Learning various classes of models of lexicographic orderings

Richard Booth[1], Yann Chevaleyre[2], Jérôme Lang[2], Jérôme Mengin[3⋆], and Chattrakul Sombattheera[1]

[1] Mahasarakham University, Thailand
[2] LAMSADE, Université Paris-Dauphine, France
[3] IRIT, Université de Toulouse, France

Abstract. We consider the problem of learning a user's ordinal preferences on multiattribute domains, assuming that the user's preferences may be modelled as a kind of *lexicographic* ordering. We introduce a general graphical representation called *LP-structures* which captures various natural classes of such ordering in which both the order of *importance* between attributes and the *local preferences* over each attribute may or may not be conditional on the values of other attributes. For each class we determine the Vapnik-Chernovenkis dimension, the communication complexity of learning preferences, and the complexity of identifying a model in the class consistent with some given user-provided examples.

1 Introduction

In many applications, especially electronic commerce, it is important to be able to learn the preferences of a user on a set of alternatives that has a combinatorial (or multiattribute) structure: each alternative is a tuple of values for each of a given number of variables (or attributes). Whereas learning *numerical* preferences (*i.e.*, utility functions) on multiattribute domains has been considered in various places, learning *ordinal* preferences (*i.e.*, order relations) on multiattribute domains has been given less attention. Two streams of work are worth mentioning.

First, a series of very recent works focus on the learning of preference relations enjoying some preferential independencies conditions. Passive learning of separable preferences is considered by Lang & Mengin (2009), whereas passive (resp. active) learning of acyclic CP-nets is considered by Dimopoulos *et al.* (2009) (resp. Koriche & Zanuttini, 2009).

The second stream of work, on which we focus in this paper, is the class of *lexicographic* preferences, considered in Schmitt & Martignon (2006); Dombi *et al.* (2007); Yaman *et al.* (2008). These works only consider very simple classes of lexicographic preferences, in which both the importance order of attributes and the local preference relations on the attributes are unconditional. These very simple lexicographic preference models exclude the possibility to represent some more complex, yet natural, relations between objects. Suppose for instance that you want to buy a computer at a simple e-shop. Assuming your cash is not unlimited, the website first asks you to enter the

⋆ Corresponding author

maximum price you can afford to pay (for simplicity, we suppose here that this is not conditioned by the computer that you may buy). The objective of the website is to find the best (according to your preferences) computer you can afford. Suppose first that you always prefer laptops to desktop computers: the distinction between laptop and desktop makes the most important attribute to order computers according to your taste. Now, there are two other important criteria: the color of the computer, and whether it has a simple DVD-reader or a powerful DVD-writer. The color may be more important than the type of optical drive in the case of a laptop, because you would not want to be seen at a meeting with the usual bland, black laptop; in fact, you always prefer a flashy yellow laptop to a black one – whereas it is the opposite with desktops, because working long hours in front of a yellow desktop may be a strain for your eyes. Interestingly, this examples indicates that both the importance of the attributes and the local preference on the values of some attributes may be conditioned by the values of some other attributes: here, the relative importance of the color and the type of optical drive depends on the type of computer; and the preferred color depends on the type of computer as well.

In this paper we go further and consider various classes of lexicographic preference models, where the importance relation between attributes and/or the local preference on an attribute may depend on the values of some more important attributes. In Section 2 we give a general model for lexicographic preference relations, and define six classes of lexicographic preference relations, only two of which have already been considered from a learning perspective. Then each of the following sections focuses on a specific kind of learning problem: in Section 3 we address the sample complexity of learning lexicographic preferences, in Section 4 we consider preference elicitation, *a.k.a.* active learning, and in Section 5 we consider passive learning, and more specifically model identification and approximation. All proofs can be found in (Booth *et al.* 2009).

2 Lexicographic preference relations: a general model

2.1 Lexicographic preferences structures

We consider a set $\mathcal{A}$ of n attributes, also called variables. Each attribute $X \in \mathcal{A}$ has an associated finite domain $\underline{X}$. We assume the domains of the various attributes are disjoint. An attribute X is binary if its domain contains exactly two values, which by convention are denoted by x and $\overline{x}$. If $U \subseteq \mathcal{A}$ is a subset of the attributes, then $\underline{U}$ is the cartesian product of the domains of the attributes in U. Attributes, as well as sets of attribute, are denoted by upper-case Roman letters (X, X_i, A etc.) and attribute values by lower-case Roman letters. An outcome is an element of $\underline{\mathcal{A}}$; we will denote outcomes using greek lower case Greek letters (α, β, etc.).

Given a (partial) assignment $u \in \underline{U}$ for some $U \subseteq \mathcal{A}$, and $V \subseteq \mathcal{A}$, we denote by $u(V)$ the assignment made by u to the attributes in $U \cap V$.

Lexicographic comparison is a general way of ordering any pair of outcomes $\{\alpha, \beta\}$ by looking at the attributes in sequence, until one attribute X is reached such that α and β have different values of X: $\alpha(X) \neq \beta(X)$; the two outcomes are then ordered according to the *local preference* relation over the values of this attribute. Such a comparison uses two types of relation: a relation of *importance* between attributes, and *local preference* relations over the domain of each attribute.

