, while the AND-like aggregation is supported by the min operator which takes the lowest individual matching degree. In the fuzzy querying relaxation, the OWA operators provide a representation of fuzzy linguistic quantifiers that are used in the relaxation of aggregation. Depending on the parameters used, the OWA operators may attain the aggregation behaviour which takes intermediate values between the lowest (min operator) value and the biggest value (max operator) of individual matching degrees. The answer, that is possibly delivered after the aggregation operation, provides a set of valued records accompanied with the overall matching degrees from the unit interval $[0, 1]$. The problem will be discussed

further.

The construction of a fuzzy relaxation system, including the attribute value relaxation and the relaxation of aggregation operator, allows us not only to obtain additional answers but also to distinguish among those records which comply better (with a higher overal matching degree) and worse (with a lower overal matching degree) with requirements of a query. The crucial point in building an effective fuzzy-logic-based relaxation system lies in an efficient fuzzy definition of linguistic labels which weaken atomic conditions in the query, and in the construction of a proper operator for the linguistic quantifier which supports the aggregation for compond queries.

4. The concept of an ordered weighted averaging (OWA) operator

The OWA operator is employed here as a general aggregation operator which exhibits the aggregating behaviour ranging from the classical AND-type one to the classical OR-type one, through all intermediate cases, depending on some parameter.

A weighted ordered averaging (OWA) operator F of dimension n has an associated with it weighting vector W

$$W = [w_1 \dots w_n]^T \tag{3}$$

such that

1. $w_j \in [0,1]$, for each $j = 1, \dots, n$;
2. $\sum_{j=1}^{n} w_j = 1$.

The aggregation $F(a_1, \dots, a_n)$ is then

$$F(a_1, \dots, a_n) = \sum_{j=1}^{n} w_j b_j \tag{4}$$

where b_j is the j^{th} largest element in the set of $\{a_1, \dots, a_n\}$.

The OWA operators have many interesting properties (Yager, 1988;1993a). First, notice that the weights are associated with a particular ordered position in the set of elements rather then with a particular element. The weights may therefore control the aggregating behaviour of the operator. For instance, for the following cases:

$$W^* = [1\ 0\ \dots\ 0]^T \tag{5}$$

$$W_A = \left[\frac{1}{n}\ \frac{1}{n}\ \dots\ \frac{1}{n}\right]^T \tag{6}$$

$$W_* = [0\ 0\ \dots\ 1]^T \tag{7}$$

it is easy to show that these weight vectors correspond to the following aggregations, Yager (1988), respectively:

- W^* corresponds to the max-type (OR-like) aggregation
$$F^*(a_1, \ldots, a_n) = \max_{j=1,\ldots,n} a_j \tag{8}$$
- W_A corresponds to the mean-type aggregation
$$F_A(a_1, \ldots, a_n) = \frac{1}{n} \sum_{j=1}^{n} a_j \tag{9}$$
- W_* corresponds to the min-type (AND-like) aggregation
$$F_*(a_1, \ldots, a_n) = \min_{j=1,\ldots,n} a_j \tag{10}$$

and

$$F_*(a_1, \ldots, a_n) \leq F(a_1, \ldots, a_n) \leq F^*(a_1, \ldots, a_n) \tag{11}$$

that is

$$\min_{j=1,\ldots,n} a_j \leq F(a_1, \ldots, a_n) \leq \max_{j=1,\ldots,n} a_j \tag{12}$$

Thus, the OWA operator is equivalent to the min operation for $w_n = 1$, and $w_j = 0$ for all $j \neq n$, the max operation for $w_1 = 1$, and $w_j = 0$ for all $j \neq 1$, and the arithmetic mean for $w_i = 1/n$, $\forall i$. For weights "in-between" an intermediate aggregation may be obtained.

As we have presented above, the concept of weights employed in the OWA operator makes easy and universal the manipulation of aggregation. Yager (1988) proposed two basic measures in order to guide and control the aggregation. With the first one, called the measure of orness, one can calculate the degree to which the operator is OR-like or AND-like in his behaviour. The main idea is that: if the orness value is greater than 0.5, then the OWA operator is considered as more OR-like than AND-like, otherwise it is considered as more AND-like than OR-like. That is, the measure of orness for a weighting vector W (3) is

$$orness(W) = \frac{1}{n-1} \sum_{i=1}^{n} ((n-1) \cdot w_i) \tag{13}$$

and the measure of andness of an OWA operator is defined as a complement of the orness measure, i. e.

$$andness(W) = 1 - orness(W) \tag{14}$$

Yet another measure, the entropy (dispersion), measures the degree to which all the elements aggregated are taken into account equally, and is defined as

$$Disp(W) = -\sum_{i} w_i \ln w_i \tag{15}$$

As we outlined above, before we use the OWA operators, we need to determine the weights. Yager (e.g. Yager, 1993a) presented several approaches to

obtain the weighting vector. On the basis of those two measures (the orness and the entropy), he introduced a whole family of parameterized OWA operators that have the property of reducing the process of selecting the weighting vector to one or two parameters. In addition, he proposed a mechanism allowing to learn the weights from data by determining the value of the parameters that minimize some error function.

