

On Tuning OWA Operators in a Flexible Querying Interface

Sławomir Zadrozny¹ and Janusz Kacprzyk²

¹ Warsaw School of Information Technology,
ul. Newelska 6, 01-447 Warsaw, Poland

² Systems Research Institute PAS,
ul. Newelska 6, 01-447 Warsaw, Poland

Abstract. The use of the Yager’s OWA operators within a flexible querying interface is discussed. The key issue is the adaptation of an OWA operator to the specifics of a user’s query. Some well-known approaches to the manipulation of the weights vector are reconsidered and a new one is proposed that is simple and efficient.

1 Introduction

We consider a flexible querying interface supporting an extended version of SQL such as those proposed by Kacprzyk and Zadrozny [1, 2] and Bosc et al. [3]. Basically an extension of a traditional querying language is meant here to support *linguistic terms* in queries exemplified by fuzzy values such as “young” and fuzzy relations (fuzzy comparison operators) such as “much greater than” in the following SQL query:

```
SELECT *
FROM   employees
WHERE  (age IS young) AND
       (salary IS MUCH GREATER THAN 50000 USD)      (1)
```

Another class of relevant linguistic terms are *linguistic quantifiers* such as “most”, “almost all” etc. In the extended query language advocated here they play the role of flexible aggregation operators. Kacprzyk and Ziółkowski [4], and then Kacprzyk, Ziółkowski and Zadrozny [5] proposed to use them to aggregate conditions in the WHERE clause of the SQL SELECT statement as, e.g. in

“*Most of conditions among ‘age IS young, salary IS high,...*’ are to be satisfied”

Bosc et al. (cf. e.g. [6]) proposed to use linguistic quantifiers with subqueries or against groups of rows as, e.g., in

```
SELECT deptno
FROM   employees
GROUP BY deptno
HAVING most_of (young are well-paid)      (2)
```

Whatever the role a linguistic quantifier, it has to be somehow modelled, and a user has to be provided with some means for its definition and manipulation. Here we assume that the linguistic quantifiers are originally defined and interpreted in the sense of Zadeh. Then, during a query execution they are automatically re-interpreted in terms of Yager’s OWA operators due to their high operability and intuitive appeal. Before the actual query execution the user may modify the OWA operators present in the query so as to better adjust them to his or her needs. We focus here on the guidelines that should be presented to the user in order to help him or her in an appropriate definition and manipulation of the OWA operator.

We will now briefly discuss linguistic quantifiers and Yager’s OWA operators, well-known approaches to their tuning, show how they are used and manipulated in queries, and finally present an algorithm implemented in our FQUERY for Access package [1, 2].

2 Linguistic Quantifiers and the OWA Operators

Our starting point is Zadeh’s calculus of linguistically quantified propositions [7] used to. It is a framework meant to model such expressions of natural language like

$$\text{“Most Swedes are tall”} \quad (3)$$

where “Most” is an example of a linguistic quantifier. Other examples include “almost all”, “much more than 50%” etc. We are here interested only in *relative* quantifiers such that: - their semantics refers to the proportion of elements possessing a certain property (in Example (3) it is the set of tall Swedes) among all the elements of the universe of discourse (in Example (3) it is the set of all Swedes);

and *nondecreasing* such that: - the larger such a proportion the higher the truth value of a proposition containing such a linguistic quantifier.

A linguistically quantified proposition exemplified by (3) might be formally written in a general form as

$$QxP(x) \quad (4)$$

where Q denotes a linguistic quantifier (e.g., *most*), $X = \{x\}$ is a universe of discourse (e.g., a set of Swedes), and $P(x)$ is a predicate corresponding to a certain property (e.g., of being *tall*).