Both the importance between attributes and the local preferences may be conditional. In the introductory example, if two computers share the value l (for laptop) for the attribute T (type), then C (color) is more important than D (the type of optical Drive), and y (yellow) is preferred to b (black); whereas when comparing computers of type d (desktops), D is more important than C, and b is preferred to y. Note that the condition on the type of computer assumes here that the two objects have the same value for this attribute. In this paper, we will only consider this simple type of conditions, which implies that the attributes that appear in the condition (T on the example) must be more important than the attribute over which a local preference is expressed (C) or the attributes, the importance of which is compared (C and D). [4]

Importance between attributes is captured by *Attribute Importance Trees:*

Definition 1. *An* Attribute Importance Tree *(or AI-tree for short) over set of attributes $\mathcal{A}$ is a tree whose nodes are labelled with attributes, such that no attribute appears twice on the same branch, and such that the edges between a non-leaf node n, labelled with attribute X, and its children are labelled with disjoint sets of values of X. An AI-tree is* complete *if every attribute appears in every one of its branches.*

For the sake of clarity, if one edge is labelled with the entire domain of an attribute X, we can omit this label (the labels on the edges are there to choose how to descend the tree according to the values of the attribute, and an edge labelled with the full domain of an attribute means there is no choice – note that there is no other "sibling" edge in this case since labels of different edges must be disjoint). Also, if an edge is labelled with a singleton $\{x\}$, we will often refer to the label by the value x itself.

We will denote by $\mathsf{Anc}(n)$ the set of ancestors of node n, that is the nodes on the path from the root to the parent of n. We will often identify $\mathsf{Anc}(n)$ with the set of attributes that label the ancestor nodes of n. We will denote by $\underline{n}$ the cartesian product of the labels of the edges on the path from the root to n.

Let us now turn to the representation of the local preferences on each attribute. When we want to compare two outcomes α and β using a lexicographic ordering, we go down the tree until we reach a node labelled with an attribute X that has different values in α and β: at this stage, we must be able to choose between the two outcomes according to some ordering over X.

Definition 2. *A* Local Preference Rule *(for attribute X, over set of attributes $\mathcal{A}$) is an expression $X, u :>$ where $u \in \underline{U}$ for some $U \subseteq \mathcal{A}$, $X \in \mathcal{A} - U$, and $>$ is a total linear order over $\underline{X}$. A* Local Preference Table *(over set of attributes $\mathcal{A}$) is a set of local preference rules.*

[4] Exploring the possibility to have the local preference on an attribute domain depend on the value of a less important attribute is an interesting research direction, but it leads to many problems, starting from the fact that it may fail to be fully defined: take $\alpha = x_1x_2$, $\beta = \overline{x_1}\overline{x_2}$, X_2 being more important than X_1, and assume we have this local preference relation for X_2: x_2 is preferred to $\overline{x_2}$ if $X_1 = x_1$ and $\overline{x_2}$ is preferred to x_2 if $X_1 = \overline{x_1}$. The most important attribute on which α and β differ is X_2, however the values of X_1 in α and β differ, therefore the local preference rules do not allow to order α and β. In other cases, preference cycles may appear.

So far, importance trees and local preference tables have been defined independently. Now, as we said above, we require that the local preference relation for an attribute depends only on the values of more important attributes. For this we need the following definition:

Definition 3. *Let T be an AI-tree, n a node of T, and P a local preference table. A rule $X, v :> $ of P is said to be* applicable at node n given assignment $u \in \underline{n}$ *if (a) n is labelled by X and (b) $v \subseteq u$. P is* unambiguous w.r.t. T *(resp.* complete w.r.t. T*) if for any node n of T and any $u \in \underline{n}$, there is at most one (resp. exactly one) rule applicable at n given u.*

Definition 4. *A* Lexicographic Preference Structure *(or LP-structure) is a pair (T, P) where T is an attribute importance tree and P an unambiguous local preference table w.r.t. T. If furthermore T is complete and P is complete w.r.t. T, then (T, P) is a* Complete Lexicographic Preference Structure.

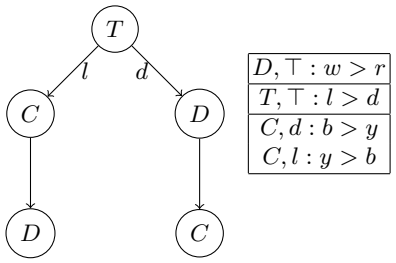

Fig. 1. Graphical representation of lexicographic orderings for Example 1

Example 1. Consider three attributes C(olor) with two values y(ellow) and b(lack), D(vd device) with two values w(riter) and r(ead-only), and T(ype) with values l(aptop) and d(deskop). The LP-structure σ depicted on Fig. 1 is a model for preferences about computers, where the type of computer is the most important criteria, with laptops always preferred to desktops, and where the second criterium is color in the case of laptops, with yellow laptops preferred to black ones, whereas the second criterium is the type of optical drive in the case of desktops. In any case, a writer is always preferred to a read-only drive. The color is third criteria for desktops, with black preferred to yellow in this case.

The semantics of LP-structures is defined by the associated orderings over outcomes:

Definition 5. *LP-structure $\sigma = (T, P)$ defines a partial strict order $>_\sigma$ over the set of outcomes as follows: given any pair of outcomes $\{\alpha, \beta\}$, go down the tree, starting at the root, following edges that correspond to assignments made in α and β, until the*

first node n is reached that is labelled with attribute X such that $\alpha(X) \neq \beta(X)$; we say that n decides *$\{\alpha, \beta\}$. If there is a rule $X, v :> $ in P that is applicable at n given $u = \alpha(\mathsf{Anc}(n)) = \beta(\mathsf{Anc}(n))$, then $\alpha >_\sigma \beta$ if and only if $\alpha(X) > \beta(X)$. If there is no rule that is applicable at n given u, or if no node that decides $\{\alpha, \beta\}$ is reached, α and β are σ-incomparable.*

Example 1 (continued). According to σ, the most preferred computers are yellow laptops with a DVD-writer, because $ywl >_\sigma \alpha$ for any other outcome $\alpha \neq ywl$; for any $x \in \underline{C}$ and any $z \in \underline{D}$ $xzl >_\sigma xzd$, that is, any laptop is preferred to any desktop computer. And $ywd >_\sigma brd$, that is, a yellow deskop with DVD-writer is preferred to a black one with DVD-reader because, although for desktops black is preferred to yellow, the type of optical reader is more important than the colour for desktop computers.