5. The concept of importance

In practice, in addition to linguistic labels that provide an imprecise description for values in atomic conditions in the query, the user may also want to add information about importance of individual elements (criteria of querying). Once the user searches the database with respect to several criteria, he may consider some of them as more important than the others (Fig.5). Although the concept of importance has a direct influence on an individual criterion, it is rather considered in the context of aggregation.

In the context of the OWA operators the inclusion of importances is still an open issue. The idea of how to include importances was proposed by Yager (1987) as a general framework. He describes the inclusion of importances in an individual criterion as closely related to the reduction of the influence on the query aggregation of these atomic conditions values' which are considered of low importance.

In case of the AND-like aggregation, the element with the lowest value plays the most significant role, so a natural solution is to transform this value into a high one to reduce its influence on the query aggregation. In the OR-like aggregation, on the other hand, it is the large values that play substantial role during the process of aggregation. Hence, we need to reduce those which are less important.

Thus, the structure of importance inclusion is different for the AND-like and the OR-like aggregation. Since the OWA operator acts like a universal aggregator, between the above two extremes, the inclusion of importances should be changeable along with the changes of the aggregation character.

In the source paper, Yager (1988), on the OWA aggregation the following approach was proposed. We assume the following function:

$$F(x) = F(a_1, \ldots, a_n) \tag{16}$$

where the arguments $(a_1, \ldots, a_n)$ and the vector of importances $[\alpha_j]_{j=1,\ldots,n}$ are modified in the following way

$$b_j = H(a_j \alpha_j) \tag{17}$$

where $H : [0,1] \times [0,1] \to [0,1]$ is, e.g.,

$$b_j = (\alpha_j \vee p) \cdot (a_j)^{\alpha_j \vee p} \tag{18}$$

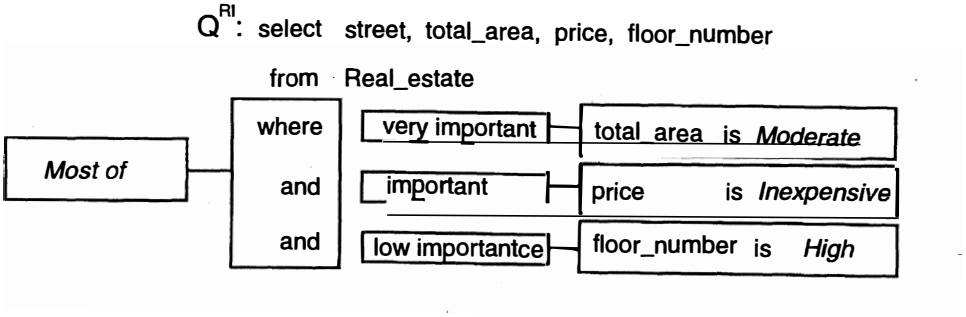


Figure 5. The query Q^R which is subject to the inclusion of importances

where: q is the degree of orness (13), and p is the degree of andness (14).

For instance, we have:

- for the pure AND, i.e. for $W_n = 1$, $p = 1$, $q = 0$,
$$F_1(x) = \min_{j=1,\ldots,n} a_j{}^{\alpha_j} \tag{19}$$
- for the pure OR, i.e. for $W_1 = 1$, $p = 0$, $q = 0$,
$$F_2(x) = \max_{j=1,\ldots,n} \alpha_j a_j \tag{20}$$

A modification of the above was given by Yager in the framework of multi-criteria decision making, Yager (1993b), as

$$b_j = \alpha_j a_j + (1 - \alpha_j)p \tag{21}$$

For instance, we have:

- for $\alpha \in (0, 1)$
$$F_{3a}(x) = F_2(x) = \max_{j=1,\ldots,n} \alpha_j a_j \text{ for } p = 0 \text{ (the pure OR)} \tag{22}$$
$$F_{3b}(x) = \min_{j=1,\ldots,n} a_j(\alpha_j - 1) + 1 \text{ for } p = 1 \text{ (the pure AND)} \tag{23}$$
- for $\alpha = 0$
$$F_4(x) = p \tag{24}$$
- for $\alpha = 1$
$$F_5(x) = a_j \tag{25}$$

