

Fuzzy Ranking of Approximate Answers

Berthier A. Ribeiro-Neto
Computer Science Department
Federal University of Minas Gerais
31270-010, Belo Horizonte, Minas Gerais
Brazil

Richard Muntz
Computer Science Department
University of California, Los Angeles
Los Angeles, California 90024
U.S.A.

Abstract

We propose an extended fuzzy algebra for ranking approximate answers relative to a multiple criteria query request. The algebra is derived from a set of basic axioms which are intuitive and can be easily grasped by the user. Further, the user has the option of stating preference relationships in qualitative terms (e.g., greater than, lesser than, etc.) and use our algebra to order the set of approximate answers according to these preferences. This last point is important because the user might find it difficult to express his preferences in numeric terms.

1 Introduction

Many applications simply cannot perform well without some notion of approximation provided by the information system. Cuppens and Demolombe [2] mention decision support systems and advice giving systems. Fuhr [3] mentions engineering applications, materials data systems, and business decision making systems. Rabitti and Savino [7] point to image databases. In such applications, the user usually poses

a query composed of multiple approximate criteria requesting the information system to return the *most relevant* answers.

To determine the most relevant answers the information system usually generates a ranking (i.e., ordering) of the answer set. The rank (i.e., order) of a given approximate answer t_k is in general a function f of the degrees with which t_k approximates the various query criteria individually. This problem can be stated more formally as follows. Consider a user query Q composed of conjunctions, disjunctions, and negation of the criteria A, B, \dots, P and let t_k be an approximate answer. Given the degrees a_1, a_2, \dots, a_n (where $0 \leq a_i \leq 1$) to which t_k approximates each of the query criteria individually (which, according to [12], are called *degrees of compatibility*), the information system must specify a function $f(a_1, a_2, \dots, a_n)$ as the rank of t_k . The definition of f is a non-trivial issue and most approaches consist of heuristics which can not be easily grasped by the user or which are difficult to justify formally.

Fuhr [3] uses a *relevance description* function to model approximation neighborhoods (i.e., data metrics). However, his approach does *not* allow Boolean operators in the query formulation because a query involving *conjuncts* and *disjuncts* would lead to complex probabilistic formulas with parameters that could not be accurately estimated in most cases. Rabitti and Savino [7] propose to sum up the contributions of an object to the different portions of an *or* condition. An *and* operation is satisfied or not satisfied depending on whether the object approximates all conjunctive conditions better than a minimum similarity threshold. Therefore, an object O_1 which barely satisfies the thresholds and another object O_2 which matches all conditions in the *and* operation exactly would receive the same ranking. This solution is inappropriate for large systems because many approximate answers would receive the same rank. Motro [6] proposes to combine metric distances (which can be interpreted as inverse functions of degrees of compatibility) in an *and* operation by a weighted square sum of the distances for each criteria. However, the meaning of computing the overall distance for an *and* condition by a weighted sum is not clear.

Classic fuzzy theory [12] adopts the *min* operator to compute the degree of membership (i.e., degree of compatibility) in conjunctive sets and the *max* operator to compute the degree of membership in disjunctive sets. The main problem with the *min* and *max* functions is that the first is a purely conjunctive operator while the sec-

and is a purely disjunctive operator. As a consequence, the influence of second order factors in the rank of an approximate answer is not considered. For instance, let $Q = (A + B)C$ be a user query and let t_1 and t_2 be two approximate answers. Assume that the degrees of compatibility of t_1 and t_2 with the query criteria are as follows: $a_{t_1} = 1$, $a_{t_2} = 0.72$, $b_{t_1} = 0.6$, $b_{t_2} = 0.5$, $c_{t_1} = 0.35$, and $c_{t_2} = 0.35$. Then, $f(a_{t_1}, b_{t_1}, c_{t_1}) = \min(\max(1, 0.6), 0.35) = 0.35$ and $f(a_{t_2}, b_{t_2}, c_{t_2}) = \min(\max(0.72, 0.5), 0.35) = 0.35$. Thus, the rankings of the answers t_1 and t_2 coincide because they both provide the same approximation to the C criteria. However, t_1 provides a better approximation to the other two query criteria A and B . Classic fuzzy theory cannot take advantage of this additional information because the operators \min and \max focus only on the most determinant degree of compatibility. Our idea is to substitute \min and \max with a pair of more generic operators which enrich the operations of conjunction and disjunction with complementary information.

The work in [11] is closer to ours. It proposes a family of *orand* fuzzy operators (called OWA operators) for deciding preferences relative to a multiple criteria query request. The operators basically compute weighted averages as the final rank associated to an approximate answer. The weighting scheme associates coefficients to ordered positions (instead of attributes) and allows specification of pure *and* operations, pure *or* operations, and operations whose semantics lies in between. This is accomplished by proper adjustments of the weighting coefficients. The main concern we have with such approach is that it might be difficult (if not impossible) for the user to mentally recompute the weighted average associated to an interesting answer in order to gain insight into the final ranking. In fact, frequently the user is not familiar with expressing his beliefs in numeric terms [9] and might find it rather difficult to reason quantitatively. However, people are at ease when expressing preference relationships qualitatively and when using these preferences to support judgemental decisions. In fact, the work in [9] deals solely with the problem of assigning numeric values to preferences stated qualitatively by a user. Our work here is an attempt to provide an alternative fuzzy ranking which is easier to be reproduced by a user who reasons in qualitative terms.

