The Complexity of Learning Separable *ceteris paribus* Preferences

Jérôme Lang
LAMSADE
Université Paris-Dauphine
75775 Paris Cedex 16, France
lang@lamsade.dauphine.fr

Jérôme Mengin
IRIT
Université de Toulouse
31062 Toulouse Cedex, France
Jerome.Mengin@irit.fr

Abstract

We address the problem of learning preference relations on multi-attribute (or combinatorial) domains. We do so by making a very simple hypothesis about the dependence structure between attributes that the preference relation enjoys, namely separability (no preferential dependencies between attributes). Given a set of examples consisting of comparisons between alternatives, we want to output a separable CP-net, consisting of local preferences on each of the attributes, that fits the examples. We consider three forms of compatibility between a CP-net and a set of examples, and for each of them we give useful characterizations as well as complexity results.

1 Introduction

In many applications, especially electronic commerce, it is important 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). For instance, suppose we observe a user making choices about airplane tickets: (a) she prefers a KLM night flight landing at Heathrow to an Aeroflot day flight landing at Gatwick, (b) she prefers an Aeroflot night flight landing at Heathrow to a KLM day flight landing at Heathrow and (c) she prefers a KLM day flight landing at Heathrow to a KLM night flight landing at Heathrow. An intuitively correct explanation (among others) is that she prefers Aeroflot to KLM (unconditionally), day flights to night flights (unconditionally), and Heathrow to Gatwick, again unconditionally. This explanation enjoys the key property that the user's preference relation is separable: the preference over the values of each attribute is independent from the values of other attributes. Such an explanation allows for predicting that she will prefer an Aeroflot day flight landing at Heathrow to anything else, and an Aeroflot day flight landing at Heathrow to an Aeroflot day flight landing at Gatwick (but is not able to predict whether she will prefer an Aeroflot day flight landing at Gatwick or an Aeroflot night flight landing at Heathrow).

Now, if we observe later that she prefers an Aeroflot night flight landing at Gatwick to an Aeroflot day flight landing at Gatwick, the previous explanation fails, and more generally, so does any explanation where separability holds. A possible explanation (among others) could be that she prefers Aeroflot to KLM, Heathrow to Gatwick, day flights to night flights when flying on KLM and *vice versa* when flying on Aeroflot.

This raises several interesting questions: how can we decide whether it is possible that the user's preference relation is separable? If so, what are her local preference relations on every variable? If not, can we find the separable preference relation "maximally compatible" with the set of examples? How expensive are these computations? This paper aims at answering these questions (but it does not intend to find *non-separable* preference relations fitting a set of examples).

In this paper we focus on *passive learning*: the system has as input a set of preferences between alternatives, that the user has already stated (not necessarily without her being asked to), and one wants to reason about her preferences so as to predict choices she will make on new alternatives. This clearly differs from *preference elicitation* (see for instance [Chen and Pu, 2004; Braziunas and Boutilier, 2006] for surveys) where the system interacts with the user by asking her specific requests, or suggesting new alternatives to lead her to express a preference on the value of some attribute [Viappiani *et al.*, 2006], until she has found her target object or left the system.

Preferences over combinatorial domains have been investigated in detail in multiattribute decision theory (starting with Keeney and Raiffa 1976) and in Artificial Intelligence. Multiattribute decision theory has focused on *modelling* preferences, while AI has focused on designing languages for *representing* preferences as succinctly as possible and algorithms for finding optimal alternatives that are as fast as possible.

Classes of models and languages can be partitioned first according to whether they represent *ordinal* preferences (consisting in ranking the alternatives), and *numerical* preferences (e.g., utility functions). Here we focus on ordinal preferences. They have the advantage of often being easier to obtain from users.

Whereas there has been quite a lot of recent work on learning ranking functions and pairwise preferences over noncombinatorial sets of alternatives (see for instance Cohen *et al.* 1999, Hüllermeier *et al.* 1999), and whereas learning or eliciting utility functions over multiattribute domains has been considered in some places (Ha and Haddawy 1997 or Gonzales and Perny 2004, among others) there

have been fewer attempts at learning preferences from ordinal comparisons, notably: (a) learning lexicographic preferences [Schmitt and Martignon, 2006; Dombi *et al.*, 2007; Yaman *et al.*, 2008]; (b) a few works on learning CP-nets [Athienitou and Dimopoulos, 2007; Koriche and Zanuttini, 2009], (c) more general *ceteris paribus* preferences [Sachdev, 2007] or (d) utility functions learned from ordinal comparisons between logical formulas [Domshlak and Joachims, 2007]. These approaches are discussed in Section 6.

In both (a) and (b), the preference relation on the multiattribute domain is built from "local" preference relations on the attribute domains; these local orderings are then combined either in a lexicographic manner for (a), or, for (b), assuming that the local comparisons are lifted to comparisons of vectors of values when all other attributes have the same value (*ceteris paribus*). CP-nets allow dependencies between variables, and then providing local preferences on the values of one attribute depending on the values of other attributes.

In order to learn general CP-nets, given a set of examples, one would have to identify the dependency structure *and* the conditional preference tables. One could then search for the simplest CP-net compatible with the examples, or for the best trade-off between simplicity and compatibility with the examples. *However*, in this paper we focus on a simpler problem, namely, learning the simplest class of CP-nets: separable CP-nets (whose associated dependency graph is empty). There are several reasons for that. First, separable preferences are a very important class of preference relations on multi-attribute domains, as *e.g.* in economics or social choice (see for instance [Bradley *et al.*, 2005]), so that learning separable preferences deserves a study on its own. Second, as we will see, learning separable preferences is computationally hard – more than one would think at first sight, since *reasoning* with separable CP-nets can be done in linear time. Third, designing methods for learning non-separable preferences, as we said, is a challenging and important issue to be investigated, but it is also a very involved question that surely cannot be answered in a single paper. Focusing first on separable preferences is a nontrivial and necessary step towards learning more complex preferences. Lastly, it is well-known that a simpler structure can have better generalization properties than a more complex one that may fit the examples better.

