Incorporating User Preferences in Multimedia Queries

(Extended Abstract)

Ronald Fagin and Edward L. Wimmers

IBM Almaden Research Center
650 Harry Road
San Jose, California 95120-6099
email: fagin@almaden.ibm.com, wimmers@almaden.ibm.com

Abstract. In a multimedia database system, queries may be fuzzy: thus, the answer to a query such as *(Color='red')* may not be 0 (false) or 1 (true), but instead a number between 0 and 1. A conjunction, such as *(Color='red')* $\wedge$ *(Sound='loud')*, is evaluated by first evaluating the individual conjuncts and then combining the answers by some scoring function. Typical scoring functions include the min (the standard scoring function for the conjunction in fuzzy logic) and the average. We address the question of how to permit the user to weight the importance of atomic subformulas. In particular, we give a semantics for permitting non-uniform weights, by giving an explicit formula (that is based on the underlying scoring function). This semantics permits an efficient implementation with a low database access cost in a multimedia database system in important cases of interest.

1 Introduction

A database system faces the task of responding to queries. In a traditional database system, all queries deal with Boolean values, since a property is either true or false. As queries over multimedia data become more prevalent, it is important to permit various shades of gray. For example, in searching for a red picture, the user is unlikely to want a Boolean value that says whether the picture is red or not. More likely, the user would prefer a "score" giving the redness of a particular picture.

In general, a user might want to query not only over a single multimedia property, but might wish to take into account several properties. For example, the user might be interested in a movie clip that has a predominantly red scene with a loud noise in the sound track. In this case, there is likely to be a score giving the redness of the scene and a different score giving the loudness of the sound. These two scores must be combined into a single score. One way to do this is to use fuzzy logic. Such an approach is taken by the Garlic system, which is being developed at the IBM Almaden Research Center, and which provides access to a variety of data sources, including multimedia. See [CHS+95, CHN+95] for a discussion of the Garlic system, and [Fa96] and [CG96] (along with Section 8 of

this paper) for a discussion of algorithms with a low database access cost for computing distributed scores (where different "black boxes" produce the various scores that must be combined).

However, there is an additional problem that must be addressed. It is unlikely that the user equally values the attributes being queried. For example, the user might like to inform the multimedia system to give extra weight to the picture and less weight to the sound. In the user interface, *sliders* are often used to convey this information to the system. Sliders are bars on the screen that indicate the importance of each attribute. The user moves his mouse to slide an indicator along the bar in order to increase or decrease the weighting of a given attribute.

The contribution of this paper is to give an explicit formula for incorporating weights that can be applied no matter what the underlying method is for combining scores (the average, the min, etc.). The formula we give is surprisingly simple, in that it involves far fewer terms than we might have guessed. In addition, its database access cost is low. It has two further desirable properties. The first desirable property is that when all of the weights are equal, then the result obtained is simply the underlying method for combining scores. Intuitively, this says that when all of the weights are equal, then this is the same as considering the unweighted case. Another way to say this is that if the user does not see any sliders, then this is the same as if the sliders exist and are all set to a default value where the weights are equal. The second desirable property is that if a particular argument has zero weight, then that argument can be dropped without affecting the value of the result. It turns out that if these two desirable properties hold, then under one additional assumption (a type of local linearity), our formula gives the unique possible answer.

2 Examples

Assume that there are two scores, namely x_1 and x_2. These scores are numbers (typically between 0 and 1, where 1 represents a perfect match) that represent how well an object rates on a particular attribute. For example, x_1 might be a score indicating how red a picture is, and x_2 might be a score indicating how loud a sound is. How should these scores be combined to reflect an "overall score" that reflects both the redness and the loudness? Should we take the average of the scores? Or what should we do? Not surprisingly, there are many possible answers, depending on the issues at hand.

One context where this issue of combining scores arises is fuzzy logic. In particular, a score must be assigned to a conjunction $A_1 \wedge A_2$ that is a function of the scores of A_1 and A_2. In his original paper [Za65], Zadeh defined the score of $A_1 \wedge A_2$ to be the min of the scores of A_1 and A_2. Similarly, he defined the score of the disjunction $A_1 \vee A_2$ to be the max of the scores of A_1 and A_2. Zadeh's choices were later justified by a famous result of Bellman and Giertz [BG73], which was extended and simplified by Yager [Ya82], Voxman and Goetschel [VG83], Dubois and Prade [DP84], and Wimmers [Wi96]. They showed that min and max are the unique choices that should be assigned to the conjunction and disjunction,

respectively, that fulfill certain natural conditions. There is a large literature on other possible choices for scoring functions in fuzzy logic: see, for example, the discussion in Zimmermann's textbook [Zi91].

The question of interest in our paper is as follows. Assume that some method, such as the average or the min, is given for combining scores. How do we modify this method if we decide now that we do not want to assign equal weight to the scores? In the example we gave earlier, assume that the user cares twice as much about the color of the picture as he does about the loudness of the sound. How should he combine the color score and the volume score to obtain an overall score? In the case of average, the answer is fairly clear. We would assign a weight $\theta_1 = 2/3$ to the color, and a weight $\theta_2 = 1/3$ to the volume. (The weights must sum to one, and the weight for color, namely θ_1, should be twice the weight θ_2 for volume.) We then take the weighted sum $\theta_1 x_1 + \theta_2 x_2$. But what if we are using a different underlying method than the average for combining scores?

