Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning
Main Track - Short paper

A Many-valued Logic for Lexicographic Preference Representation

Angelos Charalambidis , **George Papadimitriou** , **Panos Rondogiannis** , **Antonis Troumpoukis**

Department of Informatics and Telecommunications,
National and Kapodistrian University of Athens
{a.charalambidis,gspapajim,prondo,antru}@di.uoa.gr

Abstract

We introduce *lexicographic logic*, an extension of propositional logic that can represent a variety of preferences, most notably lexicographic ones. The proposed logic supports a simple new connective whose semantics can be defined in terms of finite lists of truth values. We demonstrate that, despite the well-known theoretical limitations that pose barriers to the quantitative representation of lexicographic preferences, there exists a subset of the rational numbers over which the proposed new connective can be naturally defined. Lexicographic logic can be used to define in a simple way some well-known preferential operators, like "A and if possible B", and "A or failing that B". We argue that the new logic is an effective formalism for ranking query results according to the satisfaction level of user preferences.

1 Introduction

Many formalisms have been developed for representing preferences, both in artificial intelligence (Domshlak et al. 2011) as-well-as in databases (Stefanidis, Koutrika, and Pitoura 2011). Of particular interest are the logical approaches in which the specification of preferences is performed using operators that implicitly manipulate the underlying preference values (Brewka, Benferhat, and Berre 2004; Brewka, Niemelä, and Truszczynski 2008; Agarwal and Wadge 2005; Lang 2009; Dubois and Prade 2013; Rondogiannis and Troumpoukis 2015; Charalambidis, Rondogiannis, and Troumpoukis 2018). Such formalisms are usually declarative, concise, and easy to understand.

In this paper, we develop *lexicographic logic*, a simple extension of classical propositional logic that can express a variety of preferences, most notably lexicographic ones. The proposed logic adds only one formation rule to the syntax of propositional logic: if ϕ_1 and ϕ_2 are formulas, then so is $(\phi_1 \gg \phi_2)$. The formula $(\phi_1 \gg \phi_2)$ can be read "ϕ_1 *has a lexicographic priority over* ϕ_2". The semantics of "$\gg$" can be defined in terms of finite lists of truth values of a three-valued logic. Actually, we demonstrate that such lists have a natural mapping to rational numbers in the interval $[-1, 1]$, and the meaning of "$\gg$" can also be understood as a function that maps pairs of rational numbers to rational numbers. Apart from its simplicity, an advantage of lexicographic logic is that it can be used to represent concisely other well-known preferential operators. The main contributions of the paper are the following:

- We define a novel, non-classical, propositional logic for expressing preferences. The primary connective of this logic expresses lexicographic priority, which is known to be non-trivial to specify from a quantitative point of view (Fishburn 1999). We demonstrate that the semantics of this new connective can be specified quantitatively as a function over rational numbers. The main theorem of the paper[1] (Theorem 2) asserts that "$\gg$" ensures strict monotonicity with respect to lexicographic comparison.
- We present the key properties of lexicographic logic and investigate its connections with other preferential operators that have been proposed in the literature. We demonstrate that some well-known such operators can be succinctly represented using simple formulas of lexicographic logic.

1.1 Motivation and Intuition

We introduce *lexicographic logic* as an extension of propositional logic, with the addition of a new connective "$\gg$" for describing lexicographic preferences. The following example illustrates the main ideas.

Example 1. Consider the specification of our preferences for buying a new car. The formula $(\texttt{electric} \gg \texttt{fast})$ means that if we buy a car that is both electric and fast, we will be completely satisfied; if we buy one that is electric but not fast, then we will not be entirely satisfied; if we buy a car that is only fast, then we will be dissatisfied (but not entirely); and if none of our preferences is satisfied, then this will be our least preferred state of affairs.

To specify the formal semantics of $(\phi_1 \gg \phi_2)$, we use a many-valued logic to express the different levels of preference satisfaction discussed above. In particular, if F, T denote the classical false and true values, we expect that:

$$(F \gg F) < (F \gg T) < (T \gg F) < (T \gg T)$$

where "$<$" can be read as "less preferred than". This suggests that we probably need a four-valued logic to express these four truth levels, which we could call "false", "less false", "less true", and "true". However, it turns out that four truth levels are not enough.