Proposition 1. *Given a LP-structure $\sigma = (T, P)$, the relation $>_\sigma$ is irreflexive and transitive. It is also modular, i.e., $\alpha >_\sigma \beta$ implies either $\alpha >_\sigma \gamma$ or $\gamma >_\sigma \beta$. Moreover, if σ is complete, then $>_\sigma$ is a linear order.*

The above proposition is saying $>_\sigma$ is a modular strict partial order. Every such order can be seen as the strict version of a total preorder. This means that even when $>_\sigma$ is not a linear order, it may still be viewed as "ranking" the different outcomes, with outcomes which are σ-incomparable given the same rank. (To be more precise, the relation $\geq_\sigma$ defined by $\alpha \geq_\sigma \beta$ iff [$\alpha >_\sigma \beta$ or α, β are σ-incomparable] is a total preorder.)

2.2 Classes of lexicographic preference structures

Classes of LP-structures with conditional preferences It should be clear that any LP-structure σ is equivalent to a LP-structure σ' where each edge corresponds to exactly one value of its parent node, and where each preference rule applies to exactly one node: σ' can be obtained from σ by multiplying the edges that correspond to more than one value; and by multiplying the preference rules that apply at more than one node. This structure σ' can be seen as a canonical representation of $>_\sigma$. This leads to the following defnition:

Definition 6. *A* CP&I *LP-structure, or structure with conditional local preferences and conditional attribute importance, is a structure in which each edge of the tree is labelled with a singleton value, and such that for each node n, that corresponds to exactly one partial assignment u, the preference table contains one rule of the form $X, u :> $ where X is the attribute that labels n.*

CP&I LP-structures are particular cases of Wilson's "Pre-Order Search Trees" (or POST) (2006): in POSTs, the preference relation at every node can be a non strict relation.

Example 1 (continued). A CP&I structure equivalent to the LP-structure depicted on Fig. 1 for Example 1 is depicted on Fig. 2. Note that when we draw a *CP&I* LP-structure, since local preferences at a given node can only depend on attributes above that node, we can represent the local preference relation corresponding to a node inside the node itself.

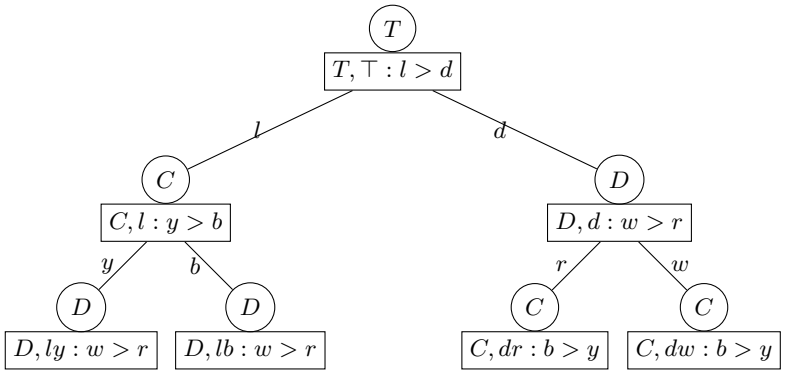

Fig. 2. A CP&I structure equivalent to that of Example 1

Another interesting class is that of structures with conditional preferences but *un*conditional attribute importance:

Definition 7. *A* CP-UI *LP-structure, or structure with conditional local preferences and* un*conditional attribute importance, is a structure in which the tree is linear, with each edge labelled with the full domain of the attribute at the parent node, and such that for each node n, for each partial assignment $u \in \underline{n}$, the preference table contains one rule of the form $X, u :>$ where X is the attribute that labels n.*

Classes of LP-structures with unconditional preferences We now turn to lexicographic preferences with unconditional preferences, like the ones studied by e.g. Schmitt & Martignon (2006); Dombi *et al.* (2007); Yaman *et al.* (2008):

Definition 8. *UP&I LP-structures, or structures with* un*conditional local preferences and* un*conditional attribute importance, are structures whose attribute importance tree is linear, each edge being labelled with the full domain of the attribute at the parent node, and whose preference table contains one unconditional rule of the form $X, \top :>$ for each attribute X that appears in the tree. UP-CI LP-structures, or structures with* un*conditional local preferences and conditional attribute importance, are structures in which each edge of the tree is labelled with a singleton value, so that each node corresponds to exactly one partial assignment, but such that for each attribute that appears in the tree, the local preference table contains only one unconditional rule of the form $X, \top :>$.*

We can also define classes of LP-structures with unconditional, **fixed** preferences:

Definition 9. *Given a non ambiguous set P of preferences rules, FP-UI(P) is the class of UP&I structures that have P for preference table. Similarly, FP-CI(P) is the class of UP-CI structures that have P for preference table.*

3 Sample complexity of some classes of LP-structures

Our aim in this paper is to study how we can learn a LP-structure that fits well some examples of comparison. We assume a set $\mathcal{E}$ of examples, that is, of pairs of outcomes over $\mathcal{A}$: we would like to find a LP-structure that is "consistent" with the examples in the following sense:

Definition 10. *LP-structure σ is said to be* consistent *with example $(\alpha, \beta) \in \underline{\mathcal{A}}^2$ if $\alpha >_\sigma \beta$; σ is consistent with set of examples $\mathcal{E}$ if it consistent with every example of $\mathcal{E}$.*

The problem of learning a structure that orders "well" the examples can be seen as a problem of classification: given σ we can define another binary relation $\leq_\sigma$ over $\underline{\mathcal{A}}^2$ as follows:

$$\alpha \leq_\sigma \beta \text{ if and only if } \beta >_\sigma \alpha \text{ or } (\alpha \not>_\sigma \beta \text{ and } \beta \not>_\sigma \alpha).$$

Because $>_\sigma$ is modular, $\leq_\sigma$ defined in this way is a total preorder over $\underline{\mathcal{A}}$ (i.e. the relation is reflexive, transitive, and for every $\alpha, \beta \in \underline{\mathcal{A}}$, at least one of $\alpha \leq_\sigma \beta$ or $\beta \leq_\sigma \alpha$ holds), and $\{\leq_\sigma, >_\sigma\}$ is a partition of $\underline{\mathcal{A}}^2$. In particular, we can define the Vapnik-Chernovenkis dimension of a class of LP-structures as the size of the biggest set of pairs (α, β) that can be "classified" correctly by some LP-structure in the class, whatever the labels ($>$ or $\leq$) associated with each pair. In general, the higher this dimension, the more examples will be needed to correctly identify a LP-structure.