The truth value of (4) is computed as follows. The relative quantifier Q is equated with a fuzzy set defined in $[0, 1]$. In particular, for a regular nondecreasing quantifier its μ_Q is assumed to be nondecreasing and normal, i.e.,

$$x \leq y \Rightarrow \mu_Q(x) \leq \mu_Q(y); \quad \mu_Q(0) = 0; \quad \mu_Q(1) = 1 \quad (5)$$

The particular $y \in [0, 1]$ correspond to proportions of elements possessing property P and $\mu_Q(y)$ assesses the degree to which a given proportion matches

the semantics of Q . For example, $Q = \text{“most”}$ might be given as:

$$\mu_Q(y) = \begin{cases} 1 & \text{for } y > 0.8 \\ 2y - 0.6 & \text{for } 0.3 \leq y \leq 0.8 \\ 0 & \text{for } y < 0.3 \end{cases} \quad (6)$$

The predicate P is modelled by an appropriate fuzzy set $P \in \mathcal{F}(X)$ ¹ characterized by its membership function μ_P .

Formally, the truth degree of (4) is computed using the following formula:

$$\text{Truth}(QxP(x)) = \mu_Q\left(\frac{1}{n} \sum_{i=1}^n \mu_P(x_i)\right) \quad (7)$$

where $r = \frac{1}{n} \sum_{i=1}^n \mu_P(x_i)$ and n is the cardinality of X .

The *ordered weighted averaging operators (OWA)* were introduced by Yager [8] and are defined as follows. Let $W \in [0, 1]^m$, $W = [w_1, \dots, w_m]$, $\sum_{i=1}^m w_i = 1$ be a weight vector. Then the OWA operator of dimension m and weight vector W is a function $O_W : [0, 1]^m \rightarrow [0, 1]$ such that:

$$O_W(a_1, \dots, a_m) = W \circ B = \sum_{i=1}^m w_i b_i \quad (8)$$

where b_i is i -th largest element among a_i 's and $B = [b_1, \dots, b_m]$; \circ denotes the scalar product.

The OWA operators generalize many widely used aggregation operators. In particular one obtains the maximum, minimum and average operators assuming $W = [1, 0, \dots, 0, 0]$, $W = [0, 0, \dots, 0, 1]$ and $W = [\frac{1}{m}, \frac{1}{m}, \dots, \frac{1}{m}, \frac{1}{m}]$, respectively. We will denote these OWA operators as O_{\max} , O_{\min} and O_{avg} , respectively.

Moreover, the OWA operators may be used to model linguistic quantifiers. Let us assume that Q is a regular non-decreasing linguistic quantifier in the sense of Zadeh (5). Then the weight vector of a corresponding OWA operator is defined due to Yager [8] as follows:

$$w_i = \mu_Q\left(\frac{i}{m}\right) - \mu_Q\left(\frac{i-1}{m}\right), \quad i = 1, \dots, m \quad (9)$$

Using this method for the linguistic quantifier in the sense of Zadeh given by (6) we obtain the OWA operator of dimension $m = 4$ with the following weight vector:

$$W = [0, 0.4, 0.5, 0.1] \quad (10)$$

Yager [8] introduced two measures characterizing the OWA operators: *ORness* and *dispersion*. The ORness of an OWA operator O_W of dimension m is denoted as $\text{ORness}(O_W)$ and computed as follows:

$$\text{ORness}(O_W) = \frac{\sum_{i=1}^m (m-i)w_i}{m-1} \quad (11)$$

¹ We will denote a family of fuzzy sets defined in X as $\mathcal{F}(X)$.

In general $\text{ORness}(O_W) \in [0,1]$ and in particular:

$$\text{ORness}(O_{\min}) = 0, \quad \text{ORness}(O_{\max}) = 1 \quad \text{ORness}(O_{\text{avg}}) = 0.5$$

It is worth noticing that the value of the ORness measure of an OWA operator O_W may be interpreted as a result of applying this OWA operator to a specific argument vector, namely:

$$\text{ORness}(O_W) = O_W \left(\frac{m-1}{m-1}, \frac{m-2}{m-1}, \dots, \frac{m-m}{m-1} \right) \quad (12)$$

The ORness is, in a sense, a measure of similarity of a given OWA operator to the “max” operator. Thus it may be used as a guideline in choosing or modifying an OWA operator. We will focus on that issue in Sections 3 and 5.