[1]The proofs of the main results have been omitted. An extended version that contains these proofs can be retrieved from http://arxiv.org/abs/2012.10940

To understand the difficulties of defining such a set, we need to become slightly more formal. Let us denote by $\mathbb{V}$ this (yet unknown) set and let "$<$" be the (yet undefined) ordering relation on $\mathbb{V}$. Let $u_1, u_2, v_1, v_2 \in \mathbb{V}$. We define the "lexicographically smaller" relation $<_L$ on $\mathbb{V} \times \mathbb{V}$, as:

$$(u_1, u_2) <_L (v_1, v_2) \text{ iff } (u_1 < v_1) \vee ((u_1 = v_1) \wedge (u_2 < v_2))$$

The meaning of "$\gg$" should be a function f that respects the $<_L$ ordering, ie., for all $u_1, u_2, v_1, v_2 \in \mathbb{V}$ it must hold:

$$(u_1, u_2) <_L (v_1, v_2) \text{ iff } f(u_1, u_2) < f(v_1, v_2)$$

One could view truth values as real numbers and attempt to define f as a function $f : (X, X) \rightarrow X$ where X is a subset of $\mathbb{R}$. For example, one could map the truth values F and T to the real numbers -1 and 1, and try to define "$\gg$" as a function $f : ([-1, 1], [-1, 1]) \rightarrow [-1, 1]$. However, there is a well-known obstacle to such an approach, described by the following folk theorem (see, (Fishburn 1999, p. 363)).

Theorem 1. *There does not exist a function $f : (\mathbb{R}, \mathbb{R}) \rightarrow \mathbb{R}$ such that $(u_1, u_2) <_L (v_1, v_2)$ iff $f(u_1, u_2) < f(v_1, v_2)$.*

To understand how we bypass the above problem, we return to Example 1. Our first idea (which we will subsequently refine) is to express different levels of preferences using a truth domain whose elements are *lists* of classical truth values F and T. The semantics of the "$\gg$" operator can then be defined as the concatenation of such lists. In our example, if both `electric` and `fast` are true, we assign to the formula the value $[T, T]$; if `electric` is true and `fast` is false, we assign the value $[T, F]$; if `electric` is false and `fast` is true, we assign the value $[F, T]$; if both atoms are false, we assign the value $[F, F]$. These lists, when viewed as words compared lexicographically over the alphabet $\{F, T\}$, where $F < T$, express the four different levels of truth ("false", "less false", "less true", and "true") that we desire for this formula, namely: $[F, F] < [F, T] < [T, F] < [T, T]$.

In the following, for uniformity reasons, even the classical truth values F and T will be written in list form, namely $[F]$ and $[T]$. Notice also that we expect $[T, T]$ to be equal to $[T]$, because this is a situation where all our preferences are satisfied; similarly, we expect $[F, F]$ to be equal to $[F]$.

There is, however, a further refinement of the above scheme that is required. The basic complication that needs to be addressed is that the operator "$\gg$" should not be associative. This issue is illustrated by the following example.

Example 2. We claim that the two formulas `electric` $\gg$ (`fast` $\gg$ `blue`) and (`electric` $\gg$ `fast`) $\gg$ `blue` should not be semantically equivalent. To see this, consider a truth assignment that assigns to `electric` the value $[T]$, to `fast` the value $[F]$, and to `blue` the value $[F]$. Intuitively speaking, the first formula evaluates to $[T] \gg [F, F]$ while the second one to $[T, F] \gg [F]$. Comparing these two values, we see that in the former one, our primary requirement is fully satisfied (truth value $[T]$), while in the latter one, our primary requirement is only partially satisfied (truth value $[T, F]$). In other words, under this truth assignment, the first formula is "more satisfied" than the second one.

The above discussion suggests that the meaning of "$\gg$" should not be associative, and therefore it should not be defined as just list-concatenation (which is associative). To properly define the semantics of "$\gg$", we will use one extra truth value, namely 0. By prefixing each concatenation operation with a 0, formulas such as the ones that appear in the above example will be discriminated. In this way, different lists are created for different parenthesizations of an expression, and this ensures non-associativity. As we will demonstrate shortly (see Theorem 2) this simple operation ensures preservation of the lexicographic property. More specifically, we demonstrate that for all $u_1, u_2, v_1, v_2 \in \mathbb{V}$:

$$(u_1, u_2) <_L (v_1, v_2) \text{ iff } (u_1 \gg u_2) < (v_1 \gg v_2)$$

Therefore, our definition of "$\gg$" bypasses the restriction, by relying on a carefully selected truth domain.