Proposition 2. *The VC dimension of any class of transitive relations over a set of binary attributes is strictly less than 2^n.*

Proof (Sketch). Follows from the fact that any graph with 2^n vertices and 2^n edges contains at least one cycle, so that no class of transitive binary relations can shatter a set of 2^n examples over $\underline{\mathcal{A}}$.

Proposition 3. *The VC dimension of both classes of CP&I LP-structures and of CP-UI structures over n* binary *attributes, is equal to $2^n - 1$.*

Proof (Sketch). It is possible to build an AI tree over n attributes with 2^k nodes at the $k - th$ level, for $0 \leq i \leq n - 1$, with $|\underline{n}| = 1$ for every node: this is a tree for CP&I structures, it has $2^n - 1$ nodes. Such a tree can shatter a set of $2^n - 1$: take one example for each node, the local preference relation that is applicable at each node can be used to give both labels to the corresponding example. The upper bound follows from Prop. 2.

This result is rather negative, since it indicates that a huge number of examples would in general be necessary to have a good chance of closely approximating an unknown target relation. This important number of necessary examples also means that it would not be possible to learn in reasonable - that is, polynomial - time. However, learning CP&I LP-structures is not hopeless in practice: decision trees have a VC dimension of the same order of magnitude, yet learning them has had great success experimentally.

As for structures with unconditional preferences, Schmitt & Martignon (2006) have shown that hhe VC dimension of UP&I structures over n *binary* attributes is exactly n. Since every UP&I structure is equivalent to a CP-UI one, the VC dimension of UP&I structures over n *binary* attributes is at least n.

4 Preference elicitation/active learning

We now turn to a the *active learning* of preferences. The setting is as follows: there is some unknown target preference relation $>$, and a *learner* wants to learn a representation of it by means of a Lexicographic Preference structure. There is a *teacher*, a kind of oracle to which the learner can submit queries of the form $\{\alpha, \beta\}$ where α and β are two outcomes: the teacher will then reply wether $\alpha > \beta$ or $\beta > \alpha$ is the case. An important question in this setting is: how many queries does the learner need in order to completely identify the target relation $>$? More precisely, we want to find the communication complexity of preference elicitation, i.e., the worst-case number of requests to the teacher to ask so as to be able to elicit the preference relation completely, assuming the target can be represented by a model in a given class. The question has already been answered in Dombi *et al.* (2007) for the FP-UI case. Here we identify the communication complexity of eliciting lexicographic preferences structures in all 5 other cases, when all attributes are binary. (We restrict to the case of binary attributes for the sake of simplicity. The results for nonbinary attributes would be similar.) We know that a lower bound of the communication complexity is the $\log$ of the number of preference relations in the class. In fact, this lower bound is reached in all 6 cases:

Proposition 4. *The communication complexities of the six problems above are as follows,* when all attributes are binary.

	FP	*UP*	*CP*
UI	$\log(n!)$ *Dombi* et al. *(2007)*	$n + \log(n!)$	$2^n - 1 + \log(n!)$
CI	$g(n) = \sum_{k=0}^{n-1} 2^k \log(n-k)$	$n + g(n)$	$2^n - 1 + g(n)$

Proof (Sketch). In the four cases FP-UI, UP&I, FP-CI and UP-CI, T and P are independent, *i.e.*, any P is compatible with any T. There are $n!$ unconditional importance trees, and $\prod_{k=0}^{n-1}(n-k+1)^{2^k}$ conditional ones. Moreover, when preferences are not fixed, there are 2^n possible unconditional preference tables. For the CP-CI case, a complete conditional importance tree contains $\sum_{k=0}^{n-1} 2^k = 2^n - 1$ nodes, and at each node there are two possible conditional preference rules. The fact that these lower bounds are reached (in all 5 cases for which this has not been proved by Dombi *et al.* , 2007), we can explicit an elicitation protocol that guarantees to identifiy the preference structure.

5 Model identifiability

We now turn to the problem of identifying a model of a given class $\mathcal{C}$, given a set $\mathcal{E}$ of examples: each example is a pair (α, β), for which we know that $\alpha > \beta$ for some target preference relation $>$. The aim of the learner is to find some LP-structure σ in $\mathcal{C}$ such that $\alpha >_\sigma \beta$ for every $(\alpha, \beta) \in \mathcal{E}$.

Dombi *et al.* (2007) have shown that the corresponding decision problem for the class of binary LP-structures with unconditional importance and unconditional, fixed local preferences can be solved in polynomial time: given a set of examples $\mathcal{E}$ and a set P of unconditional local preferences for all attributes, is there a structure in $\mathrm{FP} - \mathrm{UI}(P)$

Algorithm 1 GenerateLPStructure

INPUT: $\mathcal{A}$: set of attributes; $\mathcal{E}$: set of examples over $\mathcal{A}$;
P: set of local preference rules: initially empty,
or contains a set of unconditional preference rules for the **FLP** cases;
OUTPUT: LP-structure consistent with $\mathcal{E}$, that contains P, or FAILURE;

1. $T \leftarrow$ {unlabelled root node};
2. while T contains some unlabelled node:
   (a) choose unlabelled node n of T;
   (b) $(X, newRules) \leftarrow \mathsf{chooseAttribute}(\mathcal{E}(n), \mathsf{Anc}(n), P)$;
   (c) if $X = \mathsf{FAILURE}$ then STOP and return FAILURE;
   (d) label n with X;
   (e) $P \leftarrow P \cup newRules$;
   (f) $L \leftarrow \mathsf{generateLabels}(\mathcal{E}(n), X)$; *(Create set of labels for edges below n)*
   (g) for each $l \in L$:
   add new unlabelled node to T, attached to n with edge labelled with l;
3. return (T, P).

that is consistent with $\mathcal{E}$? In order to prove this, they exhibit a simple greedy algorithm. We will prove in this section that the result still holds for most of our classes of LP-structures, except one.

