

Bipolar queries: a way to enhance the flexibility of database queries

Sławomir Zadrozny and Janusz Kacprzyk

Abstract In many real life scenarios the use of standard query languages may be ineffective due to the difficulty to express the real user requirements (information needs). The use of fuzzy logic helps to fight this ineffectiveness making it possible to model and properly process linguistic terms in queries. This way a user may express his or her requirements in a more intuitive and flexible way. Recently another dimension of such a flexibility attracted the attention of many researchers. Namely, it is now widely advocated that by specifying his or her requirements the user is usually having in mind both negative and positive preferences. Thus, a combination of an intuitive appeal of natural language terms in queries with a bipolar nature of preferences seems to be a next promising step in enhancing the flexibility of queries. We look at various ways of how to understand bipolarity in database querying, propose fuzzy counterparts of some crisp approaches and study their properties.

1 Introduction

Databases are indispensable for the functioning of modern societies. They have been in use for many years now, but only relatively recently they have found a widespread use among novice users having a limited computer literacy. In particular, the Internet provides a platform for many applications such as hotel booking systems that rely on the database access. The effectiveness and efficiency of these applications may be considered in two dimensions. First, traditionally considered aspects such as response time are of utmost importance. They are addressed using better data

Sławomir Zadrozny

Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warszawa, Poland, e-mail: Slawomir.Zadrozny@ibspan.waw.pl

Janusz Kacprzyk

Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warszawa, Poland, e-mail: Janusz.Kacprzyk@ibspan.waw.pl

storage and retrieval techniques, faster hardware, etc. Another dimension is related to the ease of expressing the requirements of a user searching for information in a database. In order to save the user from intricacies of the traditional database querying languages, some user friendly interfaces are usually provided. Such interfaces make it *technically* easier to form a query by the use of typical controls and toolbars. However, behind the scene, such a query have to be translated into a query language supported by a database in use, most often to SQL. This may be a sufficient solution in case of search criteria that are clear-cut as, e.g., in case of the search for name of a specific product at the Internet shopping site. On the other hand in many applications, such as related to hotel booking or real estate property search, the criteria of the user may be rather vague. This is mostly due to their original form in the mind of the user which is a natural language expression as in the query: “Find *cheap* houses located *near* a railway station”. In order to support such queries a proper modelling of natural language terms, such as “cheap” and “near” in the example above mentioned, have to be provided and an essential extension to the very database querying has to be implemented.

Fuzzy logic seems to offer an excellent solution in this respect and research in this area has a long history. The reader is referred to a recently published handbook (cf. Galindo [14]) and, in particular, to a survey of the research on the flexible, fuzzy querying of databases therein (cf. Zadrozny, De Tré, De Caluwe and Kacprzyk [24]). Thus, the user does not have to translate his or her requirements from the “internal” natural language form but can express them more directly via a flexible fuzzy query. This way a kind of a *conceptual* ease of use is achieved.

Recently, in addition to vagueness also the *bipolarity* of information and a need to take it into account are getting more and more advocated. We are here interested in the bipolarity of requirements (preferences) of the user looking for information in a database. This may be understood in a few slightly different ways which we will briefly discuss later on. Now let us adopt a general, database query oriented view on this feature of information.

Basically, a database query may be identified with a *condition* that the data sought should satisfy. Such a condition may often have a complex structure and comprise of some atomic conditions combined using logical connectives. Various extensions has been proposed in the literature in the framework of flexible querying. As mentioned above, the use of *fuzzy predicates* modelling *linguistic terms* in conditions has been advocated (cf., e.g., Zadrozny, De Tré, De Caluwe and Kacprzyk [24], Bosc [5], Kacprzyk and Zadrozny [15]). Also the assignment of the *importance weights* to particular parts of the condition has been proposed and studied (cf., e.g., Dubois and Prade [9]).

In case of bipolarity it is assumed, which is well founded on results of psychological research, that the user has in mind in fact two types of conditions:

- *hard constraints* which have to be met by the data sought, and
- just *preferences* making it possible to differentiate among the data items meeting the above mentioned hard constraints.

Such conditions, if they are crisp, define therefore two sets of data items:

- *rejected, infeasible*, etc., or, equivalently, taking a complement of the former, *acceptable, satisfactory, feasible* etc., and
- *preferred, desired*, etc.

Thus the former conditions provide the *negative* information indicating what should be avoided, while the latter provide the *positive* information indicating what is really preferred. This is the motivation for the use of the term “bipolar” to characterize queries comprising both types of conditions.

The bipolar queries (or, more generally, bipolar preferences) may be studied from different perspectives – some of them are presented in Section 2. For example, one may primarily look for a very important aspect of their proper theoretical modelling (cf., e.g., Dubois [11], Beferhat [1]) or, more specifically, for the ways to properly combine several preferred (positive) conditions (cf., e.g., Bosc and Pivert [3, 4]). On the other hand, one may focus on the question of a proper combination of both types of conditions in case they are fuzzy rather than crisp.

The following general interpretation of the bipolar queries may better explain the problem. Both the negative (required) and positive (preferred) conditions are treated as atomic. The data items sought have to satisfy the former unconditionally, while the latter is of somehow secondary importance. The relation between these two types of conditions may be meant in various ways and leads to different interpretations of the bipolar queries. The simplest approach is to use the positive conditions just to order the data items which satisfy the negative conditions. However as soon as the negative conditions are fuzzy, i.e., may be satisfied *to a degree*, it is not at all obvious what their satisfaction should mean. Thus, we focus on the question how the satisfaction degrees of both the negative and positive conditions may be combined.

From the adopted perspective, the way these conditions are jointly taken into account may be treated as the question of a definition/selection of an appropriate aggregation operator to be applied. In the literature this problem has been studied under different names, and sometimes in slightly different contexts, by many authors. In the framework of database querying the paper by Lacroix and Lavency [16] was first to propose such type of queries and triggered the interest of other researchers. This has led to the development of a more general concept of a *query with preferences* and a corresponding new relational algebra operator, *winnow*, introduced by Chomicki [6, 7]. Both Lacroix and Lavency’s as well as Chomicki’s approach deal with crisp conditions only. In Zadrożny [23] we propose a direct “fuzzification” of the approach by Lacroix and Lavency and in Zadrożny and Kacprzyk [26] we study some properties of this solution. In Zadrożny and Kacprzyk [25] we discuss a similar approach with respect to Chomicki’s *winnow* operator. Here we further develop this line of research and present it in a unified framework.