2 Syntax and Semantics

The syntax of lexicographic logic extends that of propositional logic with a new formation rule.

Definition 1. Let $\mathcal{A}$ be a set of propositional atoms. The set of well-formed formulas of lexicographic logic is inductively defined as follows:

- Every element of $\mathcal{A}$ is a well-formed formula,
- If ϕ_1 and ϕ_2 are well-formed formulas, then $(\phi_1 \wedge \phi_2)$, $(\phi_1 \vee \phi_2)$, $(\neg \phi_1)$, and $(\phi_1 \gg \phi_2)$, are well-formed formulas.

We will omit the outermost parentheses from formulas and we will assume that "$\gg$" associates to the right.

The truth domain of lexicographic logic is denoted by $\mathbb{V}$, and consists of lists of the truth values F, 0, and T. By overloading notation, we will use the symbol "$\gg$" to also denote an operation on lists that corresponds to the meaning of the syntactic element "$\gg$". More formally:

Definition 2. Let u, v be lists of the elements F, T, and 0. We define:

$$(u \gg v) = \begin{cases} [F], & \text{if } u = v = [F] \\ [T], & \text{if } u = v = [T] \\ [0] \mathbin{+\!\!+} u \mathbin{+\!\!+} v, & \text{otherwise} \end{cases}$$

where $\mathbin{+\!\!+}$ is the list concatenation operation.

We now define the truth domain of lexicographic logic.

Definition 3. The set $\mathbb{V}$ of truth values is the set inductively defined as follows:

- $[F] \in \mathbb{V}$ and $[T] \in \mathbb{V}$.
- If $u, v \in \mathbb{V}$, then $(u \gg v) \in \mathbb{V}$.

Notice that due to Definition 2, lists of the form $[0, T, T]$ and $[0, F, F]$ are not allowed (because they are considered identical to $[T]$ and $[F]$ respectively).

Each element of $\mathbb{V}$ represents some degree of "true" or "false". To understand whether a given element is true or false, it suffices to look at its *sign*. We define $sign(v)$ to be the leftmost non-zero element of $v \in \mathbb{V}$. Therefore, $[0, F, 0, F, T]$ is a false value while $[0, T, F]$ a true one.

Given an arbitrary element $v \in \mathbb{V}$, we denote with $\overline{v}$ the list that results from v by inverting each F to T and each T to F. For example, $\overline{[0, F, 0, T, F]} = [0, T, 0, F, T]$. It is easy to establish that $\overline{v} \in \mathbb{V}$ for all $v \in \mathbb{V}$. As it turns out, no element of $\mathbb{V}$ is a proper prefix of another element. This property is crucial in establishing the main theorem of the paper (Theorem 2).

Lemma 1. *Let $u, v \in \mathbb{V}$. If u is a prefix of v, then $u = v$.*

By assuming the ordering $F < 0 < T$, any two elements of $\mathbb{V}$ can be compared lexicographically.

Definition 4. Let $u, v \in \mathbb{V}$ and assume $u = [u_1, \ldots, u_k]$ and $v = [v_1, \ldots, v_m]$. We will write $u < v$ if there exists $1 \leq r \leq \min\{k, m\}$ such that $u_1 = v_1, \ldots, u_{r-1} = v_{r-1}$ and $u_r < v_r$. We will write $u \leq v$ if either $u = v$ or $u < v$.

Due to Lemma 1, for any $u, v \in \mathbb{V}$, it will either be $u \leq v$ or $v \leq u$. It is easy to see that the lexicographic ordering $\leq$ of Definition 4 is a total ordering on $\mathbb{V}$.

We proceed with the definitions of the truth assignment and the meaning of a formula.

Definition 5. A truth assignment is a function from the set $\mathcal{A}$ of propositional atoms to the set $\mathbb{V}$ of truth values.