EXAMPLE 5.1 Assume that: $A = (1; 0.8; 0)$, $W = (0; 0.5; 0.5)$, $\alpha_1 = 1$, $\alpha_2 = 0.5$, $\alpha_3 = 0$. The OWA-based aggregation yields $OWA(A, W) = 0.4$, and the degree of andness (14) is

$$andness(W) = 0.75$$

Then, (21) yields

$$\begin{aligned} a_1 &= 1 \cdot 1 + (1-1) \cdot 0.75 = 1 \\ a_2 &= 0.5 \cdot 0.8 + (1-0.5) \cdot 0.75 = 0.775 \\ a_3 &= 0 \cdot 0 + (1-0) \cdot 0.75 = 0.75 \end{aligned}$$

and the new vector A_α is

$$A_\alpha = [1\ 0.775\ 0.75]^T$$

Finally, for the OWA-based aggregation with importance, $OWA(A_\alpha, W)$, according to (4), we use the new vector A_α, and obtain

$$OWA(A_\alpha, W) = 1 \cdot 0 + 0.775 \cdot 0.5 + 0.75 \cdot 0.5 \cong 0.763$$

Notice that the inclusion of importances α_1, α_2, α_3 in the aggregation increased the overall aggregated value.

In our discussion, it was implicitly assumed that the importances are provided by the user as numeric values from $[0, 1]$. This may be not really comfortable, and a linguistic scale may be preferable, see Larsen and Yager (1993).

6. The concept of a linguistically quantified aggregation

In this part, we want to describe the relaxation of aggregation operators based on a linguistic quantifier which represents the second level of relaxation in our fuzzy querying system.

In practice, very often the user needs to manipulate the query aggregation process. Traditional query constructs do not give the user too much of a choice when deciding which aggregation to perform as only two options exist, the OR-like aggregation that chooses the biggest partial matching degree the AND-like aggregation that takes the lowest one. No intermediate cases are supported by most of standard systems. A linguistic quantifier guided aggregation would provide here general enough tools.

The standard query languages allow for two quantifiers only, the universal quantifier "all", and the existential quantifier "at least one". On the other hand, the human discourse comprises a large set of quantifiers as, e.g., almost all, most, nearly half, around n, etc. To bridge this gap, Zadeh (1983) introduced the idea of a linguistic quantifier along with its formal representation within the fuzzy sets theory. Zadeh suggested a fuzzy subset Q of $[0, 1]$ (or more generally, the real line) as a description for a proportional linguistic quantifier (e.g. most).

When we disscuss the linguistic quantifier guided aggregation, we always consider the evaluation of a linguistcally quantified proposition. Quantified propositions exemplified by "almost all flats are small", "most houses are expensive", "nearly a half of apartments are two-roomed" etc. are usually written as:

$$QY's \text{ are } F \tag{26}$$

where Q is a linguistic (imprecise) quantifier (e.g. almost all), Y represents a class of objects (e.g. flats) and F is a property that must be satisfied by objects Y (e.g. small). The problem is then to find the truth value of such a statement which, due to Zadeh, proceeds in two steps: In the first step we calculate the following expression:

$$r = \frac{1}{n}\sum_{j=1}^{n} a_j \text{ where } a_j = \mu_A(Y_j),\ y_j \in Y \tag{27}$$

$$F_Q(a_1, \ldots, a_n) = \mu_Q(r) \tag{28}$$

For example, the linguistic quantifier "most" can be defined by the following membership function

$$\mu_{most}(y) = \begin{cases} 1 & \text{for} \quad y \geq 0.8 \\ 2y - 0.6 & \text{for} \quad 0.8 > y \geq 0.3 \\ 0 & \text{for} \quad y < 0.3 \end{cases} \tag{29}$$

In 1988, Yager proposed to use the OWA aggregation operator (4) as a computational tool for the linguistically quantified propositions Yager (1988). In the first step of evaluating the truth of a linguistically quantified statement we calculate the weighting vector W (3) associated with the quantifier Q:

$$w_i = \mu_Q\left(\frac{i}{n}\right) - \mu_Q\left(\frac{i-1}{n}\right) \tag{30}$$

next we compute the OWA based aggregation as in (4) using weights calculated in the first step:

$$F_Q(a_1, \ldots, a_n) = \sum_{j=1}^{k} w_j b_j \tag{31}$$

In general, (30) provides a convenient definition of OWA weights corresponding to a given fuzzy linguistic quantifier.