In Section 2, we give some background on preferences on combinatorial domains and (separable) CP-nets. We then focus successively on three forms of compatibility between a CP-net and a set of examples: *weak compatibility* in Section 3, *strong compatibility* in Section 4 and *implicative compatibility* at the end of Section 3. We show that while it can be checked in polynomial time whether there exists a SCP-net implying a set of examples, determining whether there exists a SCP-net weakly (resp. strongly) compatible with a set of examples is NP-complete. Related work is discussed in Section 6 and a concluding discussion is given in Section 7.

2 The problem

We want to learn an ordering on objects, or *alternatives*, defined by their values for attributes of a set $\mathcal{V} = \{X_1, \ldots, X_n\}$. Each attribute X_i has a finite domain D_i. We denote by $D = D_1 \times \ldots \times D_n$ the set of all possible alternatives. If $\vec{x} \in D$, then $(\vec{x})_i$ denotes the value of $\vec{x}$ for the i-th attribute X_i. An attribute X_i is *binary* if D_i has two elements, which by convention we note x_i and $\overline{x_i}$. A *(strict) preference relation* $\succ$ is a strict order on D, *i.e.*, an irreflexive and transitive (theferore asymmetric) binary relation. If moreover $\succ$ is connected then $\succ$ is a *linear preference relation* (or *linear order* for short). Note that if we see a transitive relation $\succ$ as a graph over D (where there is an edge from $\vec{x}$ to $\vec{y}$ if $x \succ y$), then $\succ$ is irreflexive if and only if the graph is acyclic.

Suppose now that we have a finite set of *examples* $\mathcal{E}$, where each example is a pair of distinct alternatives $(\vec{x}, \vec{y})$ (also denoted, equivalently, by $\vec{x} \succ \vec{y}$) such that $\vec{x}$ is preferred to $\vec{y}$. In a recommender system for instance, these examples may have been recorded in the course of a user interaction with the system. Ideally, we would like to learn how to order every unordered pair of distinct alternatives $\{\vec{x}, \vec{y}\}$. Therefore, ideally, our target is a linear preference relation[1].

In this paper, we focus on learning *separable ceteris paribus preference structures* (SCP structures for short). SCP-structures are particular cases of *conditional preference networks*, or CP-nets [Boutilier *et al.*, 2004], obtained when the dependency graph between attributes is empty, therefore when every attribute is preferentially independent of all other attributes (CP-nets allow more generally the preference of an attribute to depend on the values of other attributes). A SCP structure on $\mathcal{V}$ is a collection $\mathcal{N}$ of linear orders $\succ_i$, one on each D_i. A *swap* is a pair of alternatives $\{\vec{x}, \vec{y}\}$ that differ in the value of one attribute only. From a SCP structure $\mathcal{N}$ we induce a *preference relation* $\succ_\mathcal{N}$ as follows. First, a linear order on D is said to satisfy $\mathcal{N}$ if for every attribute X_i and every $\vec{z}, \vec{z'}$ such that $\vec{z}$ and $\vec{z'}$ coincide on all attributes except X_i, $\vec{z} \succ \vec{z'}$ if and only if $(\vec{z})_i \succ_i (\vec{z'})_i$. Then, $\succ_\mathcal{N}$ is the intersection of all preference relations satisfying $\mathcal{N}$. Note that $\succ_\mathcal{N}$ is a partial order on D. Any linear order $\succ$ that satisfies a SCP structure is *separable*. Separability means that attributes are mutually preferentially independent. Note that if $\mathcal{N}$ is a SCP structure, then $\succ_\mathcal{N}$ is irreflexive, and thus can be extended to a linear order on D [Boutilier *et al.* 2004, Th. 1].

In most of the paper, for the ease of presentation we assume all attributed are binary, although our results apply to non binary attributes as well (as shown at the end of section 3). In this case, for any pair $(\vec{x}, \vec{y})$ of alternatives, let $\mathit{Diff}(\vec{x}, \vec{y}) = \{x_i \mid (\vec{x})_i = x_i \text{ and } (\vec{y})_i = \overline{x_i}\} \cup \{\overline{x_i} \mid (\vec{x})_i = \overline{x_i} \text{ and } (\vec{y})_i = x_i\}$; we can then use the following characterisation of $\succ_\mathcal{N}$ (a corollary of a Theorems 7 and 8 by Boutilier *et al.*, 2004):

Lemma 1 *Let $\mathcal{N}$ be a SCP structure over binary variables, and $\vec{y} \neq \vec{x}$. Then $\vec{x} \succ_\mathcal{N} \vec{y}$ if and only if $\mathcal{N}$ contains $x_i \succ \overline{x_i}$ for every $x_i \in \mathit{Diff}(\vec{x}, \vec{y})$ and $\overline{x_i} \succ x_i$ for every $\overline{x_i} \in \mathit{Diff}(\vec{x}, \vec{y})$.*

Example 1 *Consider three binary attributes A, B and C, with respective domains $\{a, \overline{a}\}$, $\{b, \overline{b}\}$ and $\{c, \overline{c}\}$, and let $\mathcal{N} = \{a \succ \overline{a}, b \succ \overline{b}, \overline{c} \succ c\}$. We have $ab\overline{c} \succ_\mathcal{N} \overline{a}\overline{b}c$, while $a\overline{b}c$*

[1] Our methodology and results would easily carry over to the problem of learning nonstrict preference relations (where indifference is allowed). We stick here to strict preference relation because the presentation is simpler.

and $\overline{a}bc$ are incomparable w.r.t. $\succ_{\mathcal{N}}$.