2 On some related works

Here we briefly review research on the bipolar queries of databases. Various approaches mentioned bear different names but they may be easily shown to deal with

the bipolar information, as discussed in Section 1. The name “bipolar queries” seems to be proposed and used for the first time by Dubois and Prade [10]. However we will start with a much earlier approach which triggered the interest of the database community in this type of queries.

We will consider in a usual way a relation as a data structure to be searched through. Let $T = \{t_j\}$ denote a set of tuples of this relation.

Lacroix and Lavency [16] were the first to propose the use of a query comprising two categories of conditions: one which is mandatory (C) and another which expresses just mere preferences (desires) (P). The bipolarity of these conditions becomes evident when one adopts the following interpretation. The former condition C may be seen as expressing the *negative* preferences: the tuples which do not satisfy it are definitely not matching the whole query. The latter condition P , on the other hand, expresses the *positive* preferences: a tuple satisfying it is preferred over another tuple not satisfying it, provided both tuples satisfy the mandatory condition C . These conditions will be referred to as a positive and negative condition, for short.

We will identify the negative and positive condition of a bipolar query with the predicates that represent them and denote them as C and P , respectively. For a tuple $t \in T$, $C(t)$ and $P(t)$ will denote that the tuple t satisfies the respective condition. Then, a bipolar query may be expressed in natural language as follows:

“Find tuples t satisfying C and possibly P ”

The bipolar queries may be exemplified by:

“Find a house cheaper than USD 250,000 and possibly located not more than two blocks from a railway station” (1)

Here the negative condition excludes houses more expensive than USD 250,000 and the positive condition favors houses located closer to a railway station. Such a query may be more formally written as

C and possibly P

or, equivalently, an answer to a bipolar query may be defined as the following set of tuples:

$\{t : C(t) \text{ and possibly } P(t)\}$ (2)

The above form puts emphasis on the question of a proper modelling of the aggregation of both types of conditions, which is expressed here with the use of the “and possibly” operator. This operator was used in a different context by Yager [21, 20], and Bordogna and Pasi [2].

According to the original (crisp) approach by Lacroix and Lavency [16] such an operator has an important property: the aggregation result depends not only on the explicit arguments, i.e., $C(t)$ and $P(t)$, but also on the content of the database. If there are no tuples meeting both conditions then the result of the aggregation is determined by the negative condition C alone. Otherwise the aggregation becomes

a regular conjunction of both conditions. This dependence is best expressed by the following logical formula [16] (showing, by the way, that in case of crisp conditions C and P , a bipolar query is easily expressible in the classical querying formalism of the relational data model, i.e., in the relational calculus):

$$C(t) \text{ and possibly } P(t) \equiv C(t) \wedge \exists s(C(s) \wedge P(s)) \Rightarrow P(t) \quad (3)$$

Lacroix and Lavency [16] consider only the case of crisp conditions C and P . Then this important property is preserved if the “first select using C and then order using P ” understanding of the bipolar query is adopted, i.e., the answer to the bipolar query (C, P) is generated as follows:

- find tuples satisfying C ,
- order them according to their satisfaction degree of P .

This understanding is predominant in the literature dealing with fuzzy extensions of the original concept of Lacroix and Lavency. Both, direct extensions proposed by Bosc and Pivert [3, 4] as well as a more sophisticated possibility theory based interpretation of this concept by Dubois and Prade [11] focus, in fact, on the proper treatment of *multiple* required and preferred conditions, basically assuming the above strategy as the way of combining the negative and positive conditions.

The research on such fuzzy compound conditions may be well illustrated with the papers by Bosc and Pivert [3, 4] and Dubois and Prade [10]. First, let us briefly recall an approach to the aggregation of multiple positive conditions proposed for the crisp case by Lacroix and Lavency [16]. They consider the case in which there is a set $\{P_i\}$ of preferred (positive) conditions rather than just one, which may be formally written as

$$C \text{ and possibly } \{P_i\} \quad (4)$$

The conditions P_i are meant to be combined in a non-standard way, i.e., are not treated as a Boolean combination. Lacroix and Lavency [16] proposed a few ways to aggregate them which are interesting from a practical point of view. Basically two types of such aggregation operators are considered: one based on the cardinality of the set of satisfied conditions P_i , and one based on a varying importance of these conditions. In the former case, a tuple satisfies a query (4) if:

- it satisfies the required condition C , and
- there is no tuple s satisfying C and more conditions P_i that t satisfies.

In the latter case the positive conditions are assumed to be linearly ordered and a tuple t satisfies a query (4) if:

- it satisfies the required condition C , and
- there is no tuple s satisfying C and a condition P_i , while $\neg P_i(t)$ and $P_j(t) \equiv P_j(s)$ for all $j < i$.

For both types of such compound positive conditions Lacroix and Lavency define an equivalent query in the relational calculus, in the spirit of (3).

Bosc and Pivert [3] study fuzzy counterparts of these types of compound positive conditions. For the cardinality based combination they consider a fuzzy set H_t of positive conditions P_i satisfied by a given tuple t , where H_t 's membership function is defined as $\mu_{H_t}(P_i) = P_i(t)$; remember that P_i is now a fuzzy condition, $P_i(t) \in [0, 1]$ denotes its satisfaction degree by the tuple t and a tuple satisfies (matches) the whole bipolar query to a *degree*. Then, the scalar cardinality (the so-called Σ Count) of the fuzzy set H_t is used as the matching of degree of the tuple t with respect to the combination of the positive conditions P_i (after a proper normalization). Bosc and Pivert also propose a fuzzy counterpart of the importance based combination of the positive conditions introducing a Hierarchical Combination Operator; for details see [4]. Anyway, in their approach to the evaluation of the overall bipolar query with fuzzy negative and positive conditions Bosc and Pivert follow the already mentioned rule “first select using C and then order using P ”.