The idea of using fuzzy linguistic quantifiers in a query language was suggested by Kacprzyk, Zadrożny and Ziółkowski in a series of papers (e.g. Kacprzyk and Ziółkowski, 1986). When there exists a table R defined on a set of attributes $\{A_1, \ldots, A_n\}$ and represented by a collection of tuples $\{t_1, \ldots, t_n\}$, the essence of their proposal is the determination of the extent to which a tuple satisfies the following expression: "⟨quantifier⟩ out of $\{A_1\ \langle \text{comp}_1\rangle\ v_1, \ldots, A_n\ \langle \text{comp}_n\rangle\ v_n\}$ match" where ⟨quantifier⟩ is a linguistic imprecise quantifier (e.g. almost all, most) and $\langle \text{comp}_1\rangle$ is a comparator (operator). Each evaluation of such an expression (for each tuple) returns an overall matching degree (*omd*). To represent linguistic quantifiers (⟨quantifier⟩), Zadeh's approach to fuzzy linguistic quantifiers was employed, yielding for each tuple i

$$omd_j = \mu_Q\left(\frac{1}{k}\sum_{k=1}^{n} imd_{kj}\right) \tag{32}$$

where μ_Q is a fuzzy definition of linguistic quantifier Q, imd_{kj} stands for a partial matching degree of criterion k in tuple j. This work was then practically implemented in the system called FQUERY III+ presented in Kacprzyk, Zadrożny and Ziółkowski (1989).

7. An interactive refinement of the OWA's weights

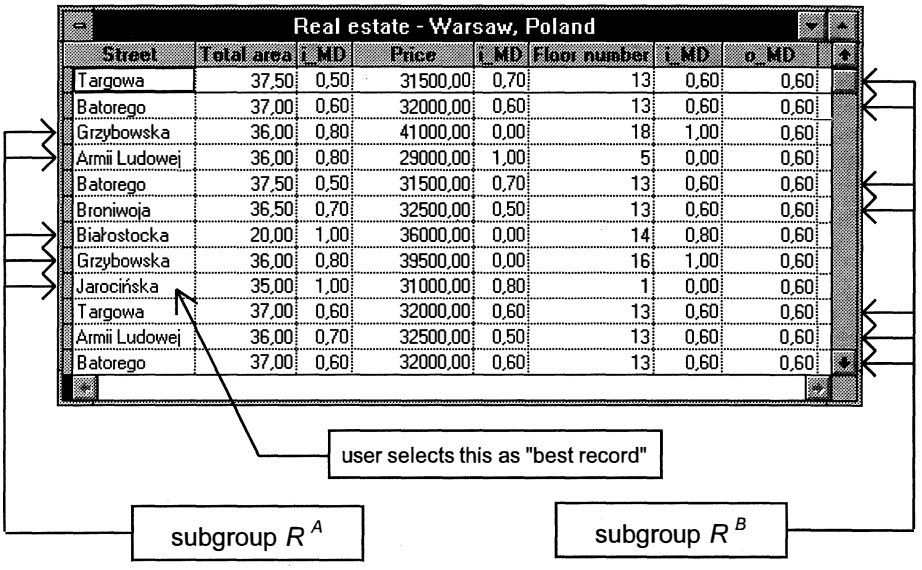
In this section we propose to employ Yager's OWA operators for representing linguistic quantifiers in the query language, extending the traditional OWA-based aggregation with an interactive refinement mechanism for the OWA's weights.

The choice of an aggregation operator, that is usually made a priori, is often incompatible with the very essence of aggregation the user has in mind. One of the crucial features of human behaviour is instablity and context-dependency. In general, the aggregation operator depends on the particular domain and time instant, and is seldom satisfactory in general. This implies efforts to seek more "appropriate" aggregation, and especially Yager has investigated this problem in detail (e.g. Yager, 1993a;b;1994). Yager (1994) exemplified that in some cases the user may exhibit different and untypical aggregation behaviour. For example, when the user deals with the medical diagnosis, the appearance of a set of indications symptomatic for a disease will make him or her more confident in diagnosing a patient as having the disease than any one of those indications alone, and the lack of that set of indications will make the user more confident that a patient is not having a specific disease.

Depending on the weights of the OWA operator, we may implement a specific type of their corresponding linguistic quantifier for guiding the aggregation. While the OWA operators are a comfortable and flexible tool for guiding the aggregation from the conceptual point of view, we must first determine appropriate weights what may be nontrivial. Yager (1993a) exhaustively considered this problem showing different approaches to obtaining these weights.

Basically, the main weak point of the linguistcally quantified OWA-based aggregation is that the OWA weights definition (30) for the aggregation is of an ad hoc type.