For the rest of this section, we assume that as in standard fuzzy logic, the underlying method is to take the min. Assume again that we wish to weight the scores, where θ_1 is the weight for color, and θ_2 is the weight for volume. Then we cannot simply take the result to be $\theta_1 x_1 + \theta_2 x_2$. For example, if we are indifferent to color versus volume, so that we weight them equally with $\theta_1 = \theta_2 = 1/2$, then we would get the wrong answer by using $\theta_1 x_1 + \theta_2 x_2$, since this does not give us the min of x_1 and x_2. (We are assuming here that we use the underlying, or "unweighted", method for combining scores when the θ_i's are equal. Later, we shall make such assumptions explicit.) What should the answer be, as a function of x_1, x_2, and θ_1? (Here we do not need to include θ_2 as a parameter, since $\theta_2 = 1 - \theta_1$.)

Assume without loss of generality that $x_1 \leq x_2$. If $\theta_1 = 1/2$, then the answer should be x_1, since as we noted, when the weights are equal, we should use the unweighted method for combining, which in this case is the min. If $\theta_1 = 0$, then the answer should be x_2. This is under the assumption that when an argument has 0 weight, then it can be "dropped"; this is another assumption that will be made explicit later. Similarly, if $\theta_1 = 1$, so that $\theta_2 = 0$, then the answer should be x_1.

What about values of θ_1 other than 0, 1, or 1/2? Since the value is x_1 when $\theta_1 = 1/2$, it is reasonable to argue that the value should be x_1 whenever $\theta_1 \geq 1/2$; after all, if the value of x_2 becomes irrelevant (as long as it is bigger than x_1) for $\theta_1 = 1/2$, then surely it should be irrelevant for any larger value of θ_1, where we are weighting the first value (the x_1 value) even more highly. Another argument that the value should be x_1 whenever $\theta_1 \geq 1/2$ is that the value is x_1 for both $\theta_1 = 1/2$ and $\theta_1 = 1$, and so it should be the same for intermediate values. Later, we shall give a local linearity argument that says that the value should be x_1 whenever $\theta_1 \geq 1/2$. Furthermore, this local linearity argument says that the value when $\theta_1 < 1/2$ should be the appropriate linearly interpolated value between the value x_2 when $\theta_1 = 0$, and the value x_1 when $\theta_1 = 1/2$: this value is $2(x_1 - x_2)\theta_1 + x_2$.

What would we do when there are three arguments x_1, x_2, and x_3, and three

weights θ_1, θ_2, and θ_3? Here the answer is not at all clear *a priori*. Our results enable us to answer this question, under reasonable assumptions. Our methods in this paper work for arbitrary scoring functions, not just average and min.

3 Definitions

We assume that we are given a finite index set $\mathcal{I}$, that intuitively represents the set of attributes. In the example we considered earlier, these attributes would include color and volume. We typically use I to denote some non-empty subset of $\mathcal{I}$. Let D be a set that represents the arguments of the scoring functions. In the case of fuzzy logic, we take D to be the closed interval $[0,1]$. We shall take a *tuple X (over I)* to be a function with domain I and range D. We shall usually write x_i for $X(i)$. Let S be the set of possible scores; we shall take S to be an interval of real numbers.[1] In the case of fuzzy logic, S (like D) is the closed interval $[0,1]$. A *scoring function b_I (over I)* is a function with domain D^I (the set of tuples over I) and range S, the set of scores. When $D = S$, as in fuzzy logic, the scoring function combines a collection of scores to obtain an overall score. In this case, it would certainly be natural to assume that $b_{\{i\}}(X) = x_i$; this says intuitively that the score when we restrict our attention to i is simply the value at i. For the sake of generality we do not assume that $b_{\{i\}}(X) = x_i$, even when $D = S$, since we never need this assumption.

Let $\mathcal{B}$ be a set of scoring functions that contains one scoring function b_I for each nonempty $I \subseteq \mathcal{I}$. This scoring function b_I is over I, that is, has domain D^I. We refer to $\mathcal{B}$ as an *unweighted collection of scoring functions*, to distinguish it from a weighted collection, which we shall define shortly.

A *weighting (over I)* is a function Θ with domain a nonempty index set $I \subseteq \mathcal{I}$ and range the closed interval $[0,1]$, whose values $\Theta(i)$ sum to 1. Addition and scalar multiplication are defined in the usual way: $(\alpha \cdot \Theta)(i) = \alpha \cdot \Theta(i)$ for real numbers α, and $(\Theta + \Theta')(i) = \Theta(i) + \Theta'(i)$. We shall write θ_i for $\Theta(i)$.

Let $\mathcal{F}$ be a set of scoring functions that contains one scoring function f_Θ for each nonempty index set $I \subseteq \mathcal{I}$ and each weighting Θ over I. If Θ is over I, that is, has domain I, then the scoring function f_Θ is also over I, that is, has domain D^I. We refer to $\mathcal{F}$ as a *weighted collection of scoring functions*. This paper is concerned with the problem of obtaining a weighted collection of scoring functions from an unweighted collection of scoring functions in a way that is always applicable.

It is sometimes notationally convenient to define $b_I(X)$ even when the tuple X is not over I, but when X is over a superset of I. In this case, intuitively, the arguments outside of I are simply ignored. Formally, we let X' be the restriction of X to I, and define $b_I(X)$ to be $b_I(X')$. Similarly, when Θ is over I, and when X, X' are as above, we define $f_\Theta(X)$ to be $f_\Theta(X')$.

[1] We would like S to be an interval so that it is convex. We do not actually need S to be an interval of real numbers—it could just as well be an interval in an arbitrary ordered field. In particular, all of our results hold if S is taken to be an interval of rational numbers and each of the weights θ_i is a rational number.

If Θ is over I, we define the *support* of Θ to be the subset of I consisting of all $i \in I$ such that $\theta_i > 0$.