Definition 6. Let ϕ_1, ϕ_2, and ϕ be formulas and let I be a truth assignment. The meaning of a formula with respect to I is recursively defined as follows:

- $[\![\mathsf{p}]\!](I) = I(\mathsf{p})$, where $\mathsf{p} \in \mathcal{A}$
- $[\![(\phi_1 \wedge \phi_2)]\!](I) = \min\{[\![\phi_1]\!](I), [\![\phi_2]\!](I)\}$
- $[\![(\phi_1 \vee \phi_2)]\!](I) = \max\{[\![\phi_1]\!](I), [\![\phi_2]\!](I)\}$
- $[\![(\neg\phi)]\!](I) = \overline{[\![\phi]\!](I)}$
- $[\![(\phi_1 \gg \phi_2)]\!](I) = [\![\phi_1]\!](I) \gg [\![\phi_2]\!](I)$.

where $\min$ and $\max$ are defined w.r.t. the $\leq$ ordering.

Given a formula ϕ and a set of different truth assignments, we can find the most preferable truth assignments for ϕ by calculating the meaning of ϕ under these assignments, and comparing the results.

Definition 7. Let ϕ be a formula and consider truth assignments I_1, I_2. We will say that I_2 is preferable to I_1 with respect to formula ϕ if $[\![\phi]\!](I_1) < [\![\phi]\!](I_2)$.

Example 3. Let ϕ be the formula `electric` $\gg$ (`fast` $\gg$ `blue`) of Example 2. Consider the truth assignments:

$$\begin{array}{lcl} I_1 & = & \{(\texttt{electric}, [T]), (\texttt{fast}, [T]), (\texttt{blue}, [T])\} \\ I_2 & = & \{(\texttt{electric}, [T]), (\texttt{fast}, [T]), (\texttt{blue}, [F])\} \\ I_3 & = & \{(\texttt{electric}, [F]), (\texttt{fast}, [T]), (\texttt{blue}, [T])\} \end{array}$$

We get that $I_1(\phi) = [T]$, $I_2(\phi) = [0, T, 0, T, F]$ and $I_3(\phi) = [0, F, T]$. By lexicographically comparing the corresponding lists, we get that w.r.t. formula ϕ, the truth assignment I_1 is preferable to I_2 which is preferable to I_3.

A weaker associativity property holds even though in general the "$\gg$" operator is not associative.

Lemma 2. *Let ϕ_1, ϕ_2, ϕ_3 be formulas and I_1, I_2 truth assignments. Then, $[\![(\phi_1 \gg \phi_2) \gg \phi_3]\!](I_1) < [\![(\phi_1 \gg \phi_2) \gg \phi_3]\!](I_2)$ iff $[\![\phi_1 \gg (\phi_2 \gg \phi_3)]\!](I_1) < [\![\phi_1 \gg (\phi_2 \gg \phi_3)]\!](I_2)$.*

3 Properties of Lexicographic Logic

One key property that needs to be established is that the "$\gg$" operator *indeed* implements lexicographic priority. This means that when we apply "$\gg$" on two distinct pairs of arguments, and the first pair is lexicographically smaller than the second, then the first result is smaller than the second one (with respect to the ordering of Definition 4). The lexicographic ordering on pairs is defined as follows:

Definition 8. Let $u_1, u_2, v_1, v_2 \in \mathbb{V}$. We write $(u_1, u_2) <_L (v_1, v_2)$ if either $u_1 < v_1$, or $u_1 = v_1$ and $u_2 < v_2$.

Theorem 2. *Let $u_1, u_2, v_1, v_2 \in \mathbb{V}$. Then, $(u_1, u_2) <_L (v_1, v_2)$ iff $(u_1 \gg u_2) < (v_1 \gg v_2)$.*

Semantic equivalence of formulas is defined as usual.

Definition 9. The formulas ϕ_1 and ϕ_2 are semantically equivalent (denoted by $\phi_1 \equiv \phi_2$) iff for every truth assignment I, $[\![\phi_1]\!](I) = [\![\phi_2]\!](I)$.

Lexicographic logic inherits from propositional logic, the *substitutivity* of logically equivalent formulas (see, for example, (Fitting 1996, pp. 20–21)). This property holds due to the compositional semantics of lexicographic logic (Definition 6), and can be established by structural induction.