Before we proceed, let us continue with our example of the real estate database. Assume that a relaxed query Q^R (Fig.3) reflects the user's typical subjective concept of a desired apartment. His search in the real estate database with the query Q^R resulted in an empty answer (assuming threshold: $\lambda = 0.6$). In such a situation, the user will try to search again for entries (data) that would be close and satisfactory (threshold equals at least to 0.6) to his former concept of the desired apartment. Let the user decide to relax the query Q^R (Fig.3) using the relaxation of the aggregation operator. Then, the aggregation of the query Q^R, quantified with the linguistic quantifier "most", is softend with the another (less "rigid") linguistic quantifier "medium" (corresponding somehow to the average-type aggregation) resulting in new relaxed query Q^R_{medium}. The



Street	Total area	i_MD	Price	i_MD	Floor number	i_MD	o_MD
Targowa	37,50	0,50	31500,00	0,70	13	0,60	0,60
Batorego	37,00	0,60	32000,00	0,60	13	0,60	0,60
Grzybowska	36,00	0,80	41000,00	0,00	18	1,00	0,60
Armii Ludowej	36,00	0,80	29000,00	1,00	5	0,00	0,60
Batorego	37,50	0,50	31500,00	0,70	13	0,60	0,60
Broniwoja	36,50	0,70	32500,00	0,50	13	0,60	0,60
Białostocka	20,00	1,00	36000,00	0,00	14	0,80	0,60
Grzybowska	36,00	0,80	39500,00	0,00	16	1,00	0,60
Jarocińska	35,00	1,00	31000,00	0,80	1	0,00	0,60
Targowa	37,00	0,60	32000,00	0,60	13	0,60	0,60
Armii Ludowej	36,00	0,70	32500,00	0,50	13	0,60	0,60
Batorego	37,00	0,60	32000,00	0,60	13	0,60	0,60

Figure 6. A sample answer to the query Q^R_{medium}

answer R for the query Q^R_{medium} includes records being the closest match to Q^R (threshold: $\lambda = 0.6$).

However, as we look closer at the records R (Fig. 6), we may notice that some of them may represent an unequal degree of satisfaction to the user. For example, in the answer R we may distinguish two distinct subgroups of records R^A and R^B. Although they have similar values of the overall matching degree ($omd_j = 0.6$) they have considerably different distributions of values of the individual (partial) matching degrees (imd_{kj}). In case of subgroup R^A, values of the individual matching degree (imd_{kj}) are distributed **unequally**: some of them have very high values (near or equal to 1) while the others have very low values (near or equal to 0). In case of subgroup R^B, values of the individual matching degree (imd_{kj}) are distributed **equally**: all of them are close to 0.6. In our example, we cannot judge "objectively", i.e. taking into account this answer alone, which of the records in R (Fig. 6) comply better or worse with the user's demand, though our subjective feeling may be decisive. The ultimate judge is clearly the user himself who will take the final decision as to which records in the answer are the most suitable from his point of view.

From such a point of view the answer R can only be viewed as an collection of admissible records that may be interesting to the user, i.e. as a point of departure. Having such an admissible answer, the user may need to refine the aggregation in order to obtain a more consistent (with his or her feeling) answer.

Admissible answers may thus be a reference point for starting to shape what we may call an ideal answer. Hence, our aim is to provide the ability to construct an OWA operator, i.e. W in (3), that in the best way would model the aggregation for the current collection of data.

Pointing out the records which are (considered by the user) the best in an admissible answer would give hints for that refining. Clearly, the most suitable paradigm is here a interactive, man-machine cooperation, and this is generally accepted, e.g., in new releases of relational DBMSs' (e.g. CASE Generator 1991, FoxPro2.6., Application Builders 1994).

Finally, we suggest a so-called refinement process of the OWA's weights, a procedure for refining the OWA-based aggregation that involves a basic form of user – DBMS interaction. Since it is the weigthing vector $W = [w_1 \ \ldots \ w_n]^T$ which has a substantial influence on the definite shape the OWA-based aggregation, we adjust the initial parameters w_i of the vector W in order to obtain an aggregation leading to an ideal answer. We modify the weights W so as to reinforce (increase) values of the overall matching degrees (omd_j) for records that are considered by the user as the most satisfactory among the acceptable ones in an admissible answer. Although during the interaction the user is asked to choose one best record, other records that have similar distributions of values of the individual matching degrees (e.g. R^A and R^B in Fig.6) are adjusted as well. Thanks to such a weight refining, the best record(s) may increase their overall matching degree values. Increasing the values of the best records moves them to the top of the collection of discriminated records that stand for the answer.