5.1 A greedy algorithm

In order to prove this, we will prove that the greedy Algorithm 1, when given a set of examples $\mathcal{E}$, returns a LP-structure that satisfies the examples if one exists. The algorithm recursively constructs the AI-tree from the root to the leaves. At a given currently unlabelled node n, step 2b considers the set $\mathcal{E}(n) = \{(\alpha, \beta) \in \mathcal{E} \mid \alpha(\mathsf{Anc}(n)) = \beta(\mathsf{Anc}(n)) \in \underline{n}\}$ of examples that correspond to the assignments made in the branch so far and that are still undecided: it looks for some attribute $X \notin \mathsf{Anc}(n)$ that can be used to order well examples in $\mathcal{E}(n)$ that can be ordered with X: there must be a set of local preferences rules of the form $X, w :>$ that is not ambiguous when put together with the current set of rules, and such that for every $(\alpha, \beta) \in \mathcal{E}(n)$, if $\alpha(X) \neq \beta(X)$ then there is a rule $X, w :>$ with $w \subseteq \alpha(U) = \beta(U)$ and $\alpha(X) > \beta(X)$. The attribute X can then be chosen for the label of n, and the set of rules added to P. Step 2f then considers the values of X that correspond to still undecided examples, and prepare labels that will be used for the edges from n to its children. P is initially empty except in the case where the local preferences are known in advance, with only the order of importance to be learned. Note that this aproach cannot work in the case of conditional importance and unconditional preferences, as will be proved in Corollary 1.

Let us briefly describe the helper functions that appear in the algorithm:

generateLabels should return a set of disjoint subsets of the domain of the attribute at the current node; it takes as parameters a set of examples $\mathcal{E}(n)$, and the attribute X at the current node: we require that for each example $(\alpha, \beta) \in \mathcal{E}(n)$ that cannot be

decided at n because $\alpha(X) = \beta(X)$, there is one label returned by generateLabels that contains $\alpha(X)$.

We will use two particular instances of the function generateLabels:

generateCondLabels$(\mathcal{E}, X)$ =
$\{\{x\} \mid x \in \underline{X}$ and there is $(\alpha, \beta) \in \mathcal{E}$ such that $\alpha(X) = \beta(X) = x\}$:
in the case of conditional importance, each branch corresponds to one value of X.

generateUncondLabel$(\mathcal{E}, X) = \{\underline{X}\}$: in the case of unconditional importance, one branch is created, except that if there is no $(\alpha, \beta) \in \mathcal{E}$ such that $\alpha(X) = \beta(X) = x$, then generateUncondLabel$(\mathcal{E}, X) = \emptyset$.

chooseAttribute takes as parameters the set of examples $\mathcal{E}(n)$ that correspond to the node being treated, the set of attributes $\mathsf{Anc}(n)$ that already appear on the current branch, and the current set of preference rules P; it returns an attribute X not already on the branch to n and a set $newRules$ of local preference rules over X: the attribute and the rules should be chosen so that they will decide well some examples of $\mathcal{E}(n)$. More precisely, we will require that $(X, newRules)$ is *choosable* with respect to $\mathcal{E}(n), \mathsf{Anc}(n), P$ in the following sense:

Definition 11. *Given a set of examples $\mathcal{E}$ over attributes $\mathcal{A}$, a set of attributes $U \subseteq \mathcal{A}$ and a set of local preference rules P, $(X, newRules)$ is* choosable with respect to *$\mathcal{E}, U, P$ if $X \in \mathcal{A} - U$, $newRules$ is a set of local preference rules for X, and:*

- *$P \cup newRules$ is not ambiguous;*
- *for every $(\alpha, \beta) \in \mathcal{E}$, if $\alpha(X) \neq \beta(X)$ then there is a (unique) rule $X, v :>$ in $P \cup newRules$ such that $v \subseteq \alpha(U)$, and $\alpha(X) > \beta(X)$.*

Moreover, we will say that $(X, newRules)$ is:

UP-choosable *if it is choosable and $newRules$ if of the form $\{X, \top :>\}$ (it contains a single unconditional rule);*

CP-choosable *if it is choosable and $newRules$ contains one rule $X, u :>$ for every $u \in \underline{U}$ such that there exists $(\alpha, \beta) \in \mathcal{E}$ with $\alpha(U) = \beta(U) = u$.*

5.2 Some examples of GenerateLPStructure

In these examples we assume three binary attributes A, B, C. Throughout this subsection we assume the algorithm checks the attributes for choosability in the order $A \to B \to C$. Furthermore we assume we are not in the FP case, i.e., the algorithm initialises with an empty local preference table $P = \emptyset$.