Lemma 3. *Let ϕ be a formula of lexicographic logic, ψ be a subformula of ϕ and ψ' be a formula such that $\psi \equiv \psi'$. Then, $\phi \equiv \phi[\psi \leftarrow \psi']$, where $\phi[\psi \leftarrow \psi']$ is the formula that results from ϕ by replacing the subformula ψ with ψ'.*

The following are some basic properties of lexicographic logic that can be easily established.

Lemma 4. *For all formulas $\phi_1, \phi_2, \phi_3, \phi$, the following equivalences hold:*

- $(\phi_1 \vee \phi_2) \gg \phi_3 \equiv (\phi_1 \gg \phi_3) \vee (\phi_2 \gg \phi_3)$
- $\phi_1 \gg (\phi_2 \vee \phi_3) \equiv (\phi_1 \gg \phi_2) \vee (\phi_1 \gg \phi_3)$
- $(\phi_1 \wedge \phi_2) \gg \phi_3 \equiv (\phi_1 \gg \phi_3) \wedge (\phi_2 \gg \phi_3)$
- $\phi_1 \gg (\phi_2 \wedge \phi_3) \equiv (\phi_1 \gg \phi_2) \wedge (\phi_1 \gg \phi_3)$
- $\neg(\phi_1 \gg \phi_2) \equiv (\neg\phi_1) \gg (\neg\phi_2)$
- $\neg(\neg\phi) \equiv \phi$

We now demonstrate that the elements of $\mathbb{V}$ can be mapped to rational numbers so as that their ordering is preserved. Formally, we demonstrate that there exists a function $val : \mathbb{V} \to \mathbb{Q}$ such that for all $u, v \in \mathbb{V}$, if $u < v$ then $val(u) < val(v)$. The key idea of defining val is that the elements of $\mathbb{V}$ can be considered as numbers in the interval $[-1, 1]$ written in the *balanced ternary number system* (Knuth 1998, p. 207). Balanced ternary is a ternary number system in which the coefficients are the numbers -1, 0, and 1. (instead of 0, 1, and 2, as it happens in standard ternary notation).

We consider the elements of $\mathbb{V}$ as representing balanced ternary numbers in the interval $[-1, 1]$. More specifically, $val(F) = -1$, $val(0) = 0$, $val(T) = 1$, and for every $u = [u_1, \ldots, u_n] \in \mathbb{V}$, we derive a rational number in the interval $(-1, 1)$ by viewing u as a balanced ternary number. Since we want our numbers to belong in the interval $(-1, 1)$, we calculate their value using the powers of $\frac{1}{3}$. Formally:

$$val([u_1, \ldots, u_n]) = \sum_{i=1}^{n} val(u_i) \cdot \frac{1}{3^{i-1}}$$

The following lemma justifies why the numerical representation of the elements of $\mathbb{V}$ is an equivalent alternative.

Lemma 5. *For all $u, v \in \mathbb{V}$, if $u < v$, then $val(u) < val(v)$.*

The above discussion suggests that the semantics of lexicographic logic can be equivalently expressed using a special subset of the set $\mathbb{Q}$ of rational numbers. In a potential implementation of a query system based on lexicographic logic, numerical values that rank query results would convey a much better intuition than the lists of truth values.

4 Modeling Alternative Operators

In this section, we demonstrate that we can use "$\gg$" as a primitive operator in order to define other interesting connectives that express levels of preference. We focus on two well-known such operators (Dubois and Prade 2013), namely "*and if possible*" and "*or at least*". We will denote the former by "&" and the latter by "$\times$".

The intuitive meaning of $(x \,\&\, y)$ is "*I want x and if possible additionally y*". If the first argument of "&" is false, the result is false (no matter what the second argument is). If both arguments are true, the result is true. If, however, the first argument is true and the second is false, then we are partially satisfied. Similarly, the intuitive meaning of $(x \times y)$ is "*I want x, or failing that, y*". We expect that if both arguments are false, then the result is false. If the first argument is true, then the result is true. If, however, the first argument is false and the second is true, then we are still satisfied, but not as much as in the case where the first argument is true; this is expressed by the value $[T] \gg [F]$ which is not absolutely true (but is still true). We can express "&" and "$\times$" as derived operators using "$\gg$", as:

$$x \,\&\, y = x \gg (x \wedge y)$$
$$x \times y = (x \vee y) \gg x$$

The above definitions are meaningful even if "&" and "$\times$" are applied on non-classical elements of $\mathbb{V}$, ie., elements that are different from $[F]$ or $[T]$. In this case, the intuition of "&", for example, is to put a strong emphasis on the value of its first argument. On the other hand, the intuition of "$\times$" is to degrade the overall result if x has some false value.