We propose an extended fuzzy algebra called *F-G algebra* which, instead of attempting to combine degrees of compatibility to a scalar value, orders them in a list according to the query semantics. These

lists are then compared to decide preference in the final ranking. The generation of these lists of degrees of compatibility is based on a set of fundamental axioms (e.g., commutativity, associativity, etc) which are intuitively appealing and which can be easily grasped by the user.

The \mathcal{F} - \mathcal{G} algebra was first introduced in [8]. In this work, we extend the preliminary results in [8] in three directions. First, we exhaust the discussion of purely conjunctive (which in the following we call \mathcal{F} - \mathcal{F}) and purely disjunctive (which in the following we call \mathcal{G} - \mathcal{G}) queries. Second, we completely redefine the semantics of our fuzzy ranking relative to queries which mix conjunctive and disjunctive operations to preserve desired properties. Third, this redefinition leads to some interesting new properties which allow the establishment of a clear relationship between the conjunctive (i.e., \mathcal{F}) and disjunctive (i.e., \mathcal{G}) operators.

The rest of the paper is organized as follows. Section 2 introduces the terminology used in the paper. Section 3 discusses how to rank the answers relative to purely conjunctive queries while section 4 deals with purely disjunctive queries. Section 5 discusses the ranking for expressions which mix conjunctive and disjunctive subexpressions. Additional properties of our algebra are presented in section 6 followed by our conclusions.

2 Basic Definitions

In this section we introduce basic definitions and terminology used throughout the paper. We start with a classic theorem regarding the *min* and *max* operators of fuzzy theory.

Lemma 1 *Let F and G be two binary operations on degrees of compatibility (e.g., a, b, c) in the range $[0,1]$. Then, $F = \min$ and $G = \max$ are the unique functions of two operands that satisfy the following conditions:*

- 1) *associativity: $F(a, F(b, c)) = F(F(a, b), c)$,
 $G(a, G(b, c)) = G(G(a, b), c)$;*
- 2) *commutativity: $F(a, b) = F(b, a)$, $G(a, b) = G(b, a)$;*
- 3) *nondecreasing: $F(a, b) \leq F(c, d)$, $G(a, b) \leq G(c, d)$, if $a \leq c$
and $b \leq d$;*
- 4) *$F(a, a) < F(b, b)$, $G(a, a) < G(b, b)$, if $a < b$;*
- 5) *$F(a, b) = 1$ if $a = 1$ and $b = 1$, $G(a, b) = 0$ if $a = 0$ and $b = 0$;*
- 6) *$F(a, b) \leq \min(a, b)$, $G(a, b) \geq \max(a, b)$;*

$$7) \text{ distributivity: } F(a, G(b, c)) = G(F(a, b), F(a, c)), \\ G(a, F(b, c)) = F(G(a, b), G(a, c)).$$

Proof Refer to [1] and [4]. \square

The properties in lemma 1 are intuitively appealing and desirable in any user interface. Particularly, properties 1 to 6 are usually expected. Thus, our approach attempts to enforce these properties to the furthest extent.

Our purpose is to extend the functions F and G to consider the impact of second order operands while preserving as many of the properties in lemma 1 as possible. Since min and max are the unique functions which preserve all seven properties, our extensions to F and G have to violate at least one of those properties.

The alternatives in the literature for the functions F and G usually attempt to combine the various grades of membership into a single quantitative value [5, 10, 11]. For this purpose a function f is always applied to the degrees of compatibility a, b, \dots, p involved in the query and the output of f used as the final quantitative judgement. Our concern is that it might be difficult (if not impossible) for the user to mentally recompute the function f (for those answers in which he is most interested in) in order to gain insight into the final ranking.

Instead of combining degrees of compatibility into a numeric value, we keep all of these degrees in an ordered list which we call a *peerage* (i.e., *list of peers*). A distinct peerage is generated for each element in the answer set and is interpreted as the *rank* of that element. By comparing peerages we are able to decide an ordering (i.e., final ranking) for the answer set.

Definition 1 *Given the degrees of compatibility a_1, a_2, \dots, a_n , there are 2^n possible orderings for them. Each of these orderings is called a peerage (i.e., list of peers) and is represented by $[a_{k_1}; a_{k_2}; \dots; a_{k_n}]$, where $1 \leq k_i \leq n$ and $i \neq j \Rightarrow k_i \neq k_j$.*

For instance, the degrees of compatibility a and b define two peerages: $[a; b]$ and $[b; a]$. A single grade of membership c defines the unique peerage $[c]$. Since the ranking is now based on peerages, we modify the functions F and G to return *peerages* instead of a combined numeric value.