The measure of dispersion (entropy), $\text{disp}(O_W)$, for an OWA operator O_W of dimension m is defined as follows:

$$\text{disp}(O_W) = - \sum_{\substack{i=1 \\ w_i \neq 0}}^m w_i \ln(w_i) \quad (13)$$

The closer the weights w_i are one to another (and, in consequence, closer to $1/m$), the higher the measure of dispersion is. The OWA operators with higher measures of dispersion take into account more arguments a_i being aggregated. The limit cases are again O_{\min} and O_{\max} that take into account one argument only: the smallest and the largest, respectively. Their dispersion measure is the lowest possible, equal to 0. The operator O_{avg} treats all arguments equally and has the highest possible measure of dispersion equal to $\ln(m)$.

3 Definition of the OWA Operators

The main problem addressed in this paper is how to define and manipulate the OWA operators. These problems are strictly related. The former has attracted much more attention. In the literature many approaches have been proposed. They may be briefly summarized as follows.

The weight vector W of an OWA operator O_W might be determined using:

1. experimental data: suppose a set of data $\{(a_1^j, \dots, a_m^j, y^j)\}_{j=1, \dots, n}$ is available; then assuming that $y_j = O_W(a_1^j, \dots, a_m^j) \forall j$ the weights vector W best matching this assumption is sought for. Filev i Yager [9] formalized that problem as the search for the weights vector minimizing the sum of squared errors $\sum_{j=1}^n (y_j - \sum_{i=1}^m w_i b_i^j)^2$ Assuming a specific parametrized form of the weight vector $w_i = e^{\lambda_i} / \sum_{j=1}^m e^{\lambda_j}$, this optimization problem may be reduced to a constraint-free form (the previous forms requires the imposition of constraints securing that the weights are non-negative and summing up to 1.)

2. a linguistic quantifier in the sense of Zadeh: the OWA operator is meant to represent a given linguistic quantifier; thus its weights vector is computed using the formula (9).
3. a fixed value of certain characteristic features of the OWA operator: a weights vector W is sought so that to obtain this fixed value, possibly optimizing the value of another characteristic feature.

The first approach might be useful in the case of a flexible querying interface. However, it requires an extensive cooperation (interaction) with the user to collect a large amount of data. The second approach is fairly obvious and does not require additional comments. The third approach seems to be the most promising from the point of view of supporting both the definition and modification of an OWA operator.

O'Hagan [10] proposed to determine the weight vector W such that the OWA operator obtained has a fixed required value of the ORness measure and at the same time has the maximum possible value of the dispersion measure. The class of such operators is referred to as MEOWA (Maximum Entropy OWA) operators. Formally, they are determined by solving the following optimization problem:

$$disp(O_W) \mapsto max \quad \text{subject to } ORness(O_W) = \alpha, \quad \sum_i^m w_i = 1, \quad w_i \geq 0 \quad \forall i \tag{14}$$

where α is the required ORness measure value.

Filev and Yager [11] simplified this optimization problem using the Lagrange multipliers method. Then the problem boils down to finding the root of a polynomial of degree $m - 1$. Fuller and Majlender [12], assuming the same approach, proposed a simpler formulae for the weight vector W .

Fuller and Majlender considered also the variance of an OWA operator defined as:

$$var(O_W) = \sum_{i=1}^m \frac{(w_i - \frac{\sum_{i=1}^m w_i}{m})^2}{m} = \frac{1}{m} \sum_{i=1}^m w_i^2 - \frac{1}{m^2} \tag{15}$$

Then they proposed [13] a class of OWA operators analogous to MEOWA (14) where the variance instead of dispersion is maximized. Also in this case Fuller and Majlender developed analytical formulae for the weight vector W .

Filev and Yager also addressed the problem of determining the weights of an OWA operator with a required fixed value of the ORness measure. In [9] they proposed a simple approximate procedure for a certain class of OWA operators, referred to as *exponential OWA operators*.

The weights of the exponential OWA operators are defined as follows:

$$w_i = \alpha(1 - \alpha)^{i-1}, \forall i \neq m; w_m = (1 - \alpha)^{m-1} \tag{16}$$

where $\alpha \in [0, 1]$ is a parameter that together with the dimension m fully characterizes an OWA operator.