We said earlier that the target of our learning process should ideally be a linear order over D, but we have just seen that a SCP structure $\mathcal{N}$ does not in general correspond to such an order, since the preference relation $\succ_{\mathcal{N}}$ induced from $\mathcal{N}$ is generally not complete. Actually, $\succ_{\mathcal{N}}$ can be seen as the set of all its completions, that is, a SCP structure expresses a *set of linear preference relations*.

If an example $(\vec{x}, \vec{y})$ is a swap, then either $\vec{x} \succ_{\mathcal{N}} \vec{y}$ or $\vec{y} \succ_{\mathcal{N}} \vec{x}$, and clearly we would like $\mathcal{N}$ to be in agreement with the example, that is, for instance, such that $\vec{x} \succ_{\mathcal{N}} \vec{y}$. But if $(\vec{x}, \vec{y})$ is not a swap, there may be completions $\succ$ and $\succ'$ of $\succ_{\mathcal{N}}$ such that $\vec{x} \succ \vec{y}$ and $\vec{y} \succ' \vec{x}$. So we should start by discussing the possible ways of measuring to which extent a given SCP structure generalizes from a given set of examples.

Athienitou and Dimopoulos (2007) propose algorithms to learn CP-nets whose associated partial order contains all the examples – we say that it *implies* the examples. We argue here that this requirement may be too strong. To see this, consider the following example:

Example 2 *We have two binary attributes X_1 and X_2 (with domains $\{x_1, \overline{x_1}\}$ and $\{x_2, \overline{x_2}\}$), and the set of examples $\mathcal{E} = \{x_1x_2 \succ x_1\overline{x_2},\ x_1\overline{x_2} \succ \overline{x_1}x_2,\ \overline{x_1}x_2 \succ \overline{x_1}\overline{x_2}\}$.*

What do we expect to learn from the above set of examples $\mathcal{E}$? The transitive closure of $\mathcal{E}$ is the complete preference relation $x_1x_2 \succ x_1\overline{x_2} \succ \overline{x_1}x_2 \succ \overline{x_1}\overline{x_2}$. This preference relation is *separable* (the agent unconditionally prefers x_1 to $\overline{x_1}$ and x_2 to $\overline{x_2}$). The fact that $x_1\overline{x_2}$ is preferred to $\overline{x_1}x_2$ simply means that when asked to choose between X_1 and X_2, the agent prefers to give up X_2 (think of X_1 meaning "getting rich" and X_2 meaning "beautiful weather tomorrow"). Intuitively, since $\mathcal{E}$ is separable, we expect to output a structure $\mathcal{N}$ that contains $x_1 \succ \overline{x_1}$ and $x_2 \succ \overline{x_2}$. However, no SCP structure implies $\mathcal{E}$ (in fact, no CP-net implies $\mathcal{E}$, whatever the dependencies). The structure $\mathcal{N}$ induces a *partial* preference relation in which $x_1\overline{x_2}$ and $\overline{x_1}x_2$ are incomparable. More generally, no *ceteris paribus* structures can "explain" that $x_1 \succ \overline{x_1}$ is "more important" than $x_2 \succ \overline{x_2}$ (i.e., with no intermediate alternative). Therefore, if we look for a structure *implying* all the examples, we will simply output "failure". On the other hand, if we look for a SCP structure that is simply *consistent* with the examples, i.e. that does not imply the contrary of the examples, we will output $\mathcal{N}$.

The explanation is that when an agent expresses a CP-net, the preference relation induced by this CP-net *is not meant to be the whole agent's preference relation, but a subset (or a lower approximation) of it*. In other terms, when an agent expresses the CP-net $\mathcal{N}$, she simply expresses that she prefers x_1 to $\overline{x_1}$ *ceteris paribus* (i.e., for a fixed value of X_2) and similarly for the preference $x_2 \succ \overline{x_2}$; the fact that $x_1\overline{x_2}$ and $\overline{x_1}x_2$ are incomparable in $\mathcal{N}$ surely does not mean that the user really sees them incomparable, but, more technically, that CP-nets are not expressive enough for representing the missing preference $x_1\overline{x_2} \succ \overline{x_1}x_2$[2].

[2]If we want to do this, we have to resort to a more expressive language such as TCP-nets [Brafman *et al.*, 2006] or conditional preference theories [Wilson, 2004].