4 Desiderata

Assume that we are given an unweighted collection $\mathcal{B}$ of scoring functions. That is, intuitively, we are given a method of combining scores, such as the average or the min. We wish to define from the unweighted collection $\mathcal{B}$ a weighted collection $\mathcal{F}$ of scoring functions. That is, intuitively, we wish to decide how to combine scores when we weight the importance of the arguments. In this section, we consider two desirable properties for the relationship between $\mathcal{F}$ and $\mathcal{B}$. Later (Theorem 1), we shall show that under the additional natural assumption of a type of local linearity (that we also define in this section), there is a unique choice of $\mathcal{F}$ that satisfies these properties.

Our first desirable property relates $\mathcal{F}$ to $\mathcal{B}$: it says intuitively that each member of $\mathcal{F}$ with all of its weights equal should equal the corresponding member of $\mathcal{B}$. This corresponds to the intuition that $\mathcal{B}$ gives methods for obtaining an overall score in the case where no argument has higher weight than any other argument. Formally, denote the evenly-balanced weighting over I by E_I; thus, $(E_I)_i = 1/card(I)$ for each $i \in I$, where $card(I)$ denotes the cardinality of I. We say that the weighted collection $\mathcal{F}$ of scoring functions is *based on* the unweighted collection $\mathcal{B}$ of scoring functions if $f_{E_I} = b_I$ for every nonempty $I \subseteq \mathcal{I}$.

Our second desirable property says intuitively that if a particular argument has zero weight, then that argument can be dropped without affecting the value of the result. Formally, a weighted collection $\mathcal{F}$ of scoring functions is *compatible* if whenever Θ and X are over the same index set, and Θ' is the restriction of Θ to the support of Θ, then $f_\Theta(X) = f_{\Theta'}(X)$.

These two desirable properties are really essential—any method of going from an unweighted collection of scoring functions to a weighted collection that does not satisfy these two properties is seriously flawed. The weighted case must bear some relation to the unweighted case, and our notion of "based on" is the only natural choice. The notion of "compatibility" is also the only natural choice for handling zero weights.

These two properties are not sufficient to determine a unique weighted collection from an unweighted collection of scoring functions. We now give another property ("local linearity") that is quite reasonable, and that, together with the other two properties, does uniquely determine a weighted collection. Furthermore, this new property leads to a simple formula for scoring functions in the weighted case. We first need some more definitions.

Two weightings are called *order-equivalent* if they never "clash" by disagreeing on the order of importance of two components. More formally, assume that Θ, Θ' are weightings over I. Then Θ, Θ' are *order-equivalent* if there do *not* exist $i, j \in I$ with $\theta_i < \theta_j$ and $\theta'_j < \theta'_i$ both holding. For example, $(.2, .7, .1)$ and $(.3, .5, .2)$ are order-equivalent because in both cases, the second entry is biggest, the first entry is next-biggest, and the third entry is smallest. It is clear that

order-equivalence is reflexive and symmetric. Order-equivalence is *not* transitive, since for example $(0,1)$ and $(1,0)$ are not order-equivalent, while $(0.5,0.5)$ is order-equivalent to both $(0,1)$ and $(1,0)$.

We now define local linearity, and argue that it is fairly natural. Intuitively, local linearity says that the scoring functions act like a balance. If two weightings are order-equivalent, then local linearity demands that the weighting that is the midpoint of two order-equivalent weightings should produce a score that is the midpoint of the two scores produced by the given weightings. In fact, local linearity extends beyond the midpoint to any weighting that is a convex combination of two order-equivalent weightings: if a weighting is a convex combination of two weightings that are order-equivalent, then local linearity demands that the associated score should be the same convex combination of the scores associated with the given weightings. Formally, we say that a weighted collection $\mathcal{F}$ of scoring functions is *locally linear* if whenever Θ and Θ' are order-equivalent and $\alpha \in [0,1]$, then

$$f_{\alpha\cdot\Theta+(1-\alpha)\cdot\Theta'}(X) = \alpha \cdot f_{\Theta}(X) + (1-\alpha)\cdot f_{\Theta'}(X). \tag{1}$$

Our main theorem (Theorem 1) gives an explicit, simple formula for obtaining a weighted collection $\mathcal{F}$ of scoring functions from an unweighted collection $\mathcal{B}$ of scoring functions. The weighted collection $\mathcal{F}$ is based on $\mathcal{B}$, compatible, and locally linear. Furthermore, the theorem says that $\mathcal{F}$ is the unique such weighted collection of scoring functions.

A weighted collection $\mathcal{F}$ of scoring functions is *totally linear* if equation (1) always holds, even when Θ and Θ' are not necessarily order-equivalent. Although we argued above that local linearity is a reasonable assumption, we do not believe that total linearity is sensible to demand. When the order of importance of two components changes, a dramatic shift might occur, and there is no reason to assume that the score associated with the midpoint has any relation to the score associated with the endpoint weightings. It might even occur that the midpoint of two weightings is not order-equivalent to either weighting. For example, $(0.3,0.4,0.3)$ is the midpoint of $(0.1,0.4,0.5)$ and $(0.5,0.4,0.1)$ but is not order-equivalent to either one.

Indeed, as we shall see from Theorem 3, extending linearity to hold for all weightings and not merely the order-equivalent ones severely restricts the possible choices for the unweighted case. That is, total linearity of $\mathcal{F}$ can be obtained only for certain very restricted classes $\mathcal{B}$.

It is helpful to have a notation for selecting the most important (i.e., the largest) component of a weighting, down to the least important (i.e., the smallest) component of a weighting. A bijection σ that provides such a service is said to "order" the weighting. If σ orders a given weighting, then $\sigma(1)$ represents the most important component and $\sigma(m)$ represents the least important component (where m is the number of components). This is formalized in the next definition.