Example 2. Suppose $\mathcal{E}$ consists of the following five examples:

$$1.\ (abc, a\bar{b}c)\ \ 2.\ (\bar{a}\bar{b}c, \bar{a}bc)\ \ 3.\ (ab\bar{c}, a\bar{b}\bar{c})\ \ 4.\ (\bar{a}\bar{b}\bar{c}, \bar{a}b\bar{c})\ \ 5.\ (\bar{a}\bar{b}c, \bar{a}b\bar{c})$$

Let's try using the algorithm to construct a UP&I structure consistent with $\mathcal{E}$. At the root node n_0 of the AI-tree we first check if $(A, newRules)$ is UP-choosable w.r.t. $\mathcal{E}, \emptyset, \emptyset$. By the definition of UP-choosability, $newRules$ must be of the form $\{A, \top :>\}$ for some total order $>$ of $\{a, \bar{a}\}$. Now since $\alpha(A) = \beta(A)$ for all $(\alpha, \beta) \in \mathcal{E}(n_0) = \mathcal{E}$,

$(A, \{A, \top :>\})$ is choosable for *any* $>$. Thus we label n_0 with A and add $\{A, \top : a?\bar{a}\}$ to P, where "?" is some arbitrary order ($<$ or $>$) over $\{a, \bar{a}\}$. Since we are working in the UP-case the algorithm then calls $\mathsf{generateUncondLabel}(\mathcal{E}, A) = \{a, \bar{a}\}$ and generates a single edge from n_0 labelled with $\{a, \bar{a}\}$ and leading to a new unlabelled node n_1. The examples $\mathcal{E}(n_1)$ corresponding to the next node will be just $\{(\alpha, \beta) \in \mathcal{E} \mid \alpha(A) = \beta(A)\} = \mathcal{E}$ (i.e., no examples in $\mathcal{E}$ are removed). [5] At the next node n_1, with A now taken care of, we check if $(B, newRules)$ is UP-choosable w.r.t. $\mathcal{E}(n_1), \{A\}, P$. We see that it is not UP-choosable, owing to the opposing preferences over B exhibited for instance in examples 1,2 of $\mathcal{E}$. However $(C, \{C, \top : c > \bar{c}\})$ *is* UP-choosable, thus the algorithm labels n_1 with C and adds $C, \top : c > \bar{c}$ to P. At the next node n_2 we have $\mathcal{E}(n_2) = \{(\alpha, \beta) \mid \alpha(\{A, C\}) = \beta(\{A, C\})\} = \{1, 2, 3, 4\}$. But the only remaining attribute B is not UP-choosable w.r.t. $\mathcal{E}(n_2), \{A, C\}, P$ (because for instance we still have $1,2 \in \mathcal{E}(n_2)$). Thus the sub-algorithm $\mathsf{chooseAttribute}(\mathcal{E}(n_2), \{A, C\}, P)$ returns FAILURE and so does GenerateLPStructure in this case (see the left-hand side of Fig. 3). Hence there is no UP&I structure consistent with $\mathcal{E}$.

However the algorithm *does* successfully return a *CP-UI* structure. This is because, at node n_1, even though $(B, newRules)$is not UP-choosable w.r.t. $\mathcal{E}(n_1), \mathsf{Anc}(n_1), P$ for any appropriate choice of $newRules$ (i.e., of the form $B, \top :>$ in the UP-case), it *is* CP-choosable. Recall that to be CP-choosable, $newRules$ must contain a rule $B, u :>$ for each $u \in \underline{n}_1 = \{a, \bar{a}\}$, and in this case we may take $newRules = \{B, a : b > \bar{b}, B, \bar{a} : \bar{b} > b\}$. After this, since there is no $(\alpha, \beta) \in \mathcal{E}(n_1)$ such that $\alpha(B) = \beta(B)$, $\mathsf{generateUncondLabel}(\mathcal{E}(n_1), B)$ generates no labels and the algorithm terminates with the CP-UI structure on the right-hand side of Fig. 3.

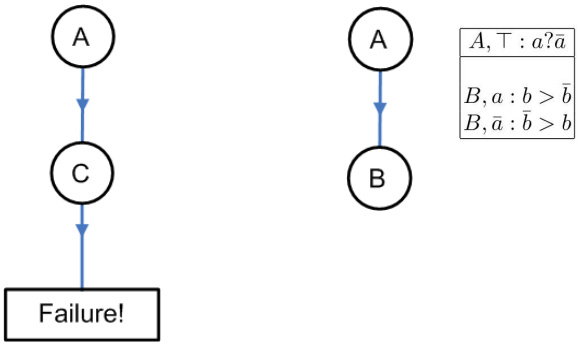

Fig. 3. Output structures for Example 2. *Left:* The output is failure for UP&I structures. *Right:* The output CP-UI structure.

[5] Note in fact A is really a completely uninformative choice here, since it does not decide any of the examples. A sensible heuristic for the algorithm - at least in the UP case - would be to disallow choosing any attribute X such that $\alpha(X) = \beta(X)$ for all examples. Such heuristics will be addressed in future work.

Example 3. Consider the following examples:

$$1.\ (ab\bar{c},\ abc)\ \ 2.\ (abc,\ a\bar{b}c)\ \ 3.\ (ab\bar{c},\ a\bar{b}\bar{c})\ \ 4.\ (\bar{a}\bar{b}c,\ \bar{a}b\bar{c})\ \ 5.\ (\bar{a}bc,\ \bar{a}\bar{b}\bar{c})$$

We will now use the algorithm to check if there is a CP&I structure consistent with these examples. We start at the root node n_0, and check whether $(A, newRules)$ is CP-choosable w.r.t. $\mathcal{E}, \emptyset, \emptyset$. As in the previous example, since $\alpha(A) = \beta(A)$ for all $(\alpha, \beta) \in \mathcal{E}$, we may label n_0 with A, and add preference rule $A, \top : a?\bar{a}$ to P, where "?" is some arbitrary preference between $a, \bar{a}$. Since we are now in the CI-case, algorithm generateCondLabels$(\mathcal{E}, A)$ is called, which generates an edge-label for each value x of A such that $\alpha(A) = \beta(A) = x$ for some $(\alpha, \beta) \in \mathcal{E}$, in this case both a (see, e.g., example 1 in $\mathcal{E}$) and $\bar{a}$ (see, e.g., example 4). Thus two edges from n_0 are created, labelled with $a, \bar{a}$ resp., leading to two new unlabelled nodes n_1 and m_1.