A weights refining procedure

The weights refining procedure module is illustrated in the scheme of refinement process of the OWA's weights (Fig.7). In our approach, we use mainly the steepest descend method. We suggest this method in order to calculate an error measure to provide the change of parameter that we use for minimizing the difference between the observed overall matching degrees (an admissible answer) and the desirable one (an ideal answer). The error measure between the observed output $a_1w_1 + \ldots + a_nw_n$ and the desired output vs (value of satisfaction) is

$$\epsilon_w = \frac{1}{2}[vs - (a_1w_1 + \ldots + a_nw_n)]^2 \tag{33}$$

In order to adjust the value for each parameter w_i, we compute first the following changes of w_i's

$$\Delta w_i = \epsilon_{w_i} = -[vs - (a_1w_1 + \ldots + a_nw_n)]a_i \tag{34}$$

and we obtain the vector

$$\Delta W = [\Delta w_1 \ \ldots \ \Delta w_n]^T \tag{35}$$

Then, we update the initial vector W with the vector ΔW multiplied by a constant value LR (learning rate)

$$W_{new} = W_{old} - \Delta W \cdot LR \tag{36}$$

The last step of this procedure is the weight normalization $norm(W_{new})$ which yields

$$\bar{W}_{new} = [\bar{w}_1 \ \ldots \ \bar{w}_n] \tag{37}$$

where

$$\bar{w}_i = \frac{w_{inew}}{\sum_{k=1}^{n} w_{inew}} \tag{38}$$

The vector $\bar{W}_{new}$ comprises the new refined weights that can be used for determining the new OWA-based aggregation.

In practice the refinement procedure starts with a simple form of the user – DBMS interaction. During an interaction session within the browser module in the refinement process of OWA's weights (Fig.7), the user chooses the best record from a collection of records that are provided as an answer. Then, having parameters of the best record (the vector A) obtained in the browser module and parameters of the weighting vector W from the OWA's weights module, new weights are calculated in a weights refining procedure module according to the algorithm described above.

EXAMPLE 7.1 Assume that we continue with the the answer R obtained from the query Q^R_{medium}. Assume also that among the records belonging to the answer R we have chosen as the best the record with $A = (1; 0.8; 0)$. The weighting vector is $W_{initial} = (0.33; 0.33; 0.33)^T$. Then, we calculate the changes of weights $\Delta W = (w_1, \ldots, w_n)^T$ according to (34) and require $vs = 1$. Thus

$$\begin{aligned}
\Delta w_1 &= \epsilon_{w_1} = -[1 - (1 \cdot 0.33 + 0.8 \cdot 0.33 + 0 \cdot 0.33)] \cdot 1 = -0.406 \\
\Delta w_2 &= \epsilon_{w_2} = -[1 - (1 \cdot 0.33 + 0.8 \cdot 0.33 + 0 \cdot 0.33)] \cdot 0.8 = -0.3248 \\
\Delta w_3 &= \epsilon_{w_3} = -[1 - (1 \cdot 0.33 + 0.8 \cdot 0.33 + 0 \cdot 0.33)] \cdot 0 = 0
\end{aligned}$$

$$\Delta W = \begin{bmatrix} \Delta w_1 \\ \Delta w_2 \\ \Delta w_3 \end{bmatrix} = \begin{bmatrix} -0.406 \\ -0.3248 \\ 0 \end{bmatrix}$$

So, we transform the initial vector W with the vector (ΔW) multiplied by constant value LR according to (36). Initially, lacking experience, we set $LR = 0.95$, i.e.

$$W_{new} = \begin{bmatrix} 0.33 \\ 0.33 \\ 0.33 \end{bmatrix} - \begin{bmatrix} -0.406 \\ -0.3248 \\ 0 \end{bmatrix} \cdot 0.95 \cong \begin{bmatrix} 0.716 \\ 0.639 \\ 0.33 \end{bmatrix}$$