Assume that $m = card(I)$. A bijection σ from $\{1,\ldots,m\}$ onto I is said to *order* a weighting Θ over I if $\theta_{\sigma(1)} \geq \theta_{\sigma(2)} \geq \ldots \geq \theta_{\sigma(m)}$. It is easy to see that every weighting is ordered by some bijection σ. We define $\sigma[i]$ to be

$\{\sigma(1), \ldots, \sigma(i)\}$, for $1 \leq i \leq m$. Intuitively, if Θ is ordered by σ, then $\sigma[i]$ is the set of indices of the i largest θ_j's.

5 Main Theorem

We now give the main theorem of the paper. Let $\mathcal{B}$ be an unweighted collection of scoring functions. The theorem gives an explicit formula for a corresponding weighted collection $\mathcal{F}$ of scoring functions. As we shall explain, this formula is surprisingly simple, in that it involves far fewer terms than we might have guessed. Furthermore, the theorem says that this weighted collection is the unique one with the desirable properties we have discussed.

Theorem 1. *For every unweighted collection $\mathcal{B}$ of scoring functions, there exists a unique weighted collection $\mathcal{F}$ of scoring functions that is based on $\mathcal{B}$, compatible, and locally linear. Furthermore, if Θ and X are over I, if Θ is ordered by σ, and if $m = card(I)$, then*

$$f_\Theta(X) = m \cdot \theta_{\sigma(m)} \cdot b_{\sigma[m]}(X) + \sum_{i=1}^{m-1} i \cdot (\theta_{\sigma(i)} - \theta_{\sigma(i+1)}) \cdot b_{\sigma[i]}(X). \quad (2)$$

This formula (2) is both simple and simple to evaluate, given Θ and the b_I's. As an example of the formula, assume that $m = 3$, and $\theta_1 \geq \theta_2 \geq \theta_3$. Then formula (2) says that $f_{(\theta_1,\theta_2,\theta_3)}(x_1, x_2, x_3)$ equals

$$(\theta_1 - \theta_2) \cdot b_{\{1\}}(x_1) + 2 \cdot (\theta_2 - \theta_3) \cdot b_{\{1,2\}}(x_1, x_2) + 3 \cdot \theta_3 \cdot b_{\{1,2,3\}}(x_1, x_2, x_3).$$

We denote the family $\mathcal{F}$ of Theorem 1 by $\mathcal{F}(\mathcal{B})$. Note that if $b_I(X)$ is rational for each I, and if each θ_i is rational, then $f_\Theta(X)$ is also rational (cf. footnote 1 of Section 3). Note also that each of the m summands in (2) is nonnegative; this turns out to be useful for proving a later theorem (Theorem 4).

Clearly, we can rewrite (2) as a linear combination of the θ_i's: the result is

$$f_\Theta(X) = \theta_{\sigma(1)} \cdot b_{\sigma[1]}(X) + \sum_{i=2}^{m} \theta_{\sigma(i)} \cdot (i \cdot b_{\sigma[i]}(X) - (i-1) \cdot b_{\sigma[i-1]}(X)). \quad (3)$$

Unlike the situation with (2), some of the summands of (3) may be negative. Sometimes (such as in the derivation of Theorem 3) it is more useful to use (3) than (2).

The next result follows easily from Theorem 1.

Corollary 2. *Assume that Θ and X are over I, that Θ is ordered by σ, that $m = card(I)$, and that $f_\Theta(X) \in \mathcal{F}(\mathcal{B})$. Then there are $\alpha_1, \ldots, \alpha_m$ (with $\alpha_m = m \cdot \theta_{\sigma(m)}$) such that (1) $\alpha_i \geq 0$ for each i, (2) $\sum_{i=1}^{m} \alpha_i = 1$, and (3) $f_\Theta(X) = \sum_{i=1}^{m} \alpha_i \cdot b_{\sigma[i]}(X)$.*

Corollary 2 is rather surprising, since it is not clear *a priori* that f_Θ should depend only on $b_{\sigma[1]}, \ldots, b_{\sigma[m]}$, and not also on other b_I's. For example, when $m = 3$, each b_I is min, and $\theta_1 \geq \theta_2 \geq \theta_3$, then the formula for f_Θ is a convex combination of the three terms x_1, $\min(x_1, x_2)$, and $\min(x_1, x_2, x_3)$ only, and not of any of the terms x_2, x_3, $\min(x_1, x_3)$, or $\min(x_2, x_3)$. In general, f_Θ depends on only m of the $2^m - 1$ functions b_I. Thus, the formula is not only simple, but it is surprisingly simple.

At this point, it is perhaps worth commenting about the equality $\alpha_m = m \cdot \theta_{\sigma(m)}$ in Corollary 2. Now $0 \leq \theta_{\sigma(m)} \leq 1/m$ (the upper bound of $1/m$ follows from the fact that $m \cdot \theta_{\sigma(m)} = \alpha_m \leq 1$). Note that when $\theta_{\sigma(m)} = 1/m$ (so that $\Theta = E_I$), then $\alpha_m = 1$; then from Corollary 2 we see that $f_\Theta(X) = b_{\sigma[i]}(X)$. This reflects the fact that $\mathcal{F}$ is based on $\mathcal{B}$. Note that $b_{\sigma[m]}$ is the only member of $\mathcal{B}$ in $\sum_{i=1}^m \alpha_i \cdot b_{\sigma[i]}(X)$ of Corollary 2 whose index set is all of I. In particular, when $\theta_{\sigma(m)} = 0$, then the index set of every member of $\mathcal{B}$ in $\sum_{i=1}^m \alpha_i \cdot b_{\sigma[i]}(X)$ is contained in the proper subset $\{\sigma(1), \ldots, \sigma(i-1)\}$ of I. This corresponds to the fact that $\mathcal{F}(\mathcal{B})$ is compatible, that is, scores with zero weight do not affect the result. We note that the fact that $\alpha_m = m \cdot \theta_{\sigma(m)}$ turns out to be useful in the proof of Theorem 4.