Therefore, rather than looking for a CP-net that implies the examples, we should rather look for a CP-net whose preference relation is consistent with the examples. A first way of understanding consistency is to require that the learned CP-net $\mathcal{N}$ is such that the examples are consistent with at least one preference relation extending $\mathcal{N}$. In some contexts it may even be too strong to require that one of the completions of $\succ_{\mathcal{N}}$ contains all the examples, in particular if they come from multiple users (given that we want to learn the generic preferences of a group of users), or a single user in different contexts:

Example 3 *Suppose that we learn that all users in a group unconditionally prefer x_1 to $\overline{x_1}$ and x_2 to $\overline{x_2}$, whereas their preferences between $x_1\overline{x_2}$ and $\overline{x_1}x_2$ may differ (think as x_1 and x_2 as, respectively, "being invited to a fine dinner" and "receiving a \$50 award"): then $\mathcal{E} \supseteq \{\overline{x_1}x_2 \succ x_1\overline{x_2},\ x_1\overline{x_2} \succ \overline{x_1}x_2\}$. $\mathcal{E}$ is clearly inconsistent, so there cannot be any preference structure whose ordering can be completed into a linear preference relation that contains $\mathcal{E}$. However, if $\mathcal{N} = \{x_1 \succ \overline{x_1}, x_2 \succ \overline{x_2}\}$, then each example in $\mathcal{E}$ is (individually) contained in at least one completion of $\succ_{\mathcal{N}}$.*

We now define three notions of compatibility of a SCP structure with a set of examples. (Note that these definitions also work for general CP-nets.)

Definition 1 *Let $\mathcal{N}$ be a SCP structure over $\mathcal{V}$. An example $(\vec{x}, \vec{y})$ is* implied by *$\mathcal{N}$ if $\vec{x} \succ \vec{y}$ for every completion $\succ$ of $\succ_{\mathcal{N}}$; it is* consistent with *$\mathcal{N}$ if there is a completion $\succ$ of $\succ_{\mathcal{N}}$ such that $\vec{x} \succ \vec{y}$. Furthermore, we will say that a set of examples $\mathcal{E}$ is:*

- implied by *$\mathcal{N}$ if every example of $\mathcal{E}$ is implied by $\mathcal{N}$;*
- globally (or strongly) compatible with *$\mathcal{N}$ if there is a completion $\succ$ of $\succ_{\mathcal{N}}$ such that for all $(\vec{x}, \vec{y}) \in \mathcal{E}$, $\vec{x} \succ \vec{y}$;*
- weakly compatible with *$\mathcal{N}$ if every example $(\vec{x}, \vec{y}) \in \mathcal{E}$ is individually consistent with $\mathcal{N}$.*

Lastly, we will say that $\mathcal{E}$ is implicatively / strongly / weakly separable *if it is implied by / strongly compatible / weakly compatible with at least one SCP structure.*

Clearly, strong compatibility implies weak compatibility, and if $\mathcal{E}$ is implied by $\mathcal{N}$, it is strongly compatible with $\mathcal{N}$.

3 Weak compatibility

We start by showing how searching for a SCP structure weakly compatible with a set of examples can be rewritten as a propositional satisfiability problem. In the case of binary attributes, consider the following translation from sets of examples to sets of clauses, the models of which correspond to SCP structures that are weakly consistent with the examples.

With each example $\vec{x} \succ \vec{y}$ we associate the clause $C_{\vec{x} \succ \vec{y}}$ that contains x_i iff $x_i \in Diff(\vec{x}, \vec{y})$ and $\neg x_i$ iff $\overline{x_i} \in Diff(\vec{x}, \vec{y})$. For instance, if $\vec{x} = x_1\overline{x_2}x_3x_4$ and $\vec{y} = \overline{x_1}x_2x_3\overline{x_4}$ then $Diff(\vec{x}, \vec{y}) = \{x_1, \overline{x_2}, x_4\}$ and $C_{\vec{x} \succ \vec{y}} = x_1 \vee \neg x_2 \vee x_4$. This clause expresses that x_1 is preferred to $\overline{x}_1$, or $\overline{x}_2$ is preferred to x_2, or x_4 is preferred to $\overline{x}_4$. (The explanation for this is that if a SCP structure $\mathcal{N}$ contains $\{\overline{x}_1 \succ x_1, x_2 \succ \overline{x}_2, \overline{x}_4 \succ x_4\}$, then by Lemma 1 we have $\vec{y} \succ_{\mathcal{N}} \vec{x}$, therefore $\mathcal{N}$ is not consistent with $\vec{x} \succ \vec{y}$.)

If $\mathcal{E}$ is a set of examples then $\Phi_{\mathcal{E}} = \{C_e \mid e \in \mathcal{E}\}$. We now define the following one-to-one correspondence between truth assignments over $\{x_1, \ldots, x_n\}$ and SCP structures over $\mathcal{V}$. If M is such a truth assignment, let $\mathcal{N}_M$ contain the preference $x_i \succ \overline{x_i}$ for every i such that $M \models x_i$ and the preference $\overline{x_i} \succ x_i$ for every i such that $M \models \neg x_i$. For instance, if $M(x_1) = \top$, $M(x_2) = \bot$, $M(x_3) = \bot$ and $M(x_4) = \top$ then $\mathcal{N}_M$ contains the preference tables $\{x_1 \succ \overline{x_1},\ \overline{x_2} \succ x_2,\ \overline{x_3} \succ x_3,\ x_4 \succ \overline{x_4}\}$.

Proposition 1 *Let M be an interpretation. A set of examples $\mathcal{E}$ is weakly consistent with $\mathcal{N}_M$ if and only if $M \models \Phi_{\mathcal{E}}$.*

Proof : We show that $\mathcal{N}_M \not\models \vec{y} \succ \vec{x}$ if and only if $M \models C_{\vec{x} \succ \vec{y}}$. Without loss of generality, let $\vec{x} = x_1 \ldots x_n$ and $\vec{y} = \overline{x_1} \ldots \overline{x_i} x_{i+1} \ldots x_n$. Then $M \not\models C_{\vec{x} \succ \vec{y}}$ if and only if $\mathcal{N}_M$ contains $\overline{x_1} \succ x_1, \ldots, \overline{x_i} \succ x_i$, which from Lemma 1 is equivalent to $\mathcal{N}_M \models \vec{y} \succ \vec{x}$. ■

Corollary 1 *$\mathcal{E}$ is weakly separable if and only if $\Phi_{\mathcal{E}}$ is satisfiable.*

As a consequence, weak separability comes down to a satisfiability test.