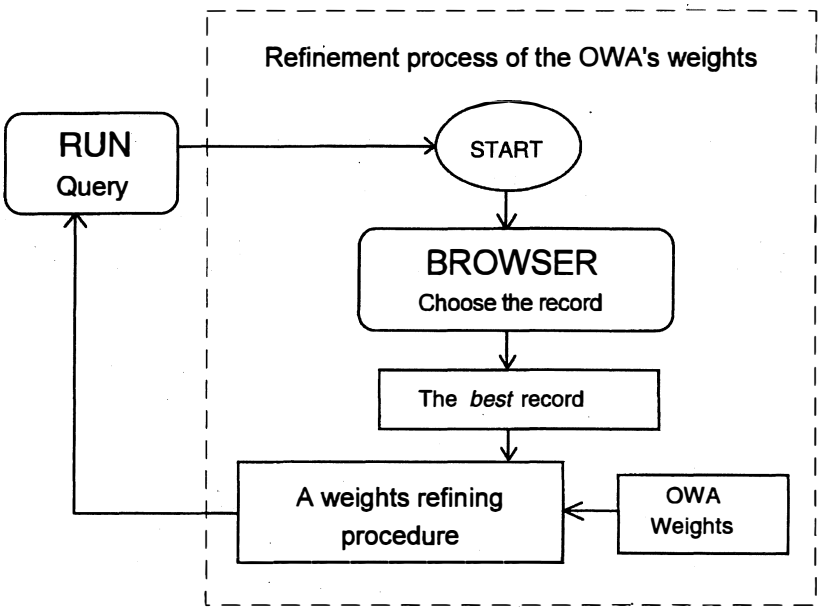


Figure 7. Scheme of the refinement process of the OWA's weights

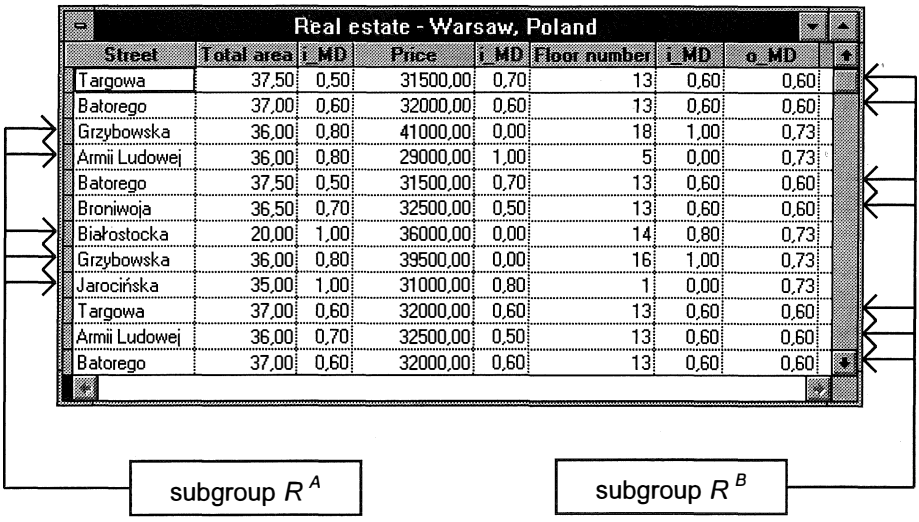


Figure 8. A sample of the refined answer to the query Q^R_{medium}

and the normalization (37) gives

$$\bar{W}_{new} = \begin{bmatrix} 0.425 \\ 0.379 \\ 0.196 \end{bmatrix}$$

and

$$OWA(A, \bar{W}_{new}) \cong 0.73$$

In comparison with the initial value of aggregation $OWA(A, W_{initial}) \cong 0.6$, the OWA-based aggregation with the new vector $\bar{W}_{new}$ increased its value, $OWA(A, \bar{W}_{new}) = 0.732$, which is quite natural. When we look at the rest of the refined answer R from Fig.6 we may notice (Fig.8), that records which have unequal distributions of values of the individual matching degrees (R^A) increased their OWA-based aggregation values (*omd*), and this is also the case of the best record derived in Example 2.

In case of subgroup R^A, where values of the individual matching degree (imd_j) are distributed **unequally**, the records increased their overall matching

degrees. In case of subgroup R^B, where values of the individual matching degree (imd_j) are distributed **equally**, the records remain unchanged. The reason is that this particular user paid more attention to higher values of only some of the attributes, and less attention to their even distribution.

Notice that a similar general idea of modification of an aggregation operator's parameters has also been proposed in the context of information retrieval by Naito, Ozawa, Hayashi and Wakami (1994). However, they applied the procedure, also based on the minimization of a squared error, to the tuning of parameters in some T-norm and S-norm only. In this paper the method is applied to the tuning of an OWA operator.

It should be noticed that a similar method for the determination of OWA's weights has been proposed by Yager (1994). However, the rationale is there different. Namely, Yager starts with a set of examples "values of arguments and results of their OWA aggregations", then – by using the degree of orness – finds some "ideal" result of OWA aggregation. Then he uses minimization of the sum of squared errors between the particular results of OWA aggregation and the above ideal result. So, Yager does not have OWA weights initially. In this paper, the rationale is different. Namely, having obtained some results of OWA aggregation, with OWA weights, we tacitly assume that there may be the same results of aggregation (in the sense of values) which are, though, different from the point of view of the user (e.g. because not all criteria have been accounted for). So, we start with some OWA weights, assuming some of them as "ideal" (i.e. assigning the aggregation result equal 1), and then refine OWA weights already available. We proposed the aggregation result of 1 and it is a default value, but the user may also specify other values. So, the rationale behind our work is different then that of Yager's.

8. Conclusions

In this paper we proposed an interactive weights refinement method for the OWA operators, an approach that would accommodate the (individual, subjective) user's needs and intentions in the OWA-based aggregation in query relaxation for the DBMSs.

References

ANDREASEN, T., PIVERT, O. (1994) Fuzzy relational query weakening. *Datalogiske skrifter*. Roskilde Universitetscenter.