Dubois and Prade [10] define a bipolar query as a set of pairs (C_i, P_i) of, respectively, negative and positive conditions imposed on values of selected attributes $\{A_i\}_{i=1,k}$. These conditions may be identified with fuzzy sets defined in the domain of a given attribute. Some of the conditions/sets C_i may be equal to the domain of an attribute A_i , i.e., for a given attribute there is no negative condition. On the other hand, some of the conditions/sets P_i may be empty, i.e., there is no positive condition for a given attribute. These pairs of conditions are combined to yield overall conditions C and P as follows:

$$(C, P) = (\times_i C_i, +_i P_i)$$

where $\times_i C_i = C_1 \times C_2 \times \dots \times C_k$, $+_i P_i = (P_1^c \times P_2^c \times \dots \times P_k^c)^c$ and X^c is a complement of the set X . Thus, the overall negative condition is obtained via the conjunction of all negative conditions concerning particular attributes while the overall positive condition is obtained via the disjunction of all positive conditions concerning particular attributes. In case of this, a more evident, equivalent formula for the pair of overall conditions is:

$$(C(t), P(t)) = (\min_i C_i(t), \max_i P_i(t))$$

A pair of the combined conditions (C, P) is then used according to the aforementioned principle “first select using C and then order using P ”. This principle is implemented via the use of the lexicographic order \preceq of tuples against the bipolar query, i.e., $t_1 \preceq t_2 \iff (C(t_1) < C(t_2)) \vee ((C(t_1) = C(t_2)) \wedge (P(t_1) \leq P(t_2)))$.

Dubois and Prade [10] consider also non-Boolean combinations of the set of positive conditions P_i . An evaluation of a bipolar query proceeds then as follows. Each tuple t is represented by a vector: $(C(t), P_{\sigma(1)}(t), \dots, P_{\sigma(n)}(t))$ where σ is a permutation of the positive conditions P_i such that $P_{\sigma(1)}(t) \geq \dots \geq P_{\sigma(n)}(t)$. Then, the lexicographic order of these vectors is used to rank-order the tuples. Thus, the *leximax* operator (cf., e.g., Dubois, Fargier and Prade [8]) is used here with respect to the positive conditions.

There is also another line of research pursued by Dubois and Prade [12, 11] which aims at providing a formal, possibility theory based framework for dealing with bipolar queries. Namely, two possibility distributions π and δ are assumed to represent query conditions (the user's preferences). The former corresponds to the negative condition, i.e., $\pi(t) = 1$ and $\pi(t) = 0$ mean, respectively, that a tuple t is totally acceptable and totally unacceptable, while the intermediate values of $\pi(t)$ express an intermediate degree of acceptability. The latter possibility distribution δ represents the positive condition: $\delta(t) = 1$ denotes the maximum degree of preference (desirability) of t but $\delta(t) = 0$ means merely that t is not specifically preferred. Both types of conditions are then syntactically represented in the framework of *possibilistic logic* (cf. Dubois [11]). The possibility distributions π and δ are generated by sets of formulas defining the negative and positive conditions, accompanied by the constraints on the minimum value of the *necessity* and *guaranteed possibility* measures, respectively.

The bipolar queries may be also seen a special case of queries which employ the concept of a *non-dominance relation*, which has been deeply studied in the context of decision making for many years. Such a general class of queries has been proposed recently, for the crisp case, by Chomicki [6] under the name of *queries with preferences*. In this approach a new relational algebra operator, called *winnow*, is introduced. This unary operator selects from a set of tuples those which are *non-dominated* with respect to a given *preference relation*, a binary relation defined on the set of tuples. A bipolar query may then be obtained using a proper combination of the *select* operator with the *winnow* operator. The negative conditions define the select operator while the positive conditions are expressed by the preference relation. Thus the new *winnow* operator may be easily combined with the traditional relational algebra operators. In the crisp case, similarly to the case of queries studied by Lacroix and Lavency [16], this combination is quite straightforward, somehow still corresponding to the "first select and then order" strategy, mentioned earlier. However in the fuzzy case such an approach becomes problematic and some special measures are needed. A possible approach, both in the case of bipolar queries in the sense of Lacroix and Lavency and in a more general context of Chomicki's query with preferences, is presented in the next section.

Recently, some attempts were undertaken to include the support for bipolar queries in the well-known fuzzy querying languages. For example, in Lietard [17] a preliminary approach related to the SQLf language (cf. Bosc [5]) is reported.

3 Fuzziness and bipolar queries: an approach

Here we follow the Lacroix and Lavency [16] original approach to bipolar queries and extend it to the case of fuzzy conditions. Moreover, we also propose a fuzzy version of the *winnow* operator and show its relation to "fuzzy" bipolar queries.

3.1 Fuzzification of the Lacroix and Lavency approach

We start with the concept of a bipolar query exemplified by (1) and formalized by (2) and (3). In a more realistic case the user will prefer to express the conditions in a query like (1) using fuzzy predicates what may result in the following query:

“Find a *cheap* house *and possibly* located *near* a railway station (5)

Let us remind that, according to the original interpretation of Lacroix and Lavency, (5) should be understood in such a way that we are looking for a house that:

- has to be *cheap*,
- if there is a cheap house near the railway station then other, just cheap houses are of no interest.

Notice that now the strategy of the bipolar query evaluation “first select using negative condition (here: cheap) and then order using the positive condition (here: near the station)” cannot be directly applied. Namely, it is not any longer clear what it should mean that a tuple (house) satisfies the negative condition (is cheap) as the satisfaction of this condition is now a matter of the degree. Let us illustrate the problem on a simple example. Let there be a house $H1$ definitely cheap (to a degree 1), but rather away from the station (near to a degree 0.2) and another house $H2$, still cheap but not that much as house $H1$ (to a degree 0.9), but located quite close to the station (to a degree 0.9). Now there is a question which of these two houses should belong to the answer to the query (5)? The “first select then order” strategy could be now implemented by the lexicographic order on the vectors of the satisfaction degrees representing both houses. For house $H1$ we have a vector $[1.0, 0.2]$ and for $H2$ a vector $[0.9, 0.9]$. Thus the lexicographic order indicates $H1$ as better than $H2$ what may be questionable, at least in certain scenarios. Moreover, the lexicographic order (nor any other order) does not give us any scalar measure of query matching by particular tuples, thus it does not help much in deciding which tuples should really form the answer to the query in question, what may be important in some applications.