Example 4 *Consider three binary attributes A, B, C, and the set of examples*

$$\mathcal{E} = \{abc \succ \overline{a}bc,\ \overline{a}\overline{b}c \succ ab\overline{c},\ \overline{a}b\overline{c} \succ a\overline{b}c,\ \overline{a}bc \succ \overline{a}\overline{b}\overline{c}\}$$

$\Phi_{\mathcal{E}}$ has a unique model, corresponding to the SCP structure $\mathcal{N} = \{a \succ \overline{a}, b \succ \overline{b}, c \succ \overline{c}\}$. Therefore, $\mathcal{N}$ is the unique SCP structure weakly compatible with $\mathcal{N}$, which implies that $\mathcal{E}$ is weakly separable.

Proposition 2 *Deciding whether a set of examples over binary attributes is weakly separable is* NP*-complete.*

Proof : Membership follows directly from Corollary 1. For hardness, we use a reduction from 3SAT: let Φ be a set of non tautological, 3-clauses. For every $C = l_1 \vee l_2 \vee l_3$ in Φ we create an example $e_C = (\vec{x} \succ \vec{y})$ with $\vec{x} = \varepsilon_1.x_1\varepsilon_2.x_2 \ldots \varepsilon_n.x_n$ and $\vec{y} = \varepsilon'_1.x_1\varepsilon'_2.x_2 \ldots \varepsilon'_n.x_n$, where, for every i, $\varepsilon_i.x_i = \varepsilon'_i.x_i = x_i$, except that if $l_j = \neg x_i$ for some j, then $\varepsilon_i.x_i = \neg x_i$; and if $l_j = x_i$ for some j, then $\varepsilon'_i.x_i = \neg x_i$. Now, let $\mathcal{E}_\Phi = \{e_C \mid C \in \Phi\}$. For example, if $\Phi = \{x_1 \vee \neg x_2 \vee x_3, \neg x_1 \vee x_2 \vee x_4, x_2 \vee x_3 \vee x_4\}$ then $\mathcal{E}_\Phi = \{x_1\overline{x_2}x_3x_4 \succ \overline{x_1}x_2\overline{x_3}x_4, \overline{x_1}x_2x_3x_4 \succ x_1\overline{x_2}x_3\overline{x_4}, x_1x_2x_3x_4 \succ x_1\overline{x_2}\overline{x_3}\overline{x_4}\}$. We easily check that $\Phi_{\mathcal{E}_\Phi} = \Phi$, and therefore that Φ is satisfiable if and only if $\mathcal{E}_\Phi$ is weakly separable. ■

The generalization to non-binary domains is not difficult. Instead of having one propositional symbol per attribute, we have one propositional symbol for every pair of values of every attribute. For instance, if we have an attribute X whose domain is $\{d_1, d_2, d_3\}$ then we have the three propositional symbols $d_1 \succ d_2$, $d_1 \succ d_3$ and $d_2 \succ d_3$ ($d_2 \succ d_1$ being equivalent to $\neg(d_1 \succ d_2)$, etc.). For instance, if Y is a binary attribute, the clause corresponding to the example $d_1 y \succ d_2 \overline{y}$ is $(d_1 \succ d_2) \vee \neg y$. The main difference with the binary case is the transitivity requirement. Let $Trans = \bigwedge_{X_i \in \mathcal{V}} Trans_{X_i}$ be the propositional formula expressing transitivity – for instance, for $D_1 = \{d_1, d_2, d_3\}$ we have $Trans_{X_1} = (d_1 \succ d_2 \wedge d_2 \succ d_3 \rightarrow d_1 \succ d_3) \wedge \ldots$. Note that $Trans$ is polynomially long. The one-to-one correspondence between interpretations and SCP structures now works only for interpretations satisfying $Trans$: $M \models \Phi_{\mathcal{E}} \wedge Trans$ if and only if $\mathcal{N}_M$ is weakly consistent with $\mathcal{N}_M$. As a consequence, $\mathcal{E}$ is weakly separable if and only if $\Phi_{\mathcal{E}} \wedge Trans$ is satisfiable.

This translation shows that weak consistency for non-binary attributes is in NP, hence it is NP-complete.

Now, as soon as $\mathcal{E}$ becomes large with respect to the number of attributes n, the chances that $\mathcal{E}$ is weakly consistent become low. In this case, we may want to determine a SCP structure that is weakly compatible with as many examples of $\mathcal{E}$ as possible. Applying Corollary 1 to subsets of $\mathcal{E}$, we get that this problem amounts to solving a MAXSAT problem. The SCP structure that best fits a set of examples (in the sense of weak compatibility) corresponds to the interpretation maximizing the number of clauses from $\Phi_{\mathcal{E}}$ satisfied. This extends to nonbinary variables, with the difference that the clauses in $Trans$ are protected.

As a consequence of Propositions 1, we can reuse algorithms for MAXSAT for computing an SCP structure that best fits a set of examples, as well as polynomial approximation schemes. Using the same kind of translation, we can easily prove that if all variables are binary and all examples in $\mathcal{E}$ differ at most on two variables, then deciding whether $\mathcal{E}$ is consistent can be done in polynomial time (however, the corresponding optimization problem remains NP-hard).