Our definitions satisfies the equivalences between the connectives '$\times$', '$\gg$', '$\wedge$' and '$\vee$' observed in (Dubois and Prade 2013) as the following lemma demonstrates.

Lemma 6. *For all $x, y \in \mathbb{V}$, it holds:*

- $x \times y = x \times (x \vee y) = (x \vee y) \,\&\, x$
- $x \,\&\, y = x \,\&\, (x \wedge y) = (x \wedge y) \times x$.

5 Related Work

There exists a great variety of preference representation formalisms that have been developed mainly in the areas of artificial intelligence (Domshlak et al. 2011), databases (Stefanidis, Koutrika, and Pitoura 2011), and logic programming (Sakama and Inoue 2000). In general, preference representation formalisms are classifiedas either "*quantitative*" or "*qualitative*". In the quantitative approach, numerical values are used in the syntax of the preference specification language in order to express degrees of preference. On the other hand, in the qualitative approach, preferences are expressed by implicitly establishing a preference relation between the objects under consideration. Lexicographic logic falls somewhere in between: its syntax is qualitative because preferences are expressed implicitly through the "$\gg$" operator; however, its semantics is quantitative because, as shown in Section 3, formulas essentially receive rational number values when evaluated. In the following we discuss the research works that we are aware of and are closer to our contribution.

Qualitative Choice Logic (QCL) In (Brewka, Benferhat, and Berre 2004), propositional logic is extended with the new connective "$\times$". Intuitively, $A \times B$ is read "*if possible A, but if A is impossible then at least B*". QCL has a similar philosophy as lexicographic logic in the sense that preferences are represented implicitly using a new operator. However, there are important differences. QCL is built on the classical boolean truth domain while lexicographic logic uses a many-valued one. It has been remarked that the semantics of QCL lead to certain limitations (Benferhat and Sedki 2007). Moreover, some intuitive tautologies, such as $\neg(\neg\phi) \equiv \phi$ do not hold, and the notion of logical equivalence between QCL formulas is defined in a non-standard way (Brewka, Benferhat, and Berre 2004, page 209). Additionally, the semantics of QCL can be used to prove that the operator "$\times$" of QCL is associative, something that does not hold for the operator "$\times$" we defined in Section 4. It is also worth noting that QCL is a non-monotonic logic, while lexicographic logic has not been extended with a non-monotonic consequence relation.

Flexible queries The operators "A and if possible B" and "A or at least B" have been studied as different forms of bipolar constraints for flexible querying in (Dubois and Prade 2013). The semantics of these operators has been modeled using possibilistic logic (Dubois and Prade 2014). It is however unclear whether a lexicographic priority operator like "$\gg$" can be encoded in that framework and whether arbitrary nestings of such operators can be supported. Overall, we believe that lexicographic logic provides a simpler and more natural means for encoding such operators.

Infinite-valued Logic In (Agarwal and Wadge 2005; Agarwal 2005) a propositional query language is developed that uses two operators to express preferences of the form "*A and optionally B*" and "*A or alternatively B*". The semantics of this language is based on the infinite-valued logic introduced in (Rondogiannis and Wadge 2005) and was later extended with recursion (Rondogiannis and Troumpoukis 2015). The infinite-valued approach has a similar philosophy with lexicographic logic: preferences are expressed implicitly using operators in the context of many-valued logics. However, there is a significant difference: as demonstrated in (Papadimitriou 2017), the infinite valued logic of (Rondogiannis and Wadge 2005) is not sufficient to express lexicographic preferences. Thus, our present approach is more powerful as it can express all the preferences of the latter, and additionally it can also express lexicographic ones.

6 Conclusions and Future Work

We introduced lexicographic logic,a many-valued logic for preference representation. Lexicographic logic has a simple semantics based on sequences of the truth values F, 0, and T, which can be alternatively defined using a subset of the rational numbers in the interval $[-1, 1]$.