We close this section with some intuitive remarks about the geometry behind obtaining the weighted collection of scoring functions from the unweighted collection. In fact, historically, this geometric intuition is what led us first to discover Theorem 1. We shall show how to determine a formula for f_Θ by induction on the number of nonzero entries of Θ. To start off the induction, note that if Θ has only one nonzero entry, so that $\theta_i = 1$ for some i, then then f_Θ is uniquely determined, since then $f_\Theta(X) = f_{E_{\{i\}}}(X) = b_{\{i\}}(X)$ from the fact that $\mathcal{F}$ is based on $\mathcal{B}$. Assume now that Θ has m nonzero entries. By compatibility, we can assume that Θ is over an index set I of size m. Let S be the $(m-1)$-dimensional hyperplane in m-dimensional Euclidean space (indexed by I), where S is defined by $\sum_{i \in I} \theta'_i = 1$. Let R be the (bounded) subregion of S where $\theta'_i \geq 0$ for each $i \in I$. For each $i \in I$, let B_i be the $(m-2)$-dimensional hyperplane that is the intersection of S with the $(m-1)$-dimensional hyperplane defined by $\theta'_i = 0$. Then the boundary B of R is the union of the B_i's. Each Θ' in B has at least one 0 entry. Therefore, by induction hypothesis (and by compatibility), we can assume that we already have determined a formula for $f_{\Theta'}$ for each Θ' in B. Now Θ is a linear combination of E_I and of some Θ' in B; say $\Theta = \alpha \cdot E_I + (1-\alpha) \cdot \Theta'$, where $\alpha \in [0, 1]$. (In fact, we see from the uniqueness part of the proof of Theorem 1 that $\alpha = m \cdot \theta_{\sigma(m)}$ when Θ is ordered by σ.) We know that $f_{E_I} = b_I$, since $\mathcal{F}$ is based on $\mathcal{B}$, and by induction hypothesis we know a formula for $f_{\Theta'}$. By local linearity, we know that we can then take f_Θ to be $\alpha \cdot b_I + (1-\alpha) \cdot f_{\Theta'}$. This turns out to give us the formula in the statement of Theorem 1.

6 Totally Linear Collections

When is the class $\mathcal{F}(\mathcal{B})$ not only locally linear, but even totally linear? Theorem 3 below tells us that this happens only in very limited circumstances, where the values of b_I with $card(I) > 1$ are completely determined by the values of b_I with $card(I) = 1$. In fact, under the assumption of total linearity, it follows from Theorem 3 below (in the fact that part 1 implies part 2) that $b_I(X)$ must be the average of $b_{\{i_1\}}(X), \ldots, b_{\{i_m\}}(X)$ if $I = \{i_1, \ldots, i_m\}$.

It is not surprising that if $f_\Theta(X)$ is the weighted average of $b_{\{i_1\}}(X)$, ..., $b_{\{i_m\}}(X)$, weighted according to Θ, then $\mathcal{F}(\mathcal{B})$ is totally linear. The fact that part 1 implies part 3 in Theorem 3 below shows that this is the only possible way for $\mathcal{F}(\mathcal{B})$ to be totally linear.

Theorem 3. *The following are equivalent:*

1. $\mathcal{F}(\mathcal{B})$ *is totally linear.*
2. $b_I(X) = (1/card(I)) \cdot \sum_{i \in I} b_{\{i\}}(X)$ *holds whenever* X *is over* I.
3. $f_\Theta(X) = \sum_{i \in I} \theta_i \cdot b_{\{i\}}(X)$ *holds whenever* Θ *and* X *are over* I.

Under the natural assumption (discussed near the beginning of Section 3) that $b_{\{i\}}(X) = x_i$, part 3 says that $f_\Theta(X) = \sum_{i \in I} \theta_i \cdot x_i$, a simple linear combination.

7 Inherited Properties

So far the scoring functions have been completely arbitrary. In practice, these functions usually enjoy many properties such as continuity, monotonicity, etc. As we discuss in this section, these properties are inherited by scoring functions in the corresponding weighted family.

If X and X' are each tuples over the same index set I, let us write $X \geq X'$ if $x_i \geq x'_i$ for each $i \in I$, and $X > X'$ if $x_i > x'_i$ for each $i \in I$. Let f be a scoring function over I. We say that f is *monotonic* if $f(X) \geq f(X')$ whenever $X \geq X'$, and *strictly monotonic* if $f(X) > f(X')$ whenever $X > X'$. We certainly expect a scoring function to be monotonic: intuitively, if the individual scores according to X are each at least as big as the corresponding scores according to X', then the overall score of X should be at least as big as the overall score of X'. Similarly, we expect a scoring function to be strictly monotonic; if it is monotonic but not strictly monotonic, then there is a portion of the domain where the scoring function is insensitive. In fact, in Section 9 we shall mention an example of a scoring function that is monotonic but not strictly monotonic, that arises under a certain rule for obtaining weighted families of scoring functions; we feel that such a scoring function is undesirable. We also expect a scoring function to be continuous: slight changes in individual scores should lead to only slight changes in the overall score.

We now define a notion of a scoring function being *strict*. This notion will be important in Section 8. For this notion, we assume that, as in fuzzy logic,

the domain D and the range S are both the closed interval $[0,1]$. Intuitively, a scoring function is strict if it takes on the value 1 precisely when all of its arguments are 1. In making this definition precise, there is a slight subtlety, brought on by the fact that, for example, we have defined $b_I(X)$ whenever X is over a superset of I. Formally, we say that b_I is strict if whenever X is over I, then $b_I(X) = 1$ iff $x_i = 1$ for every $i \in I$. Strictness is certainly a property we would expect of any scoring function that is used to evaluate the conjunction. We now define strictness in the weighted case. Assume that Θ is over I. We say that f_Θ is strict if whenever X is over I, then $f_\Theta(X) = 1$ iff $x_i = 1$ for every $i \in I$. Furthermore, we say that f_Θ *has full support* if the support of Θ is I. We would not expect f_Θ to be strict unless f_Θ has full support.