4 Strong compatibility

Characterizing strong compatibility is less easy. The difference between weak and strong compatibility is that while in weak compatibility we look for a SCP structure which is consistent with each individual example in $\mathcal{E}$, in strong compatibility we look for a SCP structure which is consistent with the whole set of examples $\mathcal{E}$.

Example 4, continued *$\mathcal{E}$ is not strongly compatible with $\mathcal{N}$, because $\mathcal{E} \cup \succ_{\mathcal{N}}$ has the following cycle:*

$$a\overline{b}c \succ_{\mathcal{N}} \overline{a}\overline{b}c \succ_{\mathcal{E}} ab\overline{c} \succ_{\mathcal{N}} \overline{a}b\overline{c} \succ_{\mathcal{E}} a\overline{b}c$$

Since $\mathcal{E}$ is not strongly compatible with any other SCP structure than $\mathcal{N}$ (because $\mathcal{N}$ is the unique SCP structure weakly compatible with $\mathcal{N}$), $\mathcal{E}$ is not strongly separable[3].

Note that all alternatives of the cycle on the example above appear in $\mathcal{E}$. More generally, if we denote by $D(\mathcal{E})$ the set of alternatives of D that appear in $\mathcal{E}$, we have the following characterisation of strong compatibility:

Proposition 3 *$\mathcal{E}$ is strongly compatible with $\mathcal{N}$ if and only if the restriction of $\succ_{\mathcal{N}} \cup \mathcal{E}$ to $D(\mathcal{E})$ is acyclic.*

Proof : By definition, if $\mathcal{E}$ is strongly compatible with $\mathcal{N}$ then $\succ_{\mathcal{N}}$ can be completed into a linear order containing $\mathcal{E}$, therefore $\succ_{\mathcal{N}} \cup \mathcal{E}$ is acyclic, and so is its restriction to $D(\mathcal{E})$. For the converse, assume that $\mathcal{E}$ is not strongly compatible with $\mathcal{N}$: $\succ_{\mathcal{N}} \cup \mathcal{E}$ cannot be extended to a linear order, this means that it is not irreflexive, so it contains a cycle Γ. Since SCP structures are consistent, Γ must contain at least one

[3]Note that $\mathcal{E}$ is both weakly consistent and does not contain any cycles as it was the case for Example 3, yet is not strongly consistent.

edge of $\mathcal{E}$. We can transform Γ into a cycle Γ' whose vertices are all in $D(\mathcal{E})$: simply replace, as long as it is possible, any sequence of vertices in Γ of the form $x \succ_{\mathcal{N}} y \succ_{\mathcal{N}} z$ by the vertex $x \succ_{\mathcal{N}} z$ (which holds because $\succ_{\mathcal{N}}$ is transitive). Now, every edge in Γ' is either in $\mathcal{E}$ or adjacent to two edges in $\mathcal{E}$ (recall it cannot be reduced to a single reflexive edge of $\succ_{\mathcal{N}}$ because $\succ_{\mathcal{N}}$ is irreflexive), so its extremities are in $D(\mathcal{E})$. So Γ' is a cycle of the restriction of $\succ_{\mathcal{N}} \cup \mathcal{E}$ to $D(\mathcal{E})$. ■

Since the restriction of $\succ_{\mathcal{N}} \cup \mathcal{E}$ to $D(\mathcal{E})$ has at most $2.|\mathcal{E}|$ vertices, checking if it possesses a cycle can be done in polynomial time, so checking whether $\mathcal{E}$ is strongly compatible with $\mathcal{N}$ is in P. Thus, since the size of a SCP-structure is linear in the number of variables, checking whether $\mathcal{E}$ is strongly separable is in NP.

Proposition 4 *Checking whether $\mathcal{E}$ is strongly separable is* NP*-complete.*

Proof : Hardness comes from the following reduction from WEAK SEPARABILITY. Let $\mathcal{E} = \{\vec{x}_1 \succ \vec{y}_1, \ldots, \vec{x}_m \succ \vec{y}_m\}$ be a set of examples, built on a set of propositional symbols X. We map $\mathcal{E}$ to the following set of examples $\mathcal{E}'$, built on the set of attributes $X \cup \{P_1, \ldots, P_m\}$. For every $e_i = \vec{x}_i \succ \vec{y}_i$ in $\mathcal{E}$ we create the example
$e'_i : \vec{x}_i \overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m \succ \vec{y}_i \overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m$.

We claim that $\mathcal{E}' = \{e'_i \mid e_i \in \mathcal{E}\}$ is strongly separable if and only if $\mathcal{E}$ is weakly separable. Assume $\mathcal{E}$ is weakly separable: there exists a SCP structure $\mathcal{N}$ consistent with each e_i in $\mathcal{E}$. Let $\mathcal{N}' = \mathcal{N} \cup \{p_1 \succ \overline{p}_1, \ldots, p_n \succ \overline{p}_n\}$. For every i, because $\mathcal{N}$ is consistent with $\vec{x}_i \succ \vec{y}_i$, there exists a linear order $\succ_i$ extending $\succ_{\mathcal{N}}$ and containing $\vec{x}_i \succ \vec{y}_i$. Let $\succ$ be the preference relation defined by $\vec{x}\vec{p} \succ \vec{y}\vec{p'}$ if and only if one of the following conditions hold:

(a) $\vec{p}$ is lexicographically larger than $\vec{p'}$ (denoted by $\vec{p} >_{lex} \vec{p'}$), that is, there exists an $i \leq m$ such that $\vec{p}$ contains p_i, $\vec{p'}$ contains $\overline{p}_i$, and for all $j < i$, $\vec{p}$ contains p_j if and only if $\vec{p'}$ contains p_j.