For a fixed dimension m the value of the ORness measure of such an exponential OWA operator increases together with the value of α . Knowing this

dependency it is possible to choose an appropriate value of α to obtain an OWA operator (16) having approximately a required level of the ORness measure.

Filev and Yager consider in [9] still another class of the OWA operators such that it is fairly easy to determine the weight vector W providing a required level of the ORness measure. The weights of these operators are all identical except for the first and the last one, i.e., w_1 and w_m . For such operators the ORness measure (11) may be expressed as follows [9]:

$$\text{ORness}(O_W) = 0.5 + 0.5(w_1 - w_m) \tag{17}$$

Thus in order to obtain a required ORness level, only w_1 and w_n satisfying (17) and $v_1, v_n \in [0, 1]$ have to be selected. The remaining weights are assumed to be identical, thus equal to:

$$w_i = \frac{1}{m-2}(1 - w_1 - w_m) \quad 2 \leq i \leq m - 1$$

Filev and Yager also proposed [9] a modified version of this method. Namely, the weights of an OWA operator of dimension m and of a required ORness measure value α are defined as:

$$w_i = \frac{1}{m}(1 - |\Delta|) \quad 2 \leq i \leq m - 1$$

$$\left. \begin{aligned} w_1 &= \frac{1}{m}(1 - |\Delta|) + \Delta \\ w_n &= \frac{1}{m}(1 - |\Delta|) \end{aligned} \right\} \text{if } \Delta > 0$$

$$\left. \begin{aligned} w_1 &= \frac{1}{m}(1 - |\Delta|) \\ w_n &= \frac{1}{m}(1 - |\Delta|) - \Delta \end{aligned} \right\} \text{if } \Delta \leq 0$$

where $\Delta \doteq 2(\alpha - 0.5)$.

It may be easily verified that such an OWA operator is a weighted average of the maximum (minimum) and the arithmetic average operators for $\Delta > 0$ ($\Delta \leq 0$). More precisely:

$$O_W = \Delta \max_i a_i + (1 - \Delta) \frac{\sum_{i=1}^m a_i}{m} \tag{18}$$

for $\Delta > 0$ and

$$O_W = \Omega \min_i a_i + (1 - \Omega) \frac{\sum_{i=1}^m a_i}{m} \tag{19}$$

for $\Delta \leq 0$; where $\Omega = -\Delta$.

Thus what is obtained is a class of OWA operators directly parametrized with the ORness measure value. The formulae (18)-(19) indicate, provide a clear interpretation for these operators in terms of the traditional aggregation operators of the maximum, minimum and arithmetic average.

Liu and Chen generalized the problem (14) replacing the requirement that the OWA operator O_W sought should have a specific value α of the ORness

measure with the requirement that the O_W applied to a certain fixed argument $A = (a_1, \dots, a_m)$ should yield the value α . Due to (12) the problem (14) is a special case of such a formulation for $A = (\frac{m-1}{m-1}, \dots, \frac{m-m}{m-1})$. Liu and Chen proposed the solution to this generalized problem which again boils down to determining the roots of a certain polynomial.

4 Linguistic Quantifiers in Queries

As mentioned in the Introduction, the use of the linguistic quantifiers in various clauses of the SQL query might be conceived. Here we will focus on the way they are used in our FQUERY for Access package [1, 2], i.e., as operators aggregating conditions in the WHERE clause. Basically, the linguistic quantifiers are here defined according to Zadeh's calculus of linguistically quantified propositions, briefly recalled in the previous section. Such a representation has some advantages and primarily an easy scalability to a varying number of conditions to be aggregated. For example, having the linguistic quantifier "most" defined by (6) one may use it to interpret such a condition as:

$$\text{"Most of the predicates } \{P_i\}_{i=1, \dots, n} \text{ are satisfied"} \quad (20)$$

for any number, n , of predicates. In the case when the linguistic quantifier "most" is explicitly represented by an OWA operator (e.g., such as (10)) of dimension m , it is directly applicable for the interpretation of (20) only for $n = m$. Thus we would have to define a separate OWA operator modelling the quantifier "most" for all conceivable numbers n of conditions to be aggregated in an expression of (20) type.