A scoring function f is called *translation-preserving* if $f(X') = f(X) + a$ provided $x'_i = x_i + a$ for every i in the domain of X. The idea behind a translation-preserving scoring function is that if all the input scores are increased by the same amount, then the output score is increased by that same amount. Unlike the situation with continuity and monotonicity, we do not necessarily expect a scoring function to be translation-preserving. Of course, the min function is translation-preserving. In fact, as we shall discuss later (Proposition 7), min is the unique monotonic, translation-preserving scoring function (up to boundary conditions).

A scoring function f *satisfies betweenness* if $\min X \leq f(X) \leq \max X$ for every X. This says that the resulting score lies between the smallest and largest of its arguments. This is certainly a natural property that we would expect scoring functions to enjoy. A scoring function f *satisfies identity* if $f(x, ..., x) = x$ for every x in the domain D. This says that if all of the "input scores" are equal, then the resulting output score has this same value. It is clear that if a scoring function satisfies betweenness, then it also satisfies identity.

The next theorem says that the properties we have discussed in this section are inherited by the weighted family of scoring functions.

Theorem 4. *1. If every scoring function in $\mathcal{B}$ is continuous (resp. is monotonic, is strictly monotonic, is translation-preserving, satisfies betweenness, satisfies identity), then every scoring function in $\mathcal{F}(\mathcal{B})$ is continuous (resp. is monotonic, is strictly monotonic, is translation-preserving, satisfies betweenness, satisfies identity) as well.*

2. If every scoring function in $\mathcal{B}$ is strict, then every scoring function in $\mathcal{F}(\mathcal{B})$ that has full support is strict as well.

We next consider a property that is a property not of an individual scoring function, but of a class of scoring functions. We would like to show that some sort of symmetry is inherited by weighted collections from unweighted ones. Normally, a function is called symmetric if it is unchanged by any permutation of its arguments. In our setting, this translates to saying that we can take any permutation of the indices without changing the result. We now formally define the notion of symmetry. In this definition, $\circ$ represents functional composition, and $\delta(I)$ represents the image of the set I under the function δ when I is a subset of the domain of δ.

An unweighted collection $\mathcal{B}$ of scoring functions is called *symmetric* if $b_{\delta(I)}(X) = b_I(X \circ \delta)$ for each permutation δ of $\mathcal{I}$, each nonempty $I \subseteq \mathcal{I}$, and each $X \in D^{\delta(I)}$. A weighted collection $\mathcal{F}$ of scoring functions is called *symmetric* if $f_\Theta(X) = f_{\Theta \circ \delta}(X \circ \delta)$ for each permutation δ of $\mathcal{I}$, each Θ over $\delta(I)$, and each $X \in D^{\delta(I)}$. (Note that f_Θ and X are over $\delta(I)$, and that $f_{\Theta \circ \delta}$ and $X \circ \delta$ are over I.) Being symmetric means intuitively that we do not distinguish among the arguments.

Theorem 5. *If $\mathcal{B}$ is symmetric, then $\mathcal{F}(\mathcal{B})$ is symmetric.*

Note that the average and the min are symmetric scoring functions. In fact, we believe that most naturally-occurring scoring functions are symmetric. In some circumstances, there might be a reason to treat the arguments differently and thereby take a nonsymmetric scoring function. One such scenario could arise if, say, all of the scores about one particular attribute are guaranteed to be at most $1/2$, but this is not true about the other attributes. Assume that in this case we are "designing" a scoring function with two arguments x_1 and x_2, where x_1 is guaranteed to be at most $1/2$. Instead of, say, taking the scoring function to be the average $(x_1 + x_2)/2$, it would probably be more reasonable to "normalize" and take $(2x_1 + x_2)/2$ as a scoring function. This leads to a family that is not symmetric.

8 Low Database Access Cost

Garlic [CHS+95, CHN+95] is a multimedia information system being developed at the IBM Almaden Research Center. It is designed to be capable of integrating data that resides in different database systems as well as a variety of non-database data servers. A single Garlic query can access data in a number of different subsystems. An example of a nontraditional subsystem that Garlic accesses is QBIC [NBE+93] ("Query By Image Content"). QBIC can search for images by various visual characteristics such as color, shape, and texture. In [Fa96], the first author developed an efficient algorithm for evaluating conjunctions in such a system, when the conjuncts are independent. In this section, we show that this algorithm can be carried over to the weighted case.

Let us begin with an example, where we deal first with the unweighted case. Consider again the fuzzy conjunction *(Color='red')* $\wedge$ *(Sound='loud')*. We denote this query by Q. Assume that two different subsystems deal with color and sound (for example, QBIC might deal with color). Garlic has to piece together information from both subsystems in order to answer the query Q. Let I be the index set {Color, Sound}, and let o be an object. Assume that the redness score of object o, as determined by the subsystem dealing with color, is x_1, and the loudness score of object o, as determined by the subsystem dealing with sound, is x_2. Then, in the setup of this paper, we would take the overall score of object o under query Q to be $b_I(x_1, x_2)$. Thus, the overall score is determined by applying the relevant scoring function (namely, b_I) to the redness score and the loudness score.