(b) $\vec{p} = \vec{p'} = \overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m$ for some i, and $\vec{x} \succ_i \vec{y}$;

(c) $\vec{p} = \vec{p'}$, $\vec{p}$ is not of the form $\overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m$ for some i, and $\vec{x} \succ^* \vec{y}$, where $\succ^*$ is any preference relation extending $\succ_N$.

It can be checked that $\succ$ is a complete preference relation $\succ$ extending $\succ_{\mathcal{N}'}$, and containing $\mathcal{E}'$, which implies that $\mathcal{E}'$ is strongly consistent with $\mathcal{N}'$, and that $\mathcal{E}'$ is strongly separable.

Conversely, assume $\mathcal{E}'$ is strongly separable. Let $\mathcal{N}'$ be strongly consistent with $\mathcal{E}'$, and $\succ$ extending $\succ_{\mathcal{N}'}$ and containing $\mathcal{E}'$. For every $i \leq m$, define $\succ_i$ as follows: $\vec{x} \succ_i \vec{y}$ *iff* $\vec{x}\overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m \succ \vec{y}\overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m$. Then, for every i: $\succ_i$ is a complete strict order on 2^X; $\succ_i$ extends $\succ_{\mathcal{N}}$; and $\succ_i$ contains $\vec{x}_i \succ \vec{y}_i$, because $\succ$ contains $\vec{x}_i \overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m \succ \vec{y}_i \overline{p}_1 \ldots \overline{p}_{i-1} p_i \overline{p}_{i+1} \ldots \overline{p}_m$. Therefore, for every i we have found an extension $\succ_i$ of $\succ_{\mathcal{N}}$ containing $\vec{x}_i \succ \vec{y}_i$, which implies that $\mathcal{E}$ is weakly compatible with $\mathcal{N}$, and that $\mathcal{E}$ is weakly separable. ■

Note that although weak and strong separability have the same complexity, weak consistency enjoys the nice property that there is a simple solution-preserving translation into SAT (the models of $\Phi_{\mathcal{E}}$ correspond bijectively to the CP-nets that are weakly consistent with $\mathcal{E}$), which allows weak consistency to be computed in practice using algorithms for SAT. In order to compute a SCP structure strongly consistent with $\mathcal{E}$, we can generate structures $\mathcal{N}$ weakly consistent with $\mathcal{E}$, and test for acyclicity of $\succ_{\mathcal{N}} \cup \mathcal{E}$ using graph algorithms.

5 Implicative compatibility

It is easy to characterize whether there exists a SCP structure that implies $\mathcal{E}$. In the case of binary variables, with each example $\vec{x} \succ \vec{y}$ we associate the conjunction of literals $\Gamma_{\vec{x} \succ \vec{y}} = \neg C_{\vec{y} \succ \vec{x}}$. Let $\Gamma_{\mathcal{E}} = \bigwedge\{\Gamma_e \mid e \in \mathcal{E}\}$. Using Lemma 1, we can prove that $M \models \Gamma_{\mathcal{E}}$ if and only if $\mathcal{N}_M$ implies $\mathcal{E}$. Searching for a model of a conjunction of literals can be done in polynomial time, thus the problem of checking if a set of examples is implicatively separable is in P.

Although implicative separability is easier than strong or weak separability, as argued before, we believe that implicative separability is much too strong.

6 Related work

There is an active stream of research on preference learning, which focuses on many facets of preference learning, such as label ranking [Hüllermeier *et al.*, 2008] or learning to rank objects [Cohen *et al.*, 1999], but so far the focus is laid on preferences on simple, noncombinatorial domains[4].

Learning numerical utility functions on combinatorial domains has been addressed in *e.g.* [Ha and Haddawy, 1997; Gonzales and Perny, 2004]. Recently, Domshlak and Joachims [2007] proposed a method to learn a utility function from general preference statements between logical formulas. As for learning *ordinal* preferences on multiattribute domains, an approach that follows a methodology close to ours (but for a very different class of preference structures) is the work on learning lexicographic preferences on multiattribute domains [Schmitt and Martignon, 2006; Dombi *et al.*, 2007; Yaman *et al.*, 2008]. A significant difference is that they learn the full preference relation, while we learn only the local preference relations. Of course, learning the full preference relation gives a richer information, but this comes with a price, namely, the hypothesis they make (lexicographic preferences) is highly restrictive. Lexicographic preferences are a very specific, special form of preferences that is not often met in practice. Note that every lexicographic preference is separable, but the set of separable preferences is much larger. Interestingly, the complexity results are similar.

Koriche and Zanuttini [2009] show that it is possible to efficiently elicitate acyclic CP-nets using simple queries of the form $\vec{x} \succ^? \vec{y}$ where $\vec{x}$ and $\vec{y}$ differ in the value of one variable only. Passive learning of CP-nets has been considered by Athienitou and Dimopoulos [2007]. As argued above, their

[4]Note that multiattribute domains are sometimes considered, but with a very different role: for instance, in label ranking one learns a preference relation on a simple domain, given some information under the form of a vector or attribute values (for instance, predict a user's ranking on a set of candidates at an election given his/her sex, age and profession).

focus on implicative compatibility limits the applicability of the approach. Sachdev [2007] proposes to learn a preference theory, in the sense of Doyle *et al.* [1991], consistent with a set of examples. He assumes that one is given (or has access to) the complete set of examples corresponding to an existing preference theory that one is trying to induce, which practically limits its applicability.