The next step in the development of lexicographic logic, is the introduction of a deductive calculus. Such an investigation could include an inference procedure, investigation of completeness issues (with respect to the model-theoretic semantics), and a corresponding complexity-theoretic study.

Acknowledgments

The authors would like to thank one of the reviewers for suggesting Lemma 2.

This research is co-financed by Greece and the European Union European Social Fund- ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning 2014- 2020" in the context of the project "Techniques for implementing qualitative preferences in deductive querying systems" (5048151).

References

Agarwal, R., and Wadge, W. W. 2005. The lazy evaluation of infinitesimal logic expressions. In Arabnia, H. R., ed., *Proceedings of The 2005 International Conference on Programming Languages and Compilers, PLC 2005, Las Vegas, Nevada, USA, June 27-30, 2005*, 3–7. CSREA Press.

Agarwal, R. 2005. A framework for expressing prioritized constraints using infinitesimal logic. Master's thesis, University of Victoria, Canada.

Benferhat, S., and Sedki, K. 2007. A revised qualitative choice logic for handling prioritized preferences. In Mellouli, K., ed., *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 9th European Conference, ECSQARU 2007, Hammamet, Tunisia, October 31 - November 2, 2007, Proceedings*, volume 4724 of *Lecture Notes in Computer Science*, 635–647. Springer.

Brewka, G.; Benferhat, S.; and Berre, D. L. 2004. Qualitative choice logic. *Artificial Intelligence* 157(1-2):203–237.

Brewka, G.; Niemelä, I.; and Truszczynski, M. 2008. Preferences and nonmonotonic reasoning. *AI Magazine* 29(4):69–78.

Charalambidis, A.; Rondogiannis, P.; and Troumpoukis, A. 2018. Higher-order logic programming: An expressive language for representing qualitative preferences. *Science of Computer Programming* 155:173–197.

Domshlak, C.; Hüllermeier, E.; Kaci, S.; and Prade, H. 2011. Preferences in AI: an overview. *Artificial Intelligence* 175(7-8):1037–1052.

Dubois, D., and Prade, H. 2013. Modeling "and if possible" and "or at least": Different forms of bipolarity in flexible querying. In Pivert, O., and Zadrozny, S., eds., *Flexible Approaches in Data, Information and Knowledge Management*, volume 497 of *Studies in Computational Intelligence*. Springer. 3–19.

Dubois, D., and Prade, H. 2014. Possibilistic logic - an overview. In Siekmann, J. H., ed., *Computational Logic*, volume 9 of *Handbook of the History of Logic*. Elsevier. 283–342.

Fishburn, P. C. 1999. Preference structures and their numerical representations. *Theoretical Computer Science* 217(2):359–383.

Fitting, M. 1996. *First-Order Logic and Automated Theorem Proving, Second Edition*. Graduate Texts in Computer Science. Springer.

Knuth, D. E. 1998. *The art of computer programming*, volume 2. Boston, MA, USA: Addison-Wesley Longman Publishing Co., 3rd edition.

Lang, J. 2009. Logical representation of preferences. In Bouyssou, D.; Dubois, D.; Pirlot, M.; and Prade, H., eds., *Decision-making Process*. Wiley. 321–363.

Papadimitriou, G. 2017. A logic query language for lexicographic preferences. Master's thesis, National and Kapodistrian University of Athens, Greece.

Rondogiannis, P., and Troumpoukis, A. 2015. Expressing preferences in logic programming using an infinite-valued logic. In Falaschi, M., and Albert, E., eds., *Proceedings of the 17th International Symposium on Principles and Practice of Declarative Programming, Siena, Italy, July 14-16, 2015*, 208–219. ACM.

Rondogiannis, P., and Wadge, W. W. 2005. Minimum model semantics for logic programs with negation-as-failure. *ACM Trans. Comput. Log.* 6(2):441–467.

Sakama, C., and Inoue, K. 2000. Prioritized logic programming and its application to commonsense reasoning. *Artificial Intelligence* 123(1-2):185–222.

Stefanidis, K.; Koutrika, G.; and Pitoura, E. 2011. A survey on representation, composition and application of preferences in database systems. *ACM Transactions on Database Systems* 36(3):19:1–19:45.