Following the right-hand branch leading to m_1 first (see Fig. 4), we have $\mathcal{E}(m_1) = \{(\alpha, \beta) \in \mathcal{E} \mid \alpha(A) = \beta(A) = \bar{a}\} = \{4, 5\}$. Here we first check if $(B, newRules)$ is CP-choosable w.r.t. $\mathcal{E}(m_1), \{A\}, P$. By definition of CP-choosable $newRules$ must be of the form $\{B, \bar{a} :>\}$. However due to the opposing preferences on their restriction to B exhibited by 4,5, we see there is no possible choice for $>$ here. Thus we have to consider C instead. Here we see $(C, \{C, \bar{a} : c > \bar{c}\})$ is CP-choosable, thus m_1 is labelled with C, and $C, \bar{a} : c > \bar{c}$ is added to P. Since generateCondLabel$(\mathcal{E}(m_1), C) = \emptyset$, no new nodes are created on this branch.

Now, moving back to n_0 and following the left-hand branch to node n_1, we have $\mathcal{E}(n_1) = \{(\alpha, \beta) \in \mathcal{E} \mid \alpha(A) = \beta(A) = a\} = \{1, 2, 3\}$. Checking B for CP-choosability first, we see $(B, \{B, a : b > \bar{b}\})$ is CP-choosable w.r.t. $\mathcal{E}(n_1), \{A\}, P$, thus n_1 is labelled with B and $B, a : b > \bar{b}$ added to P; generateCondLabel$(\mathcal{E}(n_1), B) = \{\{b\}\}$, thus one edge is generated, labelled with b, leading to new node n_2 with $\mathcal{E}(n_2) = \{(\alpha, \beta) \in \mathcal{E} \mid \alpha(\{A, B\}) = \beta(\{A, B\}) = ab\} = \{1\}$. For the last remaining attribute C on this branch we have $(C, \{C, ab : \bar{c} > c\})$ is CP-choosable w.r.t. $\mathcal{E}(n_2), \{A, B\}, P$. Thus the algorithm successfully terminates here, labelling n_2 with C and adding $C, ab : \bar{c} > c$ to P. The constructed CP&I structure in Fig. 4 is thus consistent with $\mathcal{E}$.

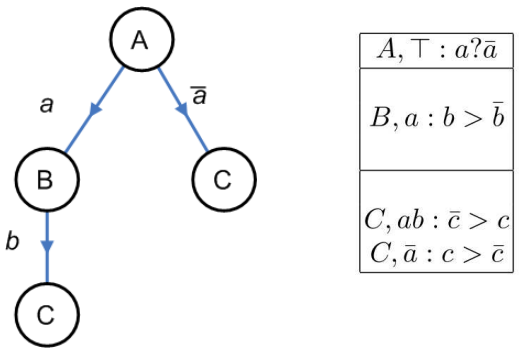

Fig. 4. Output CP&I structure for Example 3.

5.3 Complexity of model identification

The table in Fig. 5 gives the parameters for the greedy algorithm that solve five learning problems. In fact, the only problem that cannot be solved with this algorithm, as will be shown below, is the learning of a UP-CI structure without initial knwoldge of the preferences.

Proposition 5. *Using the right type of labels and the right choosability condtion and the right initial preference table, the algorithm returns, when called on a given set $\mathcal{E}$ of examples, a structure of the expected type, as described in the table of Fig. 5, consistent with $\mathcal{E}$, if such a structure exists*

learning problem	choosability	labels	initial P	structure type
CP&I	CP-choosable	conditional	$\emptyset$	CP&I
CP-UI	CP-choosable	uncond.	$\emptyset$	CP-UI
UP&I	UP-choosable	uncond	$\emptyset$	UP&I
FP-CI	UP-choosable	conditional	1 rule/attr.	UP-CI
FP-UI	UP-choosable	uncond	1 rule/attr.	UP-CI

Fig. 5. Parameters of the greedy algorithm for five learning problems

Proof (Sketch). The fact that the structure returned by the algorithm has the right type, depending on the parameters, and that it is consistent with the set of examples is quite straightforward. We now give the main steps of the proof of the fact that the algorithm will not return failure when there exists a structure of a given type consistent with $\mathcal{E}$.

Note first that given any node n of some LP-structure (T, P), labelled with X, if P_X denotes the set of rules that are applicable at n with respect to any $u \in \underline{n}$, then (X, P_X) is clearly choosable with respect to $\mathcal{E}(n)$, $\mathsf{Anc}(n)$ and P' the set of rules that are applicable at some node not in the subtree below n. So if we know in advance some LP-structure (T, P) consistent with a set $\mathcal{E}$ of examples, we can always construct it using the greedy algorithm, by choosing the "right" labels at each step.

Importantly, it can also be proved that if at some node n we choose another attribute X that is choosable, then there is some other LP-structure (T', P'), of the same type as (T, P), that is consistent with $\mathcal{E}$ and extends the current one; more precisely, (T', P') is obtained by modifying the subtree of T rooted at n, taking up X to the root of this subtree. Hence the algorithm cannot run into a dead end. This does not work in the UP-CI case, because taking an attribute upwards in the tree may require using a distinct preference rule, which may not be correct in other branches of the AI tree.