Definition 2 *Let \mathcal{F} and \mathcal{G} denote our new versions of F and G . The new functions \mathcal{F} and \mathcal{G} take a set of degrees of compatibility as input and return a peerage composed of those degrees of compatibility as output.*

Our idea is to write the user query as an expression composed by the new functions \mathcal{F} (which is interpreted as a conjunctive operator) and \mathcal{G} (which is interpreted as a disjunctive operator) and to have these functions generate peerages for every approximate answer in the answer set. These peerages can then be compared to decide preference among the various tuples. Let us look into a simple example.

Example 1 Consider the user query $Q = AB$. Let t_1 and t_2 be two approximate answers and let $a_1, b_1, a_2,$ and b_2 be the degrees of compatibility with which t_1 and t_2 approximate the query criteria A and B , respectively. Further, assume that $a_1 < b_1, a_2 < b_2,$ and $a_1 > a_2$.

The relationship between two degrees of compatibility in example 1 is interpreted as the user's preference and can be stated qualitatively. For instance, the relationship $a_1 > a_2$ could be expressed qualitatively as $(A, t_1) \succ (A, t_2)$ meaning that the user prefers t_1 to t_2 as an approximation for the individual criteria A . The query in this example is a conjunction of A and B which is interpreted as the operation $\mathcal{F}(A, B)$. As we discuss in section 3, the operation $\mathcal{F}(A, B)$ generates the peerages $[a_1; b_1]$ and $[a_2; b_2]$ corresponding to the tuples t_1 and t_2 respectively. These two peerages are then used to decide whether t_1 or t_2 is preferable as an approximate answer to the query Q . This requires comparing the two peerages.

We intentionally avoid combining the peerage components into a single number for comparison purposes (because, as already discussed, the user could find it difficult to understand this single number). Instead, we compare two peerages by comparing their degrees of compatibility from left to right as follows.

Definition 3 Let $P_1 = [a_1; a_2; \dots, a_n]$ and $P_2 = [b_1; b_2; \dots, b_m]$ be two peerages. Let j be the smallest index for which $a_j \neq b_j$. Then,

$$\begin{aligned} j > n &\Rightarrow P_1 \geq P_2 \\ j > m &\Rightarrow P_1 \leq P_2 \\ a_j > b_j &\Rightarrow P_1 > P_2 \\ a_j < b_j &\Rightarrow P_1 < P_2. \end{aligned}$$

For instance, consider the three peerages: $P_1 = [1; 0.5; 0.3]$, $P_2 = [1; 0.6]$, and $P_3 = [1]$. Then, $P_1 < P_2$, $P_1 \leq P_3$, and $P_2 \leq P_3$. Consider now a fourth peerage $P_4 = [1; 0.5; 0.3]$. Clearly, $P_1 \leq P_4$, $P_1 \geq P_4$, and it has to be $P_1 = P_4$. This approach is simple and can be easily

reproduced mentally by the user even if he does not have his preferences stated in quantitative terms. In fact, consider the query Q in example 1. The user could have stated his preferences qualitatively as follows: $(A, t_1) \prec (B, t_1)$, $(A, t_2) \prec (B, t_2)$, and $(A, t_1) \succ (A, t_2)$. Given these preferences, the user has enough information to determine that the peerages for t_1 and t_2 are $[a_1; b_1]$ and $[a_2; b_2]$ respectively. Further, since $(A, t_1) \succ (A, t_2)$, he is able to easily conclude that the system should indicate t_1 as preferable to t_2 as an approximate answer for the query Q . Obviously, there is no intent that the user recompute peerages for all approximate answers. The idea is to allow him to easily recompute peerages (by reasoning qualitatively) whenever he finds it necessary (for instance, for those answers which seem most interesting to him).

Before proceeding we distinguish the expressions in a query as belonging to one of three types.

Definition 4 *An expression that contains only references to the function \mathcal{F} is called an \mathcal{F} - \mathcal{F} expression. An expression that contains only references to the function \mathcal{G} is called a \mathcal{G} - \mathcal{G} expression. Expressions that mix references to \mathcal{F} with references to \mathcal{G} are called \mathcal{F} - \mathcal{G} expressions.*

3 \mathcal{F} - \mathcal{F} Expressions

In this section we define the new function \mathcal{F} . It takes a pair of peerages as arguments and returns a resultant peerage which includes second order operands. The form of \mathcal{F} is defined by enforcing properties of lemma 1. We first enforce property (6).

Axiom 1 *We assume that the function \mathcal{F} satisfies property (6) of lemma 1.*

Let us consider first the case in which the operand peerages are each composed of a single degree of compatibility i.e., $\mathcal{F}([a], [b])$. The axiom 1 requires that $\mathcal{F}([a], [b]) \leq \min([a], [b])$ which motivates defining \mathcal{F} as follows.