On the other hand, the OWA operators also have some advantages as a linguistic quantifier modelling tool. They offer a fine-grained control over the behavior of the modelled aggregation operator and provide for a more straightforward representation of classical operators. Thus in FQUERY for Access the linguistic quantifiers are firstly defined in the sense of Zadeh and later, during the query execution, they are interpreted and manipulated in terms of OWA operators.

FQUERY for Access maintains a dictionary of various linguistic terms defined by the user and available for the use in the queries. Among them are linguistic quantifiers in the sense of Zadeh. Each quantifier is identified with a piecewise-linear membership function and assigned a name. When inserting a quantifier into a query the user just picks up its name from the list of quantifiers available in the dictionary.

The linguistic quantifiers are used in a query to aggregate conditions like in (20). In case of such explicitly used quantifiers the user indicates if the original Zadeh interpretation should be applied or if the quantifier should be treated as the OWA operator defined by (9). Additionally, the system implicitly interpretes the AND and OR connectives used in the query as the minimum and maximum operators, respectively, that in turn are represented by the O_{min} and O_{max} OWA operators. It is assumed that the query condition is in the disjunctive normal form as it is usually the case of queries created using the Microsoft

Access querying interface. In order to easily identify the operators in a query it is assumed that the conjuncts of such a disjunctive normal form are ordered and numbered.

After starting the query execution the user has still an opportunity to adjust the vector of OWA operator weights to better match his or her understanding of the aggregation to be applied to the conditions in (20). Both explicitly and implicitly used OWA operators may be modified, and the form shown in Fig. 1 makes it possible

Advanced Query Options

Select OWA operator: 1

Weights	
1	0
2	0
3	0
4	1

Dispersion: 0.0000 <= 1.3863

ORness: 0.0000

Buttons: AND, OR, ORness, Reset, Reset all, OK, Cancel

Fig. 1. FQUERY for Access form making possible modification of the OWA operators employed in a query

The list of OWA operators appearing in the executed query is shown in the upper left corner of this form. It comprises:

- all OWA operators explicitly used by the user in the query, Each operator is assigned a number indicating a conjunct number to which it applies, or the label “Global” in case of an operator concerning the whole query, i.e., corresponding to the disjunction of the conjuncts. For each OWA operator the name of the corresponding linguistic quantifier in the sense of Zadeh is also shown next to the conjunct number (this might be seen in Fig. 2).
- all implicit OWA operators, automatically inserted by FQUERY for Access and including:
 - a global one, corresponding to the disjunction connective - if a global linguistic quantifier has not been explicitly used by the user.
 - one OWA operator (O_{min} in particular) for each conjunct comprising more than one condition provided an explicit linguistic quantifier has not been used.

An implicit OWA operator is assigned a number of the conjunct it applies to or the label “Global” – similarly to the explicit operators discussed above; however there is no name associated with it (cf. the column “Name” in Fig. 2).

Fig. 2. FQUERY for Access form making possible modification of the OWA operators employed in a query with the list of the operators visible

Figure 1 corresponds to a query without explicitly used OWA operators and just one conjunct comprising four simple conditions. Due to that, on the list there is only one automatically inserted OWA operator taking place of the AND connective.

Figure 2 corresponds to a more complex query whose condition might be informally expressed in a way similar to (20) as follows:

”All (O_{min}) of the predicates:
 $\{$
Most of the predicates $\{(\text{price} < 100000), (\text{bedrooms} = 3)\}$
 are satisfied,
Any (O_{max}) of the predicates $\{(\text{bedrooms} = 2), (\text{bathrooms} = 3)\}$
 is satisfied
 $\}$
 are satisfied”

where **price**, **bedrooms** and **bathrooms** correspond to attributes of a hypothetical real-estate database, characterizing a property (house) in terms of its price, the number of bedrooms and the number of bathrooms, respectively. Here the list contains two explicit operators. The first applies to the second conjunct and is the OWA version of the linguistic quantifier defined by the user under the name “Max Fuzzy” which is meant as a counterpart of the “max” operator (OR connective). The second operator plays the role of a global quantifier and is the OWA version of the linguistic quantifier defined by the user under the name “Min Fuzzy” which is meant as a counterpart of the “min” operator (AND connective). There is no implicit operator for the first conjunct as the user explicitly used for it a linguistic quantifier named “Most”.