Looking for a consistent solution to this problem we start with the formula (3) and interpret it in terms of fuzzy logic. Firstly, let us rewrite (3) using standard fuzzy counterparts of the logical connectives appearing there. Moreover, we will write it as a formula of the membership function of the resulting fuzzy set $ans(C, P, T)$ of tuples forming the answer to the bipolar query (C, P) against a set of tuples T as:

$$\mu_{ans(C,P,T)}(t) = \min(C(t), \max_{s \in T} (1 - \min(C(s), P(s)), P(t))) \quad (6)$$

Symbol T appears in $ans(C, P, T)$ to emphasize that the membership degree (matching degree) of a tuple t depends not only on this tuple itself and on the conditions C and P but also on the current whole set of tuples T – according to the semantics of bipolar queries in the sense of Lacroix and Lavency.

The *matching degree* of a tuple against a bipolar query is meant as the truth value of the formula (3), computed in the framework of fuzzy (multivalued) logic using

right-hand side of the formula (6). Thus, the evaluation of a bipolar query produces a fuzzy set of tuples, where the membership function value for a tuple t corresponds to the matching degree of this tuple against the query. The answer to a bipolar query is then a list of the tuples, non-increasingly ordered according to their membership degree.

In the formula (6) the \min , \max and $1 - x$ operators are used to model the connectives of conjunction, disjunction and negation, respectively. Moreover, the implication connective \Rightarrow is modelled by the Kleene-Dienes implication operator (cf., e.g., [13]) and the existential quantifier \exists is modelled via the maximum operator. We discuss the alternative choices of these operator later in this section.

The formula (6) has been proposed by Yager [21, 20, 22] for an aggregation operator in the context of the multicriteria decision making for the case of so-called *possibilistically qualified criteria*. Yager [22] intuitively characterizes a possibilistically qualified criterion as such which should be satisfied unless it interferes with satisfaction of other criteria. This is in fact the essence of bipolar queries in the sense advocated here. This concept was also applied by Bordogna and Pasi [2] for the information retrieval task.

In fact Dubois and Prade [10] considered a similar formula too. However their version of (6) employs an arbitrary parameter (instead of $\max_{s \in T} \min(C(s), P(s))$) in (6) what makes the results obtained for a certain specific range of values $(C(t), P(t))$ difficult to justify. In the current approach this expression has a meaningful interpretation providing some justification for this behavior; cf. Zadrożny [23] for details.

Now let us look at the formula (6) again. This is definitely only one of possible ways to “fuzzify” the original formula (3) proposed by Lacroix and Lavency [16]. In particular, different interpretations of the conjunction and implication connectives may be employed. Moreover, also the disjunction connective may be seen as related to the existential quantifier and its different possible forms also should be taken into account. Basically, various t -norms, t -conorms and implication operators (cf., e.g., [13]) may be assumed to model corresponding logical connectives in the framework of fuzzy logic based interpretation of formula (3).

In particular one may consider so-called De Morgan Triplets (\wedge, \vee, \neg) that comprise a t -norm operator \wedge , a t -conorm operator \vee and a negation operator \neg , where $\neg(x \vee y) = \neg x \wedge \neg y$ holds. The following three De Morgan Triplets play the most important role in fuzzy logic (cf., e.g., Fodor and Roubens [13] for a justification) $(\wedge_{min}, \vee_{max}, \neg)$, $(\wedge_{\Pi}, \vee_{\Pi}, \neg)$, (\wedge_W, \vee_W, \neg) , where the particular t -norms and t -conorms are defined as follows

$$\begin{array}{ll}
 x \wedge_{min} y = \min(x, y) & \textit{minimum} \\
 x \wedge_{\Pi} y = x \cdot y & \textit{product} \\
 x \wedge_W y = \max(0, x + y - 1) & \textit{Łukasiewicz } t\text{-norm} \\
 x \vee_{max} y = \max(x, y) & \textit{maximum} \\
 x \vee_{\Pi} y = x + y - x \cdot y & \textit{probabilistic sum} \\
 x \vee_W y = \min(1, x + y) & \textit{Łukasiewicz } t\text{-conorm}
 \end{array}$$

The negation operator \neg in case of all the above De Morgan Triplets is defined as:
 $\neg x = 1 - x$.

Due to their associativity, both t -norms and t -conorms may be employed as the m -ary operators, i.e., it is well defined what $x \wedge y \wedge \dots$ and $x \vee y \vee \dots$ mean.

Usually the general and existential quantifiers are identified in fuzzy logic, for the case of a finite universe, with the maximum and minimum operators, respectively. Namely, the following identities are employed: $\text{truth}(\forall x A(x)) = \min_x \mu_A(x)$ and $\text{truth}(\exists x A(x)) = \max_x \mu_A(x)$. As soon as we consider the use of other t -norms and t -conorms to “fuzzify” formula (3) it is reasonable to look for a consistency via the use of an appropriate t -quantifiers and s -quantifiers; cf., e.g., [18]. Thus we adopt the following definitions:

$$\text{truth}(\forall x A(x)) = \mu_A(a_1) \wedge \mu_A(a_2) \wedge \dots \wedge \mu_A(a_m) \quad (7)$$

$$\text{truth}(\exists x A(x)) = \mu_A(a_1) \vee \mu_A(a_2) \vee \dots \vee \mu_A(a_m) \quad (8)$$

The particular t - and s -quantifiers will be denoted by the \forall and \exists symbol with a subscript indicating the underlying t - or s -norm. For example, \exists_{\max} denotes the “standard” fuzzy existential quantifier which is obtained when the maximum t -conorm is used.