Definition 5

$$\mathcal{F}([a], [b]) = \begin{cases} [a; b] & \text{if } [a] < [b] \\ [b; a] & \text{otherwise} \end{cases} \quad (1)$$

The new function \mathcal{F} preserves both degrees of compatibility (ordered by their increasing values) in the body of the computed peerage. The smallest degree of compatibility is the prevailing decision factor (as in the classical fuzzy theory) when comparing two or more peerages. However, opposed to the classical theory, the second order factor might have an impact in the ordering of the retrieved answers.

Let us now extend the definition of \mathcal{F} to more complex peerages. Such extension requires enforcing additional properties of lemma 1 as follows.

Axiom 2 *We assume that the function \mathcal{F} satisfies properties (1) and (2) of lemma 1 i.e., we assume that \mathcal{F} is both associative and commutative.*

To simplify notation we refer to the trivial peerages $[a]$, $[b]$, $[c]$, \dots , $[p]$ as a , b , c , \dots , p , respectively.

Lemma 2 *Let a , b , and c be degrees of compatibility such that $a \leq b \leq c$ and let $\mathcal{F}(a, \mathcal{F}(b, c))$ return a peerage composed of a , b , and c . Then,*

$$\mathcal{F}(a, \mathcal{F}(b, c)) = [a; b; c] \quad (2)$$

Proof From axiom 2, we can write: $\mathcal{F}(a, \mathcal{F}(b, c)) = \mathcal{F}(\mathcal{F}(a, b), c)$. Due to $\mathcal{F}(a, b)$, a must precede b . Due to $\mathcal{F}(b, c)$, b must precede c . Therefore, the only ordering which is possible for the resultant peerage is $[a; b; c]$. \square

It is immediate that $\mathcal{F}(a, \mathcal{F}(b, c)) = \mathcal{F}(\mathcal{F}(a, c), b)$. Further, commutativity must hold due to property (2). This motivates adopting a more concise notation as follows.

Definition 6 $\mathcal{F}(a, b, c)$ is used to denote any of the following six \mathcal{F} - \mathcal{F} expressions: $\mathcal{F}(a, \mathcal{F}(b, c))$, $\mathcal{F}(\mathcal{F}(b, c), a)$, $\mathcal{F}(\mathcal{F}(a, b), c)$, $\mathcal{F}(c, \mathcal{F}(a, b))$, $\mathcal{F}(\mathcal{F}(a, c), b)$, and $\mathcal{F}(b, \mathcal{F}(a, c))$. These are all the \mathcal{F} - \mathcal{F} expressions which can be written with single occurrences of a , b , and c and are referred to as basic \mathcal{F} - \mathcal{F} expressions.

Lemma 3 *Let a , b , \dots , p , q be degrees of compatibility such that $a \leq b \leq \dots \leq p \leq q$. Then,*

$$\mathcal{F}(a, \mathcal{F}(b, \dots, \mathcal{F}(p, q))) = [a; b; \dots; p; q] \quad (3)$$

Proof Due to the associativity and commutativity of \mathcal{F} we can rearrange the above expression such that $\mathcal{F}(a, b)$ appears as the innermost reference to \mathcal{F} . This implies that a must precede b . Further, the expression can be rewritten such that the pairs (b, c) , (c, d) , (d, e) , \dots , (p, q) also appear in the innermost reference to \mathcal{F} . Therefore, the only possible ordering for the resultant peerage is $[a; b; \dots; p; q]$. \square

Definition 7 *The notation $F(a, b, \dots, p, q)$ is used to denote any of basic \mathcal{F} - \mathcal{F} expression which can be written with a, b, \dots, p, q because all of them return the same peerage as result.*

It can be trivially demonstrated that our new \mathcal{F} function also satisfies properties (3), (4), and (5) of lemma 1. Thus, \mathcal{F} verifies properties (1) to (6) of lemma 1.

Lemma 3 defined the semantics of basic \mathcal{F} - \mathcal{F} expressions in which one of the operands is always a trivial peerage (i.e., composed of a single degree of compatibility). This semantics is sufficient for computing the resultant peerage for any \mathcal{F} - \mathcal{F} expression as stated in the following lemma.

Lemma 4 *Let \mathcal{E}_{f_1} and \mathcal{E}_{f_2} be two basic \mathcal{F} - \mathcal{F} expressions. Without loss of generality, assume that $\mathcal{E}_{f_2} = \mathcal{F}(a, \mathcal{F}(b, \dots, \mathcal{F}(p, q)))$. Then,*

$$\mathcal{F}(\mathcal{E}_{f_1}, \mathcal{E}_{f_2}) = \mathcal{F}(a, \mathcal{F}(b, \dots, \mathcal{F}(p, \mathcal{F}(q, \mathcal{E}_{f_1})))) \quad (4)$$

which is computed as a basic \mathcal{F} - \mathcal{F} expression.

Proof Just enforce the associative property. \square

4 $\mathcal{G} - \mathcal{G}$ Expressions

Our new function \mathcal{G} is defined analogously to the function \mathcal{F} above. The fundamental difference is that the function \mathcal{G} values more the larger grades of membership instead of the smaller ones.