5 Tuning the OWA Operators in a Flexible Querying Interface

In the previous section we have briefly discussed how linguistic quantifiers are used in FQUERY for Access and what the role of the OWA operators in this respect is. Here we will focus on how these operators might be modified (tuned) before the query is executed.

In order to modify an OWA operator the user has to select it on the list shown in Fig. 1. Then its weight vector is displayed below in a control labelled “Weights” and it is available for modifications in many ways:

- each element of the vector may be modified individually; a chosen element has to be selected and then increased or decreased using the buttons labelled “+” and “-”, respectively. The user is responsible for changing other weights too so that they sum up to 1; otherwise FQUERY for Access will not go to the next stage of query execution;
- pressing the button labelled “AND” (or “OR”) sets the weights in such a way that the O_{min} (or O_{max}) operator are obtained;
- pressing the button “ORness” changes the weight vector in such a way that the ORness measure value of the resulting OWA operator is equal to the number entered in the formant labelled “ORness” located at the bottom of the form (cf. Fig. 1);
- pressing the button labelled “Reset” (or “Reset all”) restores the weight vector of the selected operator (or of all listed OWA operators) to their original values.

The value of the ORness measure does not uniquely determine the OWA operator. Thus, any of the approaches to the tuning of the OWA operators discussed in the previous section are here applicable to set the weight vector. The difference of the setting considered here is that we start with a certain OWA operator and want to tune it in order to increase its ORness. However, an additional reasonable requirement might be such that the user wants to keep the changes as limited as possible or to preserve the consistency of the change in some other sense. For example, the user may look for an OWA operator more similar in its behaviour to the maximum operator but at the same time still “similar to the original OWA operator” being modified. In FQUERY for Access we have implemented a simple approximate algorithm that for an OWA operator O_W yields a new OWA operator O_V having a smaller/larger value of the ORness measure (as required by the user) and additionally preserving the consistency of the change in such a sense that:

$$O_V(a_1, \dots, a_m) \geq O_W(a_1, \dots, a_m) \quad \forall (a_1, \dots, a_m) \quad (21)$$

when an increase of the ORness is required, and

$$O_V(a_1, \dots, a_m) \leq O_W(a_1, \dots, a_m) \quad \forall (a_1, \dots, a_m) \quad (22)$$

when a decrease of the ORness is required. This seems to be consistent with expectations of the user requiring an increase/decrease of the ORness while is not in general guaranteed, i.e., there exist such pairs of the OWA operators O_W and O_V (cf., e.g., [14]) that: $\text{ORness}(O_W) > \text{ORness}(O_V)$ and at the same time there exists $A = (a_1, \dots, a_m)$ such that $O_W(A) < O_V(A)$.

The algorithm used in FQUERY for Access consists of the following steps (here in the case when the increase of the ORness is required; in the opposite case the algorithm is analogous). Let $W = [w_1, \dots, w_n]$ denote the weight vector of the OWA operator O_W , and z^0 denote the required increased value of the ORness measure ($z^0 \in (\text{ORness}(O_W), 1)$).

Then:

- Step 1 $\Delta z := z^0 - \text{ORness}(O_W)$
 - Step 2 $k := \max_i \{i : w_i > 0\}$
 $x := 2(n - 1)\Delta z/k$
 - Step 3 if $x > w_k$ then $x := w_k$
 - Step 4 $w_k := w_k - x$
 $w_i := w_i + x/(k - 1) \quad \forall i \in [1, k - 1]$
 - Step 5 if $\text{ORness}(O_W) < z^0$ go to Step 1
- STOP

The main idea is to reduce the element of the weight vector with the highest index by such a value x (thus highly contributing to the “ANDness” of the operator) that, when equally distributed among the remaining elements of the weight vector, yields an OWA operator with the ORness measure value equal or very close to the required one.