There are two most popular ways of deriving an implication operator with respect to a given De Morgan Triple (\wedge, \vee, \neg) , namely so-called S -implications and R -implications defined as follows:

$$R\text{-implication: } x \rightarrow y = \sup\{z : x \wedge z \leq y\} \quad (9)$$

$$S\text{-implication: } x \rightarrow y = \neg x \vee y \quad (10)$$

Thus, for the particular De Morgan Triplets one obtains the following R -implication operators:

$$\text{Gödel's implication} \quad x \rightarrow_{R\text{-min}} y = \begin{cases} 1 & \text{for } x \leq y \\ y & \text{for } x > y \end{cases}$$

$$\text{Goguen's implication} \quad x \rightarrow_{R\text{-II}} y = \begin{cases} 1 & \text{for } x = 0 \\ \min\{1, \frac{y}{x}\} & \text{for } x \neq 0 \end{cases}$$

$$\text{Łukasiewicz' implication} \quad x \rightarrow_{R\text{-W}} y = \min(1 - x + y, 1)$$

and the following S -implication operators:

$$\text{Kleene-Dienes' implication} \quad x \rightarrow_{S\text{-max}} y = \max(1 - x, y)$$

$$\text{Reichenbach's implication} \quad x \rightarrow_{S\text{-II}} y = 1 - x + x \cdot y$$

The S -implication operator $\rightarrow_{S\text{-W}}$ is identical with $\rightarrow_{R\text{-W}}$.

Let us write a fuzzy version of (3) in a more general form than (6), where a specific t -norm and other related operators were assumed:

$$\mu_{\text{ans}(C,P,T)}(t) = C(t) \wedge (\exists_{\vee s \in T} (C(s) \wedge P(s)) \rightarrow P(t)) \quad (11)$$

In order to simplify the notation let us fix C , P and T in the formula (6) and denote its version for a given De Morgan Triple, its related R or S implication and a corresponding existential quantifier as, respectively, $\gamma_{\wedge,R}$ and $\gamma_{\wedge,S}$. Thus, for example, $\gamma_{\min,S}(t) = \mu_{ans(C,P,T)}(t)$ denotes the original version of the formula (6).

In Table 1 various emerging interpretations of the formula (6) are shown.

Table 1 Right-hand side of formula (6) for different interpretations of the logical connectives

$\gamma_{\wedge,\cdot}$	Resulting form of the formula (6)
$\gamma_{\min,S}$	$\min(C(t), \max(1 - \max_{s \in T} \min(C(s), P(s)), P(t)))$
$\gamma_{\min,R}$	$\begin{cases} C(t) & \text{if } \max_{s \in T} \min(C(s), P(s)) \leq P(t) \\ \min(C(t), P(t)) & \text{otherwise} \end{cases}$
$\gamma_{\Pi,S}$	$C(x) \cdot (\prod_i (1 - C(y_i) \cdot P(y_i)) \cdot (1 - P(x)) + P(x))$
$\gamma_{\Pi,R}$	$\begin{cases} C(t) & \text{if } \exists_{\Pi} (C(s_i) \cdot P(s_i)) = 0 \\ C(t) \cdot \min(\frac{P(t)}{\exists_{\Pi} (C(s_i) \cdot P(s_i))}, 1) & \text{otherwise} \end{cases}$
γ_W	$C(t) \wedge_W (\exists_W (C(s) \wedge_W P(s)) \rightarrow_W P(t))$

The issue of how to appropriately model the logical connectives (including the existential quantifier) in formula (3) may be basically approached in two ways. First, one may look for some axioms that should characterize the behavior of this formula and try to check which operators modelling the connectives provide for the expected behavior (in what follows we will refer to the operators modelling the logical connectives as *logical operators*). The second, more modest, approach consists in studying the properties of (11) under different choices of logical operators. In the framework of the former approach we discuss below a property which should seem to be reasonable and which eliminates a class of the logical operators. We also show a few properties in the vein of the second approach.

The property which we impose on any bipolar query evaluation scheme, implied by a choice of logical operators in (11), may be – in a bit informal way – expressed as follows. It should not be the case that a tuple t satisfying very well the negative condition (let $C(t) = 1$) and not satisfying at all the positive condition (i.e., $P(t) = 0$) is indicated by the evaluation scheme as inferior to any tuple s satisfying both conditions to an infinitely small, but greater than 0, extent (i.e., $C(s) = P(s) = \varepsilon$, $\varepsilon \in [0, 1]$ and ε is a very small number).

It may be easily shown that for a t -norm without zero divisors (exemplified by the minimum and product t -norms, \wedge_{\min} and \wedge_{Π}) and related R -implication operator this property does not hold. This may be shown as follows. Let us assume that

$$\exists_{\vee} s (C(s) \wedge P(s)) > 0$$

Then $\exists_{\vee} u (C(u) \wedge P(u)) \rightarrow_{\wedge,R} P(t)$ is equal 0 for t such that $C(t) = 1$ and $P(t) = 0$, while it is greater than 0 for s such that $C(s) = P(s) = \varepsilon$. Then the right-hand side

of the formula (11) is equal 0 for t and greater than 0 for S (the t -norm is assumed to have no zero divisors) and thus a tuple s emerges as preferred to t .

In the framework of the second approach to the choice of the logical operators, mentioned above, we have shown some properties in Zadrozny and Kacprzyk [26]. First, we note two general properties, valid for any combination of a t -norm, t -conorm and S -implication or R -implication. Namely, if there exists a tuple t such that $C(t) = 1$ and $P(t) = 1$, then (11) turns into $C(t) \wedge P(t)$, where \wedge is represented by a given t -norm. Thus, whatever the choice of the logical operators is, a characteristic feature of bipolar queries is preserved: if there is a tuple satisfying both the required and preferred conditions, then (11) turns into a conjunction. On the other hand, if for a tuple $t \in T$ $P(t) = 1$, then the formula (11) turns into $C(t)$. This property is implied by the characteristic features of the t -norm and implication operators: $x \rightarrow 1 = 1$ and $x \wedge 1 = x$, for any choice of operators \wedge and \rightarrow (cf., e.g., [13]). Thus, if a tuple fully satisfies the positive condition P , then its overall matching degree is equal to its satisfaction of the negative condition C .