Axiom 3 *We assume that the function \mathcal{G} satisfies property (1), (2), and (6) of lemma 1.*

Thus,

Definition 8

$$\mathcal{G}([a], [b]) = \begin{cases} [a; b] & \text{if } [a] > [b] \\ [b; a] & \text{otherwise} \end{cases} \quad (5)$$

Also,

Definition 9 *The notation $G(a, b, \dots, p, q)$ is used to denote any of the basic \mathcal{G} - \mathcal{G} expressions which can be written with a, b, \dots, p, q because all of them return the same peerage as result.*

Further,

Lemma 5 *Let a, b, \dots, p, q be grades of membership such that $a \leq b \leq \dots \leq p \leq q$ and let $G(a, b, \dots, p, q)$ return a peerage composed of a, b, \dots, p, q . Then,*

$$\mathcal{G}(a, b, \dots, p, q) = [q; p; \dots; b; a] \quad (6)$$

Proof Analogous to the proof for lemma 3. \square

It can be trivially demonstrated that our new \mathcal{G} function also satisfies properties (3), (4), and (5) of lemma 1. Thus, \mathcal{G} verifies properties (1) to (6) of lemma 1.

Lemma 5 defined the semantics of basic \mathcal{G} - \mathcal{G} expressions in which one of the operands is always a trivial peerage (i.e., composed of a single degree of compatibility). This semantics is sufficient for computing the resultant peerage for any \mathcal{G} - \mathcal{G} expression because such expression can be transformed into a basic \mathcal{G} - \mathcal{G} expression by enforcing associativity.

5 F-G Expressions

In this section we discuss the generation of peerages for expressions that mix references to the functions \mathcal{F} and \mathcal{G} . The natural procedure would be to enforce property (7) of lemma 1 i.e., to attempt to make \mathcal{F} and \mathcal{G} distributive. Unfortunately, the functions \mathcal{F} and \mathcal{G} are not distributive as we now prove.

Lemma 6 *The new functions \mathcal{F} and \mathcal{G} are not distributive.*

Proof The proof is by contradiction. Recall that the *min* and *max* functions are the only functions which satisfy all seven properties of

lemma 1. Further, recall that our \mathcal{F} and \mathcal{G} functions preserve properties 1 to 6 of lemma 1. Define a function \mathcal{P} which takes a generic peerage as input and generates a corresponding real number (i.e., $\mathcal{P} : [a_1; a_2; \dots; a_n] \rightarrow R$) in the range $[0, 1]$ as follows:

$$\mathcal{P}([a_1; a_2; \dots; a_n]) = \sum_{i=1}^n 2^{-i} \times a_i \quad (7)$$

It can be easily verified that there is a *natural isomorphism* from $[a_1; a_2; \dots; a_n]$ to $\mathcal{P}([a_1; a_2; \dots; a_n])$. Consider the following F-G expression: $\mathcal{F}(a, \mathcal{G}(b, c))$. If distributivity were to hold then $\mathcal{F}(a, \mathcal{G}(b, c)) = \mathcal{G}(\mathcal{F}(a, b), \mathcal{F}(a, c))$ i.e., these two expressions would generate the same peerage as output. But then, it also would be $\mathcal{P}(\mathcal{F}(a, \mathcal{G}(b, c))) = \mathcal{P}(\mathcal{G}(\mathcal{F}(a, b), \mathcal{F}(a, c)))$ which contradicts lemma 1 because \mathcal{F} is not the *min* function and \mathcal{G} is not the *max* function. \square

Since distributivity does not hold, we have to rely on properties (1) to (6) for defining the semantics of an F-G expression.

Lemma 7 *Let a , b , and c be three grades of membership such that $c \geq b$. Then, if property (6) is to hold, it must be*

$$\mathcal{F}(a, \mathcal{G}(b, c)) = \begin{cases} [a; c; b] & \text{if } a < c \\ [c; b; a] & \text{if } a \geq c \end{cases} \quad (8)$$

$$\mathcal{G}(a, \mathcal{F}(b, c)) = \begin{cases} [a; b; c] & \text{if } a \geq b \\ [b; c; a] & \text{if } a < b \end{cases} \quad (9)$$

Proof Consider first the case of $\mathcal{F}(a, \mathcal{G}(b, c))$. The expression $\mathcal{G}(b, c)$ requires that c appears before b in the final peerage (we assume that the ordering dictated by an inner operator cannot be affected by an \mathcal{F} or \mathcal{G} operation) which restricts its form to one of $\{[a; c; b], [c; a; b], [c; b; a]\}$. If $a < c$ then the final peerage must be $[a; c; b]$ due to property (6). If $a \geq c$ then the final peerage must be $[c; b; a]$. The alternative would be $[c; a; b]$. However, this is the resultant peerage for the expression $\mathcal{F}(c, \mathcal{G}(a, b))$ (due to property 6) and thus cannot be the peerage for $\mathcal{F}(a, \mathcal{G}(b, c))$ because the semantics of these two expressions is not the same. The argumentation for the case of $\mathcal{G}(a, \mathcal{F}(b, c))$ is analogous. \square

The \mathcal{F} - \mathcal{G} expression $\mathcal{F}(a, \mathcal{G}(b, c))$ can be interpreted as $\mathcal{F}([a], [c; b])$ i.e., as the application of the operator \mathcal{F} to two operand peerages. Due to lemma 7, the \mathcal{F} operator can no longer sort the degrees of compatibility in the operand peerages (as done for the case of \mathcal{F} - \mathcal{F} expressions).