Let us say that we are interested in finding the top 10 answers to query Q (that is, the 10 objects with the highest overall scores, along with their scores). One way to do this would be to evaluate the query on every single object in the database, and take the 10 objects with the highest overall scores (ties would be broken arbitrarily). The problem with this naive algorithm is that there is a very high database access cost[2]: every single object in the database must be accessed. The first author [Fa96] gives an algorithm that is much more efficient, provided that the conjuncts are independent. He shows that if the scoring function (in this case, b_I) is monotonic, then the database access cost is of the order of the square root of the number of objects in the database. (More precisely, it is shown that if there are m conjuncts, and N objects in the database, then the database access cost for finding the top k objects in the database is $O(N^{(m-1)/m}k^{1/m})$, with arbitrarily high probability. For details about the probabilistic assumptions, see [Fa96].) Furthermore, it is shown that if the scoring function is strict, then this is optimal.

What about the weighted case, where, say, we care twice as much about the color as about the sound? It follows from Theorem 4 that if every scoring function in $\mathcal{B}$ is monotonic and strict, then every scoring function in $\mathcal{F}(\mathcal{B})$ that has full support is monotonic and strict as well. Now the upper bound in [Fa96] depends only on the scoring functions being monotonic, and the matching lower bound depends only on the scoring functions being strict. Therefore, we have the following theorem.

Theorem 6. *Assume that every scoring function in $\mathcal{B}$ is monotonic and strict. There is an algorithm $\mathcal{A}$ for finding the top k answers to the query determined by $f_\Theta \in \mathcal{F}(\mathcal{B})$. If the support of Θ consists of m independent attributes, then the database access cost for algorithm $\mathcal{A}$ is $O(N^{(m-1)/m}k^{1/m})$ with arbitrarily high probability, and this is optimal.*

9 Related Work

There is much work in the economics literature about indifference curves. This includes work on computing optimal indifference curves (which depend on user-supplied weightings). Since the focus there is on computing optimality and the focus in this paper is on combining scores in a way that preserves desirable properties, the other work is only tangentially related. See [KR76] for a more complete discussion.

We now discuss two methods from the literature for obtaining a weighted family of scoring functions from an unweighted family, and compare them with our approach. Each of these methods deals only with certain unweighted families, rather than, as in our approach, with arbitrary unweighted families.

[2] The cost model, including the definition of "database access cost", is defined formally in [Fa96]. Intuitively, the database access cost corresponds to the number of elements accessed in the database.

Method 1: The first method is inspired by a paper by Dubois and Prade [DP86]. It deals with the case where the min function is used in the unweighted case (in fact, the title of Dubois and Prade's paper is "Weighted Minimum and Maximum Operations in Fuzzy Set Theory"). Their underlying scenario and goals are actually quite different from ours (for example, instead of dealing with probabilities θ_i, they deal with possibility distributions [Za78]). Nonetheless, it is instructive to compare the explicit formula that they obtain for the "weighted min", and see how it fares under our criteria.

Assume again that the range of scores is $[0,1]$. Let X be a vector over I, let $b_I(X) = min_{i \in I}\{x_i\}$ and let

$$f_\Theta(X) = \min_{i \in I}\{\max\{1 - (\theta_i/M), x_i\}\}, \tag{4}$$

where $M = \max_{i \in I}\{\theta_i\}$. It is easy to check that $\mathcal{F}$ is a compatible collection of scoring functions that is based on $\mathcal{B}$. Note that $\mathcal{F}$ is not locally linear.

An attractive feature of this formula for f_Θ is that it is simple, it is continuous, monotonic, and strict, it satisfies betweenness and identity, and the corresponding family is symmetric. Since it is monotonic and strict, it follows that for this choice of f_Θ, there is an algorithm $\mathcal{A}$ as in Theorem 6 such that the last sentence of that theorem holds. However, the formula in (4) is not strictly monotonic. For example, let $\theta_1 = 2/3$ and $\theta_2 = 1/3$. The reader can easily verify that $f_\Theta(.7,.3)$ and $f_\Theta(.8,.4)$ are each equal to .5. In fact, it is easy to verify that if $x_2 \leq .5 \leq x_1$, then $f_\Theta(X) = .5$. This is undesirable, since it says intuitively that f_Θ is insensitive to its arguments in this region.

Also, f_Θ is not translation-preserving, since $f_\Theta(.8,.4) \neq f_\Theta(.7,.3) + .1$. We consider this undesirable because, as the following proposition shows, the essence of min (the underlying scoring function) is that it is monotonic and translation-preserving. That is, up to boundary conditions, it is uniquely determined by being monotonic and translation-preserving.

Proposition 7. min *is the unique monotonic, translation-preserving function* f *on* $[0,1]$ *for which* $f(0,1) = 0 = f(1,0)$.

The formula (4) is computationally a little simpler than (2). Therefore, there might be situations, where, say, computational simplicity is more important than strict monotonicity and translation invariance, when (4) would be preferable to use rather than (2).

Method 2: The second method is from a paper by Salton, Fox and Wu on information retrieval [SFW83], and deals with the case where a version of the Euclidean distance is used in the unweighted case.

Assume again that the range of scores is $[0,1]$. Let X be a vector over I, let

$$b_I(X) = \sqrt{\frac{\sum_{i \in I} x_i^2}{card(I)}}, \tag{5}$$

and let

$$f_\Theta(X) = \sqrt{\frac{\sum_{i \in I} \theta_i^2 x_i^2}{\sum_{i \in I} \theta_i^2}}. \tag{6}$$

It is easy to check that $\mathcal{F}$ is a compatible collection of scoring functions that is based on $\mathcal{B}$. Note that $\mathcal{F}$ is not locally linear. Unlike other scoring functions we have discussed, these take us out of the rationals.

The formula (6) is quite reasonable: it gives a natural generalization of the unweighted formula (5); it is continuous, strictly monotonic, and strict, it satisfies betweenness and identity, and the corresponding family is symmetric. It is not translation-preserving, but we would not expect it to be, since the formula in the unweighted case is not translation-preserving.