The most important question concerning the choice of the logical operators is the following: does this choice influence the resulting order of the tuples in the answer to a bipolar query ?

Contributing to an answer to this question we notice the following properties. First, as to the property concerning the choice of the existential quantifier model, the usual fuzzy existential quantifier \exists_{max} : (A) yields greater or equal matching degrees, and (B) may change the resulting ordering of the tuples, when used instead of the \exists_{Π} or \exists_W quantifiers. Property A is a direct consequence of the fact that the maximum operator is the smallest of all t -conorms and that implication operators and t -norms are monotonic. Property B may be shown on an example; cf. Zadrozny and Kacprzyk [26].

Second, we have shown that, in general, the choice between an S -implication and an R -implication, keeping all other logical operators fixed, may change the order of the tuples.

On the other hand, we have pointed out a specific case where such a change does not occur. Namely, for the (t_{min}, s_{max}, N) De Morgan Triplet and for tuples t verifying the conditions: $(P(t) \geq \exists CP)$ or $((P(t) \leq \exists CP) \text{ and } (P(t) \geq 1 - \exists CP))$ it holds that: (A) $\gamma_{min,R}(C, P, t, T) \geq \gamma_{min,S}(C, P, t, T)$, and (B) replacing the R -implication with S -implication or vice-versa preserves the resulting order of the tuples ($\exists CP$ denotes here the truth value of the formula $\exists_{\vee} s \in T C(s) \wedge P(s)$, which is a part of (11)).

Thus for the tuples t satisfying the specified condition their resulting order does not depend on the choice between the S -implication and the R -implication. The “troublesome” tuples may appear both for high and low values of $\exists CP$. However, in the former case these are less interesting as for them $P(t)$ is small while there are tuples well satisfying both C and P (as $\exists CP$ is high).

4 Queries with preferences and bipolar queries

The concept of a bipolar query we adopt here may be interpreted as an extension of the original idea of Lacroix and Lavency [16] in the framework of *fuzzy logic*. Here, we study its relation to the concept of a query with preferences, introduced by Chomicki [6, 7]. This concept may be conveniently presented in terms of a new operator of the relational algebra, called *winnnow*, which is proposed in Chomicki [6]. This is an unary operator which selects from a set of tuples those which are *non-dominated* with respect to a given preference relation. Chomicki defines this operator for the crisp case only, i.e., where preference relations and sets of tuples are crisp sets. We propose a fuzzy version of the *winnnow* operator and show its relation to bipolar queries.

The *winnnow* operator is defined with respect to a *preference relation*. In [6, 7] this is any binary relation R defined on the set of tuples T : $R \subseteq T \times T$. If two tuples $t, s \in T$ are in relation R , i.e., $R(t, s)$, then it is said that tuple t *dominates* tuple s with respect to the relation R .

Let T be a set of tuples and R a preference relation defined on T . Then the *winnnow* operator ω_R is defined as follows

$$\omega_R(T) = \{t \in T : \neg \exists s \in T R(s, t)\} \quad (12)$$

Thus, for a given set of tuples it yields a subset of the *non-dominated* tuples with respect to R .

A relational algebra query employing the *winnnow* operator is referred to as a *query with preferences*. It may be easily shown (cf, Chomicki [6]) that the *winnnow* operator may be expressed as a combination of the standard classical relational algebra operators. However, distinguishing the *winnnow* operator makes it possible to study its behavior. Furthermore, some specialized methods of its execution may be conceived taking into account the optimal plans of the execution of the whole query.

The concept of the *winnnow* operator may be illustrated with the following simple example. Let us consider a database of a real-estate agency with a table `HOUSES` describing the details of particular real-estate properties offered by the agency (each house is represented by a tuple). The schema of the relation `HOUSES` contains, among other, the attributes `city` and `price`. Let us assume that we are interested in the list of the *cheapest* houses in each city. Then the preference relation should be defined as follows

$$R(t, s) \Leftrightarrow (t.\text{city} = s.\text{city}) \wedge (t.\text{price} < s.\text{price})$$

where $t.A$ denotes the value of attribute A (e.g., `price`) at a tuple t . Then the *winnnow* operator $\omega_R(\text{HOUSES})$ will select the houses that are sought (here a database table, such as `HOUSES`, is treated as a set of tuples). Indeed, according to the definition of the *winnnow* operator, we will get as an answer a set of houses, which are non-dominated with respect to R , i.e., for which there is no other house in the same city which has a lower price.

A bipolar query with *crisp* conditions: the negative C and positive P may be expressed using the *winnow* operator in the following way. Let us show this first on an example of a bipolar query given in (1), i.e. “Find a house cheaper than USD 250,000 *and possibly* located not more than two blocks from a railway station”. The preference R relation should be now defined as follows:

$$R(t,s) \Leftrightarrow (t.to_station \leq 2) \wedge (s.to_station > 2)$$

assuming that the `to_station` attribute characterizes each house, indicating how many blocks away it is located from a closest railway station. Then, the following relational algebra query with the *winnow* operator yields the required results

$$\omega_R(\sigma_{price \leq 250000}(HOUSES))$$

where σ_ϕ is the classical *selection* operator that selects from a set of tuples those for which condition ϕ holds.

This query preserves the characteristic property of bipolar queries, discussed earlier, i.e., if there are houses cheaper than USD 250,000 and located closer than two blocks from the station then only them will be selected (houses satisfying only the negative condition will be ignored). Otherwise, all houses satisfying the negative condition will be selected, if such exist.