Corollary 1. *The problems of deciding if there exists a LP-structure of a given class consistent with a given set of examples over binary attributes have the following complexities:*

	FLP	*ULP*	*CLP*
UI	***P*** *(Dombi* et al. *, 2007)*	***P***	***P***
CI	***P***	***NP****-complete*	***P***

Proof (Sketch). For the CP&I, CP-UI, FP-CI, FP-UI and UP&I cases, the algorithm runs in polynomial time because it does not have more than $|\mathcal{E}|$ leaves, and each leaf cannot be at depth greater than n; and every step of the loop except (2b) is executed in linear time, whereas in order to choose an attribute, we can, for each remaining attribute X, consider the relation $\{(\alpha(X), \beta(X)) \mid (\alpha, \beta) \in \mathcal{E}(n)\}$ on $\underline{X}$: we can check in polynomial time if it has cycles, and, if not, extend it to a total strict relation over $\underline{X}$.

For the UP-CI case, one can guess a set of unconditional local preference rules P, of size linear in n, and then check in polynomial time (FP-CI) case if there exists a attribute importance tree T such that (T, P) is consistant with $\mathcal{E}$; thus the problem in **NP**. Hardness comes from a reduction from WEAK SEPARABILITY – the problem of checking if there is a CP-net without dependencies *weakly consistent* with a given set of examples – shown to be **NP**-complete by Lang & Mengin (2009).

5.4 Complexity of model approximation

In practice, a general problem in machine learning is that there is often no structure of a given type that is consistent with all the examples at the same time. It is then interesting to find a structure that is consistent with the most examples. Schmitt & Martignon (2006) have shown that finding a UI&LP-structure, with a fixed set of local preferences, that satisfies as many examples from a given set as possible, is **NP**-complete, in the case where all attributes are binary. We extend these results here.

Proposition 6. *The complexities of finding a LP-structure in a given class , which wrongly classifies at most k examples of a given set $\mathcal{E}$ of examples over* binary *attributes, for a given k, are as follows:*

	FLP	*ULP*	*CLP*
UI	***NP**-completeSchmitt & Martignon (2006)*	***NP**-complete*	***NP**-hard*
CI	***NP**-complete*	***NP**-complete*	***NP**-complete*

Proof (Sketch). These problems are in NP because in each case a witness is the LP-structure that has the right property, and such a structure need not have more nodes than there are examples. For the UP-CI case, the problem is already NP-complete for $k = 0$, so it is NP-hard. NP-hardness of the other cases follow from successive reductions from the case proved by Schmitt & Martignon (2006).

6 Conclusion and future work

We have proposed a general, lexicographic type of models for representing a large family of preference relations. We have defined six interesting classes of models where the attribute importance as well as the local preferences can be conditional, or not. Two of these classes correspond to the usual unconditional lexicographic orderings, and to a variant of Wilson's "Pre-Order Search Trees" (or POST) (2006). Interestingly, classes where preferences are conditional have an exponentional VC dimension.

We have calculated the cardinality of five of these six classes, and proved that the communication complexity for each class is not greater than the $\log$ of this cardinality, thereby generalizing a previous result by Dombi *et al.* (2007).

As for passive learning, we have proved that a greedy algorithm like the ones proposed by Schmitt & Martignon (2006); Dombi *et al.* (2007) for the class of unconditional preferences can identify a model in another four classes, thereby showing that the model identification problem is polynomial for these classes. We have also proved that the problem is NP-complete for the class of models with conditional attribute importance but unconditional local preferences. On the other hand, finding a model that minimizes the number of mistakes turns out to be NP-complete in all cases.

Our LP-structures are closely connected to decision trees. In fact, one can prove that the problem of learning a decision tree consistent with a set of examples can be reduced to a problem of learning a CP-CI LP structure. There remains to see if CP-CI structures can be as efficiently learnt in practice as decision trees.

In the context of machine learning, usually the set of examples to learn from is not free of errors in the data. Our greedy algorithm is quite error-sensitive and therefore not robust in this sense; it will even fail in the case of a collapsed version space. Robustness toward errors in the training data is clearly an important property of real world applications.

As future work, we intend to test our algorithms, with appropriate heuristics to guide the choice of variables a each stage. A possible heuristics would be the mistake rate if some unconditional structure is built below a given node (which can be very quickly done). Another interesting aspect would be to study mixtures of conditional and unconditional structures, with e.g. the first two levels of the structure being conditional ones, the remaining ones being unconditional (since it is well-known that learning decision trees with only few levels can be as good as learning trees with more levels).

Acknowledgements We thank the reviewers for helpful comments. We even borrowed from them some of the sentences in this concluding section.

References

Booth, Richard, Chevaleyre, Yann, Lang, Jérôme, Mengin, Jérôme, & Sombattheera, Chattrakul. 2009. *Learning various classes of models of lexicographic orderings*. Tech. rept. Institut de Recherche en Informatique de Toulouse.

Boutilier, Craig (ed). 2009. *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*.

Dimopoulos, Yannis, Michael, Loizos, & Athienitou, Fani. 2009. Ceteris Paribus Preference Elicitation with Predictive Guarantees. *In:* Boutilier (2009).

Dombi, József, Imreh, Csanád, & Vincze, Nándor. 2007. Learning Lexicographic Orders. *European Journal of Operational Research*, **183**, 748–756.

Koriche, Frédéric, & Zanuttini, Bruno. 2009. Learning conditional preference networks with queries. *In:* Boutilier (2009).

Lang, Jérôme, & Mengin, Jérôme. 2009. The complexity of learning separable ceteris paribus preferences. *In:* Boutilier (2009).

Schmitt, Michael, & Martignon, Laura. 2006. On the Complexity of Learning Lexicographic Strategies. *Journal of Machine Learning Research*, **7**, 55–83.

Wilson, Nic. 2006. An Effcient Upper Approximation for Conditional Preference. *In:* Brewka, Gerhard, Coradeschi, S., Perini, A., & Traverso, P. (eds), *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006)*.

Yaman, Fusun, Walsh, Thomas J., Littman, Michael L., & desJardins, Marie. 2008. Democratic Approximation of Lexicographic Preference Models. *In: Proceedings of the 25th International Conference on Machine Learning (ICML'08)*.