As was the case with Method 1 in this section, the formula for f_Θ, namely (6), is computationally easier than our formula (2) in this case, since only one square root is involved in (6), whereas m square roots are involved in (2). Also, as with Method 1 in this section, because of monotonicity and strictness, it follows that for this choice (6) of f_Θ, there is an algorithm $\mathcal{A}$ as in Theorem 6 such that the last sentence of that theorem holds.

One possible objection to (6) is that it is not clear why θ_i^2, rather than θ_i, is being used in the formula; either seems like a reasonable alternative. In fact, QBIC [NBE+93] also uses a variation of Euclidean distance, and in the weighted case uses θ_i rather than θ_i^2. Because of the specific form of the formula (5) for b_I (the unweighted case), there is a natural extension to f_Θ (the weighted case), which is (6). This is often not the situation; min is a good example.

The point of this example is that for certain special unweighted collections $\mathcal{B}$, there may be a natural way to obtain a weighted collection that is not the weighted collection $\mathcal{F}(\mathcal{B})$ that our methodology gives us. In fact, the extension (6) is more in the spirit of the unweighted case (5) than our extension (2). But our methodology has the advantage that it *always* gives us a (simple) way to obtain a weighted collection of scoring functions from an unweighted collection, no matter what the unweighted collection is.

10 Summary

The typical user query in a multimedia system asks for "good" matches (as indicated by a score) rather than "perfect" matches (as indicated by a Boolean value of 0 or 1). There are many possible methods for combining subscores into a single score. These methods include combining functions such as average, min, etc. Of course, it is rare that users wish to give equal weight to all components. This paper presents, by means of a surprisingly simple formula, a general method that extends any unweighted collection $\mathcal{B}$ of scoring functions to a weighted collection $\mathcal{F}(\mathcal{B})$ of scoring functions. This general method preserves a number of desirable properties, such as continuity and monotonicity. Furthermore, $\mathcal{F}(\mathcal{B})$ is the only weighted collection that is based on $\mathcal{B}$, compatible, and locally linear.

11 Acknowledgements

We thank Joe Halpern for many useful discussions and for the suggestion that we extend our results to include the non-symmetric case. We also thank Laura Haas and Joe Halpern for comments on the paper.

References

[BG73] R. Bellman and M. Giertz, On the Analytic Formalism of the Theory of Fuzzy Sets, *Information Sciences* **5** (1973), pp. 149–156.

[CG96] S. Chaudhuri and L. Gravano, Optimizing Queries over Multimedia Repositories, *Proc. ACM SIGMOD Conference*, 1996, pp. 91–102.

[CHS+95] M. J. Carey, L. M. Haas, P. M. Schwarz, M. Arya, W. F. Cody, R. Fagin, M. Flickner, A. W. Luniewski, W. Niblack, D. Petkovic, J. Thomas, J. H. Williams, and E. L. Wimmers, Towards Heterogeneous Multimedia Information Systems: the Garlic Approach, RIDE-DOM '95 (5th Int'l Workshop on Research Issues in Data Engineering: Distributed Object Management), 1995, pp. 124–131.

[CHN+95] W. F. Cody, L. M. Haas, W. Niblack, M. Arya, M. J. Carey, R. Fagin, M. Flickner, D. S. Lee, D. Petkovic, P. M. Schwarz, J. Thomas, M. T. Roth, J. H. Williams, and E. L. Wimmers, Querying Multimedia Data from Multiple Repositories by Content: the Garlic Project, *IFIP 2.6 3rd Working Conference on Visual Database Systems (VDB-3)*, 1995.

[DP84] D. Dubois and H. Prade, Criteria Aggregation and Ranking of Alternatives in the Framework of Fuzzy Set Theory, in *Fuzzy Sets and Decision Analysis* (H. J. Zimmermann, L. A. Zadeh, and B. Gaines, Eds.), TIMS Studies in Management Sciences **20** (1984), pp. 209–240.

[DP86] D. Dubois and H. Prade, Weighted Minimum and Maximum Operations in Fuzzy Set Theory, *Information Sciences* **39** (1986), pp. 205–210.

[Fa96] R. Fagin, Combining Fuzzy Information from Multiple Systems, *Fifteenth ACM Symp. on Principles of Database Systems*, 1996, pp. 216–226. Full version appears in `http://www.almaden.ibm.com/cs/people/fagin/pods96rj.ps`

[KR76] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, John Wiley & Sons, New York (1976).

[NBE+93] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, and P. Yanker, The QBIC Project: Querying Images by Content Using Color, Texture and Shape, *SPIE Conference on Storage and Retrieval for Image and Video Databases* (1993), volume 1908, pp. 173–187. QBIC Web server is `http://wwwqbic.almaden.ibm.com/`

[SFW83] G. Salton, E. A. Fox, and H. Wu, Extended Information Retrieval, *Comm. ACM* **26**,12 (1983), pp. 1022–1036.

[VG83] W. Voxman and R. Goetschel, A Note on the Characterization of the Max and Min Operators, *Information Sciences* **30** (1983), pp. 5–10.

[Wi96] E. L. Wimmers, Minimal Bellman-Giertz Theorems, to appear.

[Ya82] R. R. Yager, Some Procedures for Selecting Fuzzy Set-Theoretic Operations, *International Journal General Systems* **8** (1982), pp. 115–124.

[Za65] L. A. Zadeh, Fuzzy Sets, *Information and Control* **8** (1965), pp. 338–353.

[Za78] L. A. Zadeh, Fuzzy Sets as a Basis for a Theory of Possibility, *Fuzzy Sets and Systems* **1** (1978), pp. 3–28.

[Zi91] H. Zimmermann, *Fuzzy Set Theory*, Kluwer Academic Publishers, Boston (1991).