A general scheme for translating a bipolar query characterized by a pair of negative and positive conditions (C, P) into its corresponding query with preferences is the following. The preference relation R is defined as

$$R(t,s) \Leftrightarrow P(t) \wedge \neg P(s) \tag{13}$$

and then the overall query with preferences takes the form:

$$\omega_R(\sigma_C(T))$$

Now we propose a fuzzy counterpart of the *winnow* operator, which also will make it possible to express (fuzzy) bipolar queries. We have to take into account that:

- R should be assumed to be a *fuzzy preference relation*,
- a fuzzy counterpart of the *non-dominance* concept has to be employed,
- the set of tuples T should also be assumed to be a *fuzzy set*.

In order to address the above requirements it is convenient to use the concept of a *fuzzy choice function* (cf. Świtalski [19]). In this approach the set of non-dominated elements with respect to a fuzzy preference relation may be conveniently expressed. Let us start with a concept of a crisp set $R^-(s)$, defined as follows:

$$R^-(s) = \{u \in T : R(s,u)\} \tag{14}$$

and gathering all tuples dominated by a tuple s with respect to a crisp preference relation R . Then $N(T,R)$, defined as follows:

$$N(T, R) = T \cap \bigcap_{s \in T} \overline{R^-(s)} \quad (15)$$

denotes the set of all non-dominated tuples of a crisp set T with respect to a crisp preference relation R , while \overline{A} denotes the complement of the set A . The above definition is a bit more complicated than necessary (cf. the intersection with the set T), however this is useful for deriving its fuzzy counterpart. For a further “fuzzification” it is convenient to rewrite (15) as a predicate calculus formula

$$N(T, R)(t) \Leftrightarrow T(t) \wedge \forall_{s \in T} \neg R^-(s)(t) \quad (16)$$

where particular predicates are denoted with the same symbols as corresponding sets (in particular, $R^-(s)$ denotes a predicate corresponding to a set (14) defined for a tuple s).

Using (15) we can define the *winnow* operator in the following way, equivalent to (12)

$$\omega_R(T) = N(T, R) \quad (17)$$

Now let us adapt (17) to the case of fuzzy preference relation R . A *fuzzy preference relation* on a crisp set of tuples T is any *fuzzy* binary relation \tilde{R} , $\tilde{R} \in \mathcal{F}(T \times T)$, where $\mathcal{F}(X)$ denotes the set of all fuzzy sets defined over the universe X . It will be identified with its membership function $\mu_{\tilde{R}}$.

As soon as the preference relation becomes fuzzy, then also the dominance (and non-dominance) naturally becomes a matter of degree. Thus we define a fuzzy set of tuples non-dominated with respect to a fuzzy preference relation \tilde{R} , using the formulas (14)-(15) and interpreting the set operations of the intersection and complement appearing thereof as standard operations on fuzzy sets. We start with a fuzzy counterpart of the set (14), defining the membership function of the fuzzy set $\tilde{R}^-(s)$ of tuples dominated (to a degree) by a tuple s with respect to the fuzzy preference relation \tilde{R} :

$$\mu_{\tilde{R}^-(s)}(u) = \mu_{\tilde{R}}(s, u) \quad (18)$$

Next let us rewrite (16), replacing a preference relation R with a fuzzy preference relation \tilde{R} and replacing R^- with \tilde{R}^- , according to (18):

$$N(T, \tilde{R})(t) \Leftrightarrow T(t) \wedge \forall_{s \in T} \neg \tilde{R}^-(s, t) \quad (19)$$

We still have to take into account that the set T (and corresponding to it predicate) is, in general, fuzzy. Thus, we denote it as \tilde{T} and replace the restricted quantifier $\forall_{s \in T}$ in (19) with an equivalent non-restricted form obtaining:

$$N(\tilde{T}, \tilde{R})(t) \Leftrightarrow \tilde{T}(t) \wedge \forall_s (\tilde{T}(s) \rightarrow \neg \tilde{R}(s, t)) \quad (20)$$

Finally, we can define a fuzzy counterpart of the *winnow* operator in the following way. Let \tilde{T} be a fuzzy set of tuples and \tilde{R} a fuzzy preference relation, both defined on the same set of tuples T . Then, the *fuzzy winnow operator* $\omega_{\tilde{R}}$ is defined as:

$$\omega_{\tilde{R}}(\tilde{T})(t) = N(\tilde{T}, \tilde{R})(t) \quad t \in T \quad (21)$$

where the fuzzy predicate $N(\tilde{T}, \tilde{R})$ is determined by (20), and $\omega_{\tilde{R}}(\tilde{T})(t)$ denotes the value of the fuzzy membership function of the set of tuples defined by $\omega_{\tilde{R}}(\tilde{T})$ for a tuple t .

Again, as in the case of fuzzy bipolar queries, one may study the effect of the choice of various logical operators to model logical connectives in the formula (20). We leave a general study of this issue for further research. Now we will just show how a bipolar query may be expressed using the concept of the fuzzy *winnow* operator.

Let us consider a bipolar query defined by a pair of fuzzy conditions (C, P) . These conditions will be identified with fuzzy predicates, denoted with the same symbols. Let \tilde{R} be a fuzzy preference relation of the following form (cf. (13))

$$\tilde{R}(t, s) \Leftrightarrow P(t) \wedge \neg P(s) \quad (22)$$

Then, the bipolar query may be expressed as the following combination of the selection and fuzzy *winnow* operators:

$$\omega_{\tilde{R}}(\sigma_C(T)) = N(C(T), \tilde{R}) \quad (23)$$

where $C(T)$ is a fuzzy set of the elements of T satisfying (to a degree) the condition C , i.e., $\mu_{C(T)}(t) = C(t)$. Using (20) we can define the predicate (set) $N(C(T), \tilde{R})$ in (23) as follows

$$N(C(T), \tilde{R})(t) \Leftrightarrow C(t) \wedge \forall_s (C(s) \rightarrow \neg(P(s) \wedge \neg P(t))) \quad (24)$$

Note that the selection operator σ_C in (23) may also be applied to a fuzzy set of tuples T , which may be convenient if the set of tuples T is a result of another fuzzy query.

In Zadrozny and Kacprzyk [25] we show that for the conjunction, negation and implication connectives in (24) modelled by the operators of the minimum, $n(x) = 1 - x$ and the Kleene-Dienes implication, respectively, the fuzzy set of tuples obtained using (23) is identical with the fuzzy set defined by (6).