7 Conclusion and further work

Our contribution is twofold: we have seen that finding a separable *ceteris paribus* preference relation (or equivalently, finding a separable CP-net) compatible with a set of examples is not straighforward to define, due to the fact that, unlike lexicographicity, separability does not permit to lift the local preference relation into a complete preference relation on the multiattribute domain. We have proposed and discussed three different forms of compatibility between a separable CP-net and a set of examples. For each of these, we have identified its complexity and given characterizations that can be used for computing a separable CP-net that fits best a set of examples. Although implicative separability is easier than strong or weak separability, as argued before, implicative separability is far less interesting than the other two notions.

Now, our definitions of implicative, strong and weak compatibility would work for any CP-net. So far our characterizations and complexity results for each of these three forms of consistency concern only the case of separable preferences, where the dependency graph has no edge. A long-term challenge would consist in investigating methods for learning CP-nets with any structure, focusing first on simple structures (*e.g.*, hypertrees). A first step would consist in learning CP-nets with a *fixed* graph: given a set of examples $\mathcal{E}$ and a dependency graph G, output the CP-net maximizing G-compatibility (w.r.t. any of the three forms) with $\mathcal{E}$. Unfortunately, this is not simple. Our translation of examples into propositional clauses can be extended, using, for each binary attribute, one propositional variable for every combination of values of its parents. For instance, if A has no parent and is the only parent of B, we use three propositional variables: a, $(a\!:\!b)$ and $(\overline{a}\!:\!b)$, where, for instance, $(a\!:\!b)$ assigned to true corresponds to the entry $a\!:\!b\succ\overline{b}$ in the preference table for B. Weak compatibility can then be encoded using, for each example $\vec{x}\succ\vec{y}$, one clause for each sequence of swaps from $\vec{y}$ to $\vec{x}$, expressing that at least one step of this sequence is blocked. This ensures that $\vec{y}\succ_{\mathcal{N}}\vec{x}$ does not hold. But there can be exponentially many such clauses, even in the case where the number of parents in the dependency graph is bounded, which considerable limits the applicability of this translation.

Acknowledgements

Many thanks to Richard Booth, Yann Chevaleyre, Kevin Garcia, Peter Haddawy, Frédéric Koriche, Mathieu Serrurier, Chattrakul Sombatheera and Bruno Zanuttini for helpful discussions. Anonymous referees also gave helpful comments. This research has been partially supported by the project ANR-05-BLAN-0384 "Preference Handling and Aggregation in Combinatorial Domains".

References

F. Athienitou and Y. Dimopoulos. Learning CP-networks: a preliminary investigation. In *Proc. 3rd Multidisciplinary Work. on Advances in Preference Handling (PREF'07)*, 2007.

C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets: a tool for representing and reasoning with conditional ceteris paribus preference statements. *JAIR*, 21:135–191, 2004.

W. J. Bradley, J. K. Hodge, and D.M. Kilgour. Separable discrete preferences. *Mathematical Social Science*, 49(3):335–353, 2005.

R. Brafman, C. Domshlak, and S. Shimony. On graphical modeling of preference and importance. *JAIR*, 25:389–424, 2006.

D. Braziunas and C. Boutilier. Elicitation of factored utilities. *AI Magazine*, 29 (4), 2008.

L. Chen and P. Pu. Survey of preference elicitation methods. Tech. Rep. 200467, EPFL, 2004.

W. Cohen, R. Schapire, and Y. Singer. Learning to order things. *JAIR*, 10:243–270, 1999.

J. Dombi, C. Imreh, and N. Vincze. Learning lexicographic orders. *Eur. J. of Operational Research*, 183:748–756, 2007.

C. Domshlak and T. Joachims. Efficient and non-parametric reasoning over user preferences. *User Modeling and User-Adapted Interaction (UMUAI)*, 17(1-2):41–69, April 2007.

J. Doyle, Y. Shoham, and M. Wellman. A logic of relative desire (preliminary report). In *Proc. ISMIS '91*, 16–31. 1991.

C. Gonzales and P. Perny. GAI networks for utility elicitation. In *Proc. KR'04*, pages 224–233. AAAI Press, 2004.

V. Ha and P. Haddawy. Problem-focused incremental elicitation of multi-attribute utility models. In *Proc. UAI-97*, pages 215–222, 1997.

E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker. Label ranking by learning pairwise preferences. *AIJ*, 172:1897–1917, 2008.

R. Keeney and H. Raiffa. *Decision with Multiple Objectives: Preferences and Value Trade-offs*. Wiley, 1976.

F. Koriche and B. Zanuttini. Learning conditional preference networks with queries, 2009. In *Proc. IJCAI'09*.

M. Sachdev. On learning of ceteris paribus preference theories. Master's thesis, Graduate Faculty of North Carolina State University, 2007.

M. Schmitt and L. Martignon. On the complexity of learning lexicographic strategies. *J. Mach. Learn. Res.*, 7:55–83, 2006.

P. Viappiani, B. Faltings, and P. Pu. Preference-based search using example-critiquing with suggestions. *JAIR*, 27:465–503, 2006.

N. Wilson. Extending CP-Nets with Stronger Conditional Preference Statements. In *AAAI-04*, pages 735–741, 2004.

F. Yaman, T. Walsh, M. Littman, and M. desJardins. Democratic approximation of lexicographic preference models. In *Proc. ICML '08*, pages 1200–1207, 2008.