In Step 1 the required increase (Δ) of the ORness is computed. In Step 2 a positive element of the weight vector with the highest index is selected (of index k) and it is computed how much it should be reduced (x) to obtain a required value of the ORness measure. In Step 3 it is verified if the required reduction of the k -th element w_k of the weight vector (i.e., x) does not exceed its current value: if it is the case then the x is set equal to w_k (another iteration of the algorithm will be required). In Step 4 the weight vector is modified: w_k is reduced by x and the remaining elements of the vector W are increased by the same part of x , i.e., $x/(k - 1)$. In Step 5 it is verified if the ORness measure value of the modified operator reaches the required level: if it is not the case, the next iteration starts from Step 1.

It is obvious that by construction of the algorithm the modified vector W verifies conditions required for the OWA operators, i.e., $W \in [0, 1]^m$ and $\sum_{i=1}^m w_i = 1$ as well as has the required ORness measure value and satisfies (21), where W and V denote the original and modified weight vectors, respectively.

6 Concluding Remarks

We reconsidered the role of fuzzy linguistic quantifiers in flexible database queries, notably showing that Yager’s OWA operators provide a flexible and

efficient means of a fuzzy-quantifier-based aggregation. We discussed the tuning (selection of weights) of the OWA operators, and proposed an algorithm that is effective and efficient in the context of our FQUERY for Access package [1, 2].

References

1. Kacprzyk J., Zadrozny: The paradigm of computing with words in intelligent database querying. In Zadeh L.A., Kacprzyk J., eds.: *Computing with Words in Information/Intelligent Systems*. Part 1. Foundations. Part 2. Applications. Springer-Verlag, Heidelberg and New York (1999) 382–398
2. Kacprzyk J., Zadrozny S.: Computing with words in intelligent database querying: standalone and internet-based applications. *Information Sciences* (134) (2001) 71–109
3. Bosc P., Pivert O.: SQLf: A relational database language for fuzzy querying. *IEEE Transactions on Fuzzy Systems* **3**(1) (1995) 1–17
4. Kacprzyk J., Ziolkowski A.: Database queries with fuzzy linguistic quantifiers. *IEEE Transactions on System, Man and Cybernetics (SMC-16)* (1986) 474–479
5. Kacprzyk J., Zadrozny S., Ziolkowski A.: FQUERY III+: a “human consistent” database querying system based on fuzzy logic with linguistic quantifiers. *Information Systems* (6) (1989) 443–453
6. Bosc P., Prade H.: An introduction to the fuzzy set and possibility theory-based treatment of flexible queries and uncertain or imprecise databases. In: *Uncertainty Management in Information Systems*. Kluwer Academic Publishers, Boston (1996) 285–324
7. Zadeh L.A.: A computational approach to fuzzy quantifiers in natural languages. *Computers and Mathematics with Applications* **9** (1983) 149–184
8. Yager R.R.: On ordered weighted averaging aggregation operators in multi-criteria decision making. *IEEE Transactions on Systems, Man and Cybernetics* **18** (1988) 183–190
9. Filev D., Yager R.R.: On the issue of obtaining OWA operator weights. *Fuzzy Sets and Systems* **94** (1998) 157–169
10. O’Hagan M.: Aggregating template or rule antecedents in real-time expert systems with fuzzy set logic. In: *Proceedings of the IEEE Asilomar Conference on Signals, Systems, Computers*, Pacific Grove, USA (1988) 81–89
11. Filev D., Yager R.R.: Analytic properties of maximum entropy OWA operators. *Information Sciences* **85** (1995) 11–27
12. Fuller R., Majlender P.: An analytic approach for obtaining maximal entropy OWA operator weights. *Fuzzy Sets and Systems* **124**(1) (2001) 53–57
13. Fuller R., Majlender P.: On obtaining minimal variability OWA operator weights. *Fuzzy Sets and Systems* **136**(2) (2003) 203–215
14. Liu X., Chen L.: On the properties of parametric geometric owa operator. *International Journal of Approximate Reasoning* **35** (2004) 163–178