Instead, it simply concatenates the operand peerages giving precedence to the smaller one. This motivates the following generalization.

Definition 10 *Let \mathcal{E} be a generic expression of the \mathcal{F} - \mathcal{G} algebra. Further, let \mathcal{E}_g be an expression which contains a reference to the \mathcal{G} operator and let \mathcal{E}_f be an expression which contains a reference to the \mathcal{F} operator. Also, let $P(\mathcal{E}) = [a_1; a_2; \dots; a_r]$, $P(\mathcal{E}_f) = [b_1; b_2; \dots; b_s]$, and $P(\mathcal{E}_g) = [c_1; c_2; \dots; c_t]$ be the peerages generated for \mathcal{E} , \mathcal{E}_f , and \mathcal{E}_g , respectively. Then,*

$$\mathcal{F}(\mathcal{E}, \mathcal{E}_g) = \begin{cases} [a_1; \dots; a_r; c_1; \dots; c_t] & \text{if } P(\mathcal{E}) \leq P(\mathcal{E}_g) \\ [c_1; \dots; c_t; a_1; \dots; a_r] & \text{if } P(\mathcal{E}) > P(\mathcal{E}_g) \end{cases} \quad (10)$$

$$\mathcal{G}(\mathcal{E}, \mathcal{E}_f) = \begin{cases} [a_1; \dots; a_r; c_1; \dots; c_t] & \text{if } P(\mathcal{E}) \geq P(\mathcal{E}_f) \\ [c_1; \dots; c_t; a_1; \dots; a_r] & \text{if } P(\mathcal{E}) < P(\mathcal{E}_f) \end{cases} \quad (11)$$

This rule allows computing the resultant peerage for generic \mathcal{F} - \mathcal{G} expressions such that property (6) is preserved (which is not true in [8]).

6 Additional Properties

In this section we discuss some properties of the \mathcal{F} - \mathcal{G} algebra. We demonstrate, for instance, that the \mathcal{F} and \mathcal{G} operations are complementary for ranking purposes.

The distributive property restricts the evaluation of multiple criteria queries to the the *min* and *max* functions (lemma 1). The problem is that distributivity is a too strict rule. For instance, it requires the output of the expressions $\mathcal{F}(a, \mathcal{G}(b, c))$ $\mathcal{G}(\mathcal{F}(a, b), \mathcal{F}(a, c))$ to be exactly the same. For ranking purposes, it is possible to define less restrictive transformations which, instead of preserving the quantitative rank associated to an approximate answer, preserve the order of this answer in the final ranking. Such transformations are called *ranking preserving* and lead to an objective interpretation for the relationship between the \mathcal{F} and \mathcal{G} operators.

Definition 11 *Let D_{fg} be the domain of all \mathcal{F} - \mathcal{G} expressions which can be written with the degrees of compatibility a_1, a_2, \dots, a_n and let $\mathcal{E} \in D_{fg}$ be an \mathcal{F} - \mathcal{G} expression in this domain. The transformation $\phi, \phi : D_{fg} \rightarrow D_{fg}$, is said to be ranking preserving (also called an rp-transformation) iff the ranking generated by \mathcal{E} is the same as the*

ranking generated by $\phi(\mathcal{E})$. To indicate that the ranking is preserved we introduce the operator $=_{rp}$ as follows.

$$\mathcal{E} =_{rp} \phi(\mathcal{E}) \quad (12)$$

Thus, an rp-transformation preserves the ordering of the approximate answers relative to a query written as an \mathcal{F} - \mathcal{G} expression. We are now ready to define some rp-transformations involving \mathcal{F} and \mathcal{G} .

Lemma 8 *Let $[a_1; a_2; \dots; a_n]$ be the peerage generated for an approximate answer t_a relative to an \mathcal{F} - \mathcal{F} or \mathcal{G} - \mathcal{G} query expression. Such peerage defines the order of the approximate answer t_a in the final ranking \mathcal{R} . Let $\overline{\mathcal{R}}$ be the ranking which is the reverse of \mathcal{R} (i.e., if $\mathcal{R} = |t_a \succ t_b \succ \dots \succ t_p|$ then $\overline{\mathcal{R}} = |t_p \succ \dots \succ t_b \succ t_a|$) and let $[\overline{a_1}; \overline{a_2}; \dots; \overline{a_n}]$ be a reference to the position of t_a in the ranking $\overline{\mathcal{R}}$. The transformation*

$$\phi([a_1; a_2; \dots; a_n]) = [\overline{a_1}; \overline{a_2}; \dots; \overline{a_n}] \quad (13)$$

is an rp-transformation i.e.,

$$[a_1; a_2; \dots; a_n] =_{rp} [\overline{a_1}; \overline{a_2}; \dots; \overline{a_n}] \quad (14)$$