5 Concluding remarks

We discuss the concept of bipolar queries. We show its origin in the work of Lacroix and Lavency [16] and briefly review some selected relevant approaches recently proposed in the literature. In particular we point out two main lines of research. One focuses on the formal representation within some well established theories and the question of a meaningful combinations of multiple conditions. Another one is concerned first of all with the study of some appropriate ways to aggregate negative (required) and positive (desired) conditions. We follow the second line of research and show the relation of this concept, from this point of view, with other approaches, both concerning database querying (exemplified by Chomicki [6] as well as other

domains (exemplified by Yager [21]). In the former case we offer a fuzzy counterpart of a new relational algebra operator *winnow*.

As in many cases of fuzzy logic applications the resulting concepts exhibit some arbitrariness with respect to a choice of the way logical connectives should be modelled. We contribute with some preliminary conclusions here but a further research is definitely needed.

References

1. Benferhat, S., Dubois, D., Kaci, S., Prade, H.: Bipolar possibility theory in preference modeling: Representation, fusion and optimal solutions. *Information Fusion* **7**(1), 135–150 (2006)
2. Bordogna, G., Pasi, G.: Linguistic aggregation operators of selection criteria in fuzzy information retrieval. *International Journal of Intelligent Systems* **10**(2), 233–248 (1995)
3. Bosc, P., Pivert, O.: Discriminated answers and databases: fuzzy sets as a unifying expression means. In: *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 745–752. San Diego, USA (1992)
4. Bosc, P., Pivert, O.: An approach for a hierarchical aggregation of fuzzy predicates. In: *Proceedings of the Second IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'93)*, pp. 1231–1236. San Francisco, USA (1993)
5. Bosc, P., Pivert, O.: SQLf: A relational database language for fuzzy querying. *IEEE Transactions on Fuzzy Systems* **3**(1), 1–17 (1995)
6. Chomicki, J.: Querying with intrinsic preferences. *Lecture Notes in Computer Science* **2287**, 34–51 (2002)
7. Chomicki, J.: Preference formulas in relational queries. *ACM Transactions on Database Systems* **28**(4), 427–466 (2003)
8. Dubois, D., Fargier, H., Prade, H.: Refinement of the maximin approach to decision-making in fuzzy environment. *Fuzzy Sets and Systems* (81), 103–122 (1996)
9. Dubois, D., Prade, H.: Using fuzzy sets in flexible querying: why and how? In: T. Andreasen, H. Christiansen, H. Larsen (eds.) *Flexible Query Answering Systems*, pp. 45–60. Kluwer Academic Publishers (1997)
10. Dubois, D., Prade, H.: Bipolarity in flexible querying. In: Andreasen T. et al. (ed.) *FQAS 2002, LNAI*, vol. 2522, pp. 174–182. Springer-Verlag, Berlin, Heidelberg (2002)
11. Dubois, D., Prade, H.: Handling bipolar queries in fuzzy information processing. In: Galindo [14], pp. 97–114
12. Dubois, D., Prade, H.: An introduction to bipolar representations of information and preference. *International Journal of Intelligent Systems* **23**(8), 866–877 (2008)
13. Fodor, J., Roubens, M.: *Fuzzy Preference Modelling and Multicriteria Decision Support. Series D: System Theory, Knowledge Engineering and Problem Solving*. Kluwer Academic Publishers (1994)
14. Galindo, J. (ed.): *Handbook of Research on Fuzzy Information Processing in Databases*. Information Science Reference, New York, USA (2008)
15. Kacprzyk, J., Zadrozny, S.: Computing with words in intelligent database querying: standalone and internet-based applications. *Information Sciences* **134**(1-4), 71–109 (2001)
16. Lacroix, M., Lavency, P.: Preferences: Putting more knowledge into queries. In: *Proceedings of the 13 International Conference on Very Large Databases*, pp. 217–225. Brighton, UK (1987)
17. Lietard, L., Rocacher, D., Tbahriti, S.E.: Towards an extended SQLf: Bipolar query language with preferences. *International Journal of Applied Mathematics and Computer Sciences* **4**(1), 58–63 (2008)

18. Mesiar, R., Thiele, H.: On T-Quantifiers and S-Quantifiers. In: V. Novak, I. Perfilieva (eds.) *Discovering the World with Fuzzy Logic*, pp. 310–326. Physica-Verlag, Heidelberg New York (2000)
19. Świtalski, Z.: Choice functions associated with fuzzy preference relations. In: Kacprzyk J., Roubens M. (eds.) *Non-Conventional Preference Relations in Decision Making*, pp. 106–118. Springer-Verlag, Berlin (1988)
20. Yager, R.: Fuzzy sets and approximate reasoning in decision and control. In: *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 415–428. San Diego, USA (1992)
21. Yager, R.: Higher structures in multi-criteria decision making. *International Journal of Man-Machine Studies* **36**, 553–570 (1992)
22. Yager, R.: Fuzzy logic in the formulation of decision functions from linguistic specifications. *Kybernetes* **25**(4), 119–130 (1996)
23. Zadrozny, S.: Bipolar queries revisited. In: Torra V., Narukawa Y., Miyamoto S. (eds.) *Modelling Decisions for Artificial Intelligence (MDAI 2005)*, *LNAI*, vol. 3558. Springer-Verlag, Berlin, Heidelberg (2005)
24. Zadrozny, S., De Tre, G., De Caluwe, R., Kacprzyk, J.: An overview of fuzzy approaches to flexible database querying. In: Galindo [14], pp. 34–53
25. Zadrozny, S., Kacprzyk, J.: Bipolar queries and queries with preferences. In: *Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA'06)*, pp. 415–419. IEEE Computer Society, Krakow, Poland (2006)
26. Zadrozny, S., Kacprzyk, J.: Bipolar queries using various interpretations of logical connectives. In: *Foundations of Fuzzy Logic and Soft Computing, Lecture Notes in Computer Science*, pp. 181–190. Springer (2007)