ANVARI, M., ROSE, G.F. (1987) *Fuzzy Relational Databases. The Analysis of Fuzzy Information*, J.Bezdek, ed., CRC Press, Boca Raton.

BOOKSTEIN, A. (1980) Fuzzy requests: an approach to weighted Boolean searches. *J.Am.Soc.Inform. Sci.*, **31**, 240-247.

BOSC, P., GALIBOURG, M., HAMON, G. (1988) Fuzzy querying with SQL: extensions and implementation aspects. *Fuzzy Sets and Systems*, **28**, 333-

349.

BUCKLES, B.P. and PETRY, F.G. (1982) Query language for fuzzy databases. *Management Decision Support Systems Using Fuzzy Sets and Possibility Theory*, J.Kacprzyk and R.R. Yager, eds., 241-252. Verlag TUV Rheinland, Cologne.

CASE GENERATOR (1991) for ORACLE the relational database management system, Oracle.

CHU, W.W., CHEN, Q. (1992) Neighborhood and associative query answering. *Journal of Intelligent Information Systems*, **1**, 355-382.

CUPPENS, F. AND DEMOLOMBE, R. (1991) Extending answers to neighbour entities in a cooperative answering context. *Decision Support Systems* **7**, 1-11, North-Holland.

FOXPRO 2.6 (1994) *A relational database management system for Windows 3.1*, Microsoft.

GAASTERLAND, T., GODFREY, P. AND MINKER, J. (1992) Relaxation as a platform of cooperative answering. *Journal of Intelligent Information Systems*, **1**, 293-321.

GAL, A. AND MINKER, J. (1988) Informative and cooperative answers in databases using integrity constraints. In: V. Dahl and P. Saint-Dizier (eds.) *Natural Language Understanding and Logic Programming*, 277-300, Amsterdam, North-Holland.

GUYOMARD, M. AND SIROUX J. (1989) Suggestive and corrective answers: single mechanism. In: M.M. Taylor, F. Neel and D.G. Bouwhuis (eds.), *The Structure of Multimodal Dialogue*, 361-374, Amsterdam, North-Holland.

NAITO, E., OZAWA, J., HAYASHI, I. AND WAKAMI, N. (1994) A proposal of a fuzzy connective with learning function and query networks for fuzzy retrieval systems, *Fuzziness in Database Management Systems*, P. Bosc and J. Kacprzyk eds., Physica-Verlag, Heidelberg, 345-364.

KACPRZYK, J. (1986) *Zbiory rozmyte w analizie systemowej*. PWN, Warszawa.

KACPRZYK, J. AND ZIÓŁKOWSKI, A. (1986) Database queries with fuzzy linguistic quantifiers. *IEEE Transactions on Systems, Man, Cybernetics*, **SMC-16**, 3, 474-478.

KACPRZYK, J., ZADROŻNY, S. AND ZIÓŁKOWSKI, A. (1989) FQUERY III+: A human-consistent database querying system based on fuzzy logic with linguistic quantifiers. *Information Systems*, **14**, 6, 443-453.

LARSEN, L.L., YAGER, R.R. (1993) Retrieving information by fuzzification of queries. *Journal of Intelligent Information Systems*, **2**, 421-441.

MOTRO, A. (1988) VAGUE: A user interface for relational databases that permits vague queries. *ACM Transactions on Office Information Systems*, **6**, 187-214.

TAHANI, V.A. (1977) A conceptual framework for fuzzy query processing – a step toward very intelligent database systems. *Inform. Process. Management*, **13**, 289-303.

YAGER, R.R. (1987) A note on weighted queries in information retrieval systems. *Journal of American Society of Information Sciences*, **38**, 23-24.

YAGER, R.R. (1988) On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man, Cybernetics*, **18**, 1.

YAGER, R.R. (1993A) *Families of OWA Operators.* Tech. Report MII-1301, Machine Intelligence Institute, New Rochelle, New York.

YAGER, R.R. (1993B) *On the Issue of Importance Qualifications in Fuzzy Multi-Criteria Decision Making.* Tech. Report MII-1323, Machine Intelligence Institute, New Rochelle, New York.

YAGER, R.R. (1994) *Essentials of Fuzzy Modeling.* R. Yager and D. Filev eds., Wiley, New York.

YAGER, R.R., RYBALOV, A. (1994) *Full reinforcement operators in aggregation techniques.* Tech. Report MII-1502, Machine Intelligence Institute, New Rochelle, New York.

ZADEH, L.A. (1965) Fuzzy sets. *Information and Control*, **8**.

ZEMANKOWA, M. AND KANDEL, A. (1985) Implementing imprecision in information systems. *Information Science*, **37**, 1, 2, 3 (Dec. 1985).