Proof Let t_a and t_b be two tuples and let the respective peerages be $[a_1; a_2; \dots; a_n]$ and $[b_1; b_2; \dots; b_n]$. Consider first the case of an \mathcal{F} - \mathcal{F} expression. Clearly, it has to be $a_1 < a_2 < \dots < a_n$ and $b_1 < b_2 < \dots < b_n$. Without loss of generality, assume that t_a appears before t_b in the final ranking \mathcal{R} (i.e., $t_a \succ t_b$ and t_a is preferable to t_b). It has to be $[a_1; a_2; \dots; a_n] > [b_1; b_2; \dots; b_n]$. Let k be the smallest integer for which $a_k > b_k$. Clearly, $a_i = b_i$ for $1 \leq i < k$. Consider now the negated peerages $[\overline{a_1}; \overline{a_2}; \dots; \overline{a_n}]$ and $[\overline{b_1}; \overline{b_2}; \dots; \overline{b_n}]$. Recall that $a_i = 1 - \overline{a_i}$ and $b_i = 1 - \overline{b_i}$. Then, $\overline{a_i} = \overline{b_i}$ for $1 \leq i < k$ and $\overline{a_k} < \overline{b_k}$. Thus, according to the negated peerages, it is $t_b \succ t_a$ which is reversed to yield $t_a \succ t_b$ and the transformation is ranking preserving. The demonstration for the case of a \mathcal{G} - \mathcal{G} expression is analogous. \square

An immediate consequence of the rp-transformation above is as follows.

Lemma 9 *Let $\overline{\mathcal{F}}$ and $\overline{\mathcal{G}}$ indicate the operation of reversing the ranking generated by \mathcal{F} and \mathcal{G} respectively. Then,*

$$\mathcal{F}(a, b, \dots, q) =_{rp} \overline{\mathcal{G}}(\overline{a}, \overline{b}, \dots, \overline{q}) \quad (15)$$

$$\mathcal{G}(a, b, \dots, q) =_{rp} \overline{\mathcal{F}}(\overline{a}, \overline{b}, \dots, \overline{q}) \quad (16)$$

Proof Without loss of generality, assume that $\mathcal{F}(a, b, \dots, q) = [a; b; \dots; q]$. Then, $G(\bar{a}, \bar{b}, \dots, \bar{q}) = [\bar{a}; \bar{b}; \dots; \bar{q}]$. From lemma 8, $\mathcal{F}(a, b, \dots, q) = \overline{\mathcal{G}}(\bar{a}, \bar{b}, \dots, \bar{q})$. The proof for $\mathcal{G}(a, b, \dots, q)$ is analogous. \square

Thus, \mathcal{F} and \mathcal{G} can be interpreted as complementary operators in rp-transformations. Further, notice that the above properties resemble the De Morgan's laws for boolean algebra but with two key differences. First, lemma 9 does not establish a mathematical identity because the peerages generated by $\mathcal{F}(a, b, \dots, q)$ and $\overline{\mathcal{G}}(\bar{a}, \bar{b}, \dots, \bar{q})$ are not the same. Second, in the De Morgan's laws the operation of negating individual atoms and the operation of negating conjunctive/disjunctive expressions is unique while in the \mathcal{F} - \mathcal{G} algebra we must distinguish between the operation of negating a degree of compatibility (e.g., \bar{a}) and the operation of reversing the ranking (e.g., $\overline{\mathcal{F}}$) which have a very distinct nature.

We extend the above result as follows.

Lemma 10

$$\begin{aligned} \mathcal{F}(\mathcal{F}(a_1, \dots, a_n), \mathcal{G}(b_1, \dots, b_m)) &=_{rp} \overline{\mathcal{G}}(\mathcal{G}(\bar{a}_1, \dots, \bar{a}_n), \mathcal{F}(\bar{b}_1, \dots, \bar{b}_n)) \\ \mathcal{G}(\mathcal{F}(a_1, \dots, a_n), \mathcal{G}(b_1, \dots, b_m)) &=_{rp} \overline{\mathcal{F}}(\mathcal{G}(\bar{a}_1, \dots, \bar{a}_n), \mathcal{F}(\bar{b}_1, \dots, \bar{b}_n)) \end{aligned}$$

Proof Without loss of generality, let $a_1 < a_2 < \dots < a_n, b_1 < b_2 < \dots < b_m$, and $[a_1; a_2; \dots; a_n] < [b_1; b_2; \dots; b_m]$. Then, $\mathcal{F}(\mathcal{F}(a_1, \dots, a_n), \mathcal{G}(b_1, \dots, b_m)) = [a_1; \dots; a_n; b_1; \dots; b_m]$ and $\overline{\mathcal{G}}(\mathcal{G}(\bar{a}_1, \dots, \bar{a}_n), \mathcal{F}(\bar{b}_1, \dots, \bar{b}_n)) = [\bar{a}_1; \dots; \bar{a}_n; \bar{b}_1; \dots; \bar{b}_m]$ which ensures that the transformation preserves the ranking. The proof for $\mathcal{G}(\mathcal{F}(a_1, \dots, a_n), \mathcal{G}(b_1, \dots, b_m))$ is analogous. \square

This result can be generalized as follows.

Definition 12 Let \mathcal{E} be a generic \mathcal{F} - \mathcal{G} expression. Define \mathcal{E}^c , the complement of \mathcal{E} , as the \mathcal{F} - \mathcal{G} expression generated from \mathcal{E} by changing: (1) all appearances of \mathcal{F} into \mathcal{G} , (2) all appearances of \mathcal{G} into \mathcal{F} , and (3) all appearances of a degree of compatibility (e.g., a) into its negated form (e.g., \bar{a}).

For instance, if $\mathcal{E} = \mathcal{F}(\mathcal{G}(a, b), \mathcal{F}(c, d))$ then $\mathcal{E}^c = \mathcal{G}(\mathcal{F}(\bar{a}, \bar{b}), \mathcal{G}(\bar{c}, \bar{d}))$. Then,

Lemma 11

$$\mathcal{F}(\mathcal{E}, \mathcal{E}_g) = \overline{\mathcal{G}}(\mathcal{E}^c, \mathcal{E}_g^c) \tag{17}$$

$$\mathcal{G}(\mathcal{E}, \mathcal{E}_f) = \overline{\mathcal{F}}(\mathcal{E}^c, \mathcal{E}_f^c) \tag{18}$$

Proof Sketch Let $P(\mathcal{E}) = [a_1; a_2; \dots; a_n]$ be the peerage generated for \mathcal{E} . It can be demonstrated that $P(\mathcal{E}^c) = [\bar{a}_1; \bar{a}_2; \dots; \bar{a}_n]$ and the result above follows immediately. \square

7 Conclusions

We proposed an extended fuzzy algebra called *F-G algebra* for ranking approximate answers relative to a multiple criteria query request. Instead of combining the multiple degrees of compatibility into a number, the F-G algebra creates a list of them according to the query expression. These lists are then compared to decide the ranking of the corresponding tuples.

The main contributions of our research are three. First, the F-G algebra is simple and yet general enough to solve the problem of deciding preferences in the presence of multiple approximate criteria. Second, the algebra is derived from a set of basic axioms which are intuitive and can be easily grasped by the user. Third, the user has the option of stating preference relationships among attribute values in qualitative terms (e.g., greater than, lesser than, etc.) and use the F-G algebra to order the set of approximate answers according to these preferences. This last point is important because the user might find it difficult to express his preferences in numeric terms and/or reason quantitatively.

References

- [1] R. Bellman and M. Giertz. On the analytic formalism of the theory of fuzzy sets. *Inform. Sci.*, 5:149–156, 1973.
- [2] F. Cuppens and R. Demolombe. Cooperative answering: a methodology to provide intelligent access to databases. In *Second International Conference on Expert Database Systems*, Virginia, U.S.A., 1988.
- [3] Norbert Fuhr. A probabilistic framework for vague queries and imprecise information in databases. In *Proceedings of 16th VLDB Conference*, Brisbane, Australia, 1990.

- [4] H. Hamacher. On logical connectives of fuzzy statements and their affiliated truth function. In *Proc. Third European Meeting Cybernetics and Systems Res.*, Vienna, Austria, 1976.
- [5] E. P. Klement. Construction of fuzzy g-algebras using triangular norms. *Journal Math. Anal. Appl.*, 85, 1982.
- [6] Amihai Motro. Vague: A user interface to relational databases that permits vague queries. *ACM Transactions on Office Information Systems*, 6(3):187–214, July 1988.
- [7] F. Rabitti and P. Savino. An information retrieval approach for image databases. In *Proceedings of 18th VLDB Conference*, Vancouver, British Columbia, Canada, 1992.
- [8] Berthier A.N. Ribeiro and Richard Muntz. F-g: A fuzzy algebra for approximate answering in databases. In *Proc of the X Brazilian Symposium on Databases*, pages 365–376, Recife, Brazil, 1995.
- [9] S.K.M. Wong and Pawan Lingras. Representation of qualitative user preference by quantitative belief functions. *IEEE Transactions on Knowledge and Data Engineering*, 6(1):72–78, February 1994.
- [10] Ronald R. Yager. Some procedures for selecting fuzzy set-theoretic operators. *Int. Journal Gen. Syst.*, 8, 1982.
- [11] Ronald R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. In *Didier Dubois and Henri Prade and Ronald R. Yager, editors, Readings in Fuzzy Sets for Intelligent Systems*, California, 1993. Morgan Kaufmann.
- [12] L.A. Zadeh. Fuzzy sets. In D. Dubois, H. Prade, and R.R. Yager, editors, *Readings in Fuzzy Sets for Intelligent Systems*. Morgan Kaufmann, 1993.