Computational Techniques for a Simple Theory of Conditional Preferences

Nic Wilson
Cork Constraint Computation Centre
Department of Computer Science
University College Cork
Cork, Ireland
n.wilson@4c.ucc.ie
Telephone: +353 21 4205954
Fax: +353 21 4205369

Abstract

A simple logic of conditional preferences is defined, with a language that allows the compact representation of certain kinds of conditional preference statements, a semantics and a proof theory. CP-nets and TCP-nets can be mapped into this logic, and the semantics and proof theory generalise those of CP-nets and TCP-nets. The system can also express preferences of a lexicographic kind. The paper derives various sufficient conditions for a set of conditional preferences to be consistent, along with algorithmic techniques for checking such conditions and hence confirming consistency. These techniques can also be used for totally ordering outcomes in a way that is consistent with the set of preferences, and they are further developed to give an approach to the problem of constrained optimisation for conditional preferences.

Keywords: conditional preferences, comparative preferences, *ceteris paribus* preferences, CP-nets, TCP-nets, constrained optimisation, lexicographic preferences.

1 Introduction

The formalism *CP-nets* (Boutilier, Brafman, Hoos, & Poole, 1999; Boutilier, Brafman, Domshlak, Hoos, & Poole, 2004a) is designed for compactly expressing conditional comparative preferences in multivariate problems. A CP-net involves statements of the form: $u : x > x'$, where x, x' are values of a variable X and u is an assignment to a set of variables U (called the *parents of* X). The interpretation is that, given u, x is (strictly) preferred to x', *all else being equal* (*ceteris paribus*); that is, for all assignments s to the other variables S, sux is preferred to sux', where e.g., sux is the outcome (complete assignment) α such that $\alpha(X) = x$, $\alpha(U) = u$ and $\alpha(S) = s$. The statement therefore compactly represents exponentially many preferences between outcomes. This is a *conditional* preference, since the preference between values of X is conditional on the values of other variables U. It represents *comparative* preferences, in that the preference statements relate directly to the ordering between outcomes; this is in contrast to many theories of preference which assign some form of grade to outcomes, and outcomes are compared by comparing their grades. Comparative preference statements can be easier to reliably elicit: often it is easier to judge that one alternative is preferred to another than it is to allocate particular grades of preference to the alternatives.

Another key feature of CP-nets and related languages is the *ceteris paribus* aspect of the interpretation. If someone tells us they'd prefer a green car to a white car, they wouldn't usually mean that they'd prefer any green car to any white car; a *ceteris paribus* interpretation, that any green car is preferred to a car which is similar except being white, seems much more natural. However, this will tend to lead to quite weak inferences, and a user will sometimes want to express much stronger statements such as those of the form: *x is preferred to x' irrespective of the values of other variables*, where the variable X is the most important variable, and, for example, x' represents a value that should be avoided if at all possible.

This paper develops a formalism along similar lines to CP-nets, but where a richer language of preference statements can be expressed: stronger conditional preference statements as well as the usual CP-nets *ceteris paribus* statements; The language consists of statements of the form $u : x > x' [W]$ (where W is a subset of S), which represents that for all assignments w, w' to W and assignments t to $S - W$, $tuxw$ is preferred to $tux'w'$. So, given u and any t, x is preferred to x' irrespective of the values of W. CP-nets *ceteris paribus* statements are represented by such statements with $W = \emptyset$, and the strong conditional preference statement in the previous paragraph corresponds to $\top : x > x' [V - \{X\}]$, where V is the set of all variables. As in CP-nets, and their extension TCP-nets (Brafman & Domshlak, 2002; Brafman, Domshlak, & Shimony, 2006), this is a compact representation: each statement typically corresponds to many preferences between outcomes.

The next section introduces the new formalism, which can be viewed as a simple logic of conditional preferences. A *cp-theory* Γ has an associated preference relation $>_\Gamma$ on outcomes; Γ can be considered to be a compact repre-

sentation of $>_\Gamma$. A semantics is given and also a complete proof theory, based on 'swapping sequences', which is a natural generalisation of flipping sequences in CP-nets and TCP-nets. Section 3 examines the relative expressivity of the language as compared with CP-nets. It shows how CP-net orderings (Section 3.1) and TCP-net orderings (Section 3.2) can be represented within the language; however, this stronger kind of preference statement, which can be used, for example, to construct a lexicographic order on outcomes, is not expressible within the languages of CP-nets or TCP-nets (see Sections 3.3 and 3.4). Section 3.5 illustrates that the *ceteris paribus* statements of CP-nets tend to be rather weak, by showing how hard it is for a CP-net to generate a total order on outcomes.

Sections 4, 5, 6 and 7, are all concerned with the inter-related topics of determining consistency of a cp-theory, totally ordering sets of outcomes, and constrained optimisation. Most of the work on CP-nets and TCP-nets has assumed a very strong acyclicity property on the variables (though see (Domshlak & Brafman, 2002; Domshlak, 2002)); here we generally make much weaker assumptions, which is desirable since natural sets of conditional preference statements can easily fail to be acyclic in this sense. A necessary condition for consistency is derived, "local consistency" (Section 4.1), and some sufficient conditions; determining whether these conditions hold is much less hard than determining consistency.

A cp-theory is consistent if and only if there exists a strict total order on outcomes that satisfies it. We focus on a particular kind of strict total order: one generated by a complete search tree (or "cs-tree", see Section 4.2), as used in backtracking search for solutions of a constraint satisfaction problem; the associated strict total order is the order in which outcomes are visited by such a search tree. We derive various sufficient conditions for a cs-tree to satisfy a cp-theory. If we can show that there exists a cs-tree satisfying a cp-theory Γ, then we have proved that Γ is consistent. Furthermore, we can use such a satisfying cs-tree for totally ordering sets of outcomes, and in a constrained optimisation algorithm (Section 4.4), making use of an upper approximation of the preference relation, i.e., a relation on outcomes that extends the preference relation.

For the fully acyclic case, i.e., when the graph formed by dependencies and importance is acyclic, defining a satisfying cs-tree is straightforward, as shown in Section 5; this implies that Γ is consistent if and only if it is locally consistent, with the latter condition often being very easy to check. For more general cases, the situation is more complicated, and in Sections 6 and 7 we derive more complex methods for constructing satisfying cs-trees. Section 6 considers weaker forms of acyclicity for cp-theories, that we call strong conditional acyclicity (Section 6.1) and cuc-acyclicity (Section 6.2), and which are sufficient conditions for a cp-theory to be consistent. A polynomial upper approximation is derived for cuc-acyclic cp-theories.

Proving consistency of a cp-theory by explicitly giving a cs-tree that satisfies the cp-theory will typically not be feasible, since the cs-tree is an exponentially large object. However, cs-trees can be defined in a compact way based on

implicit representations of the variable and value orderings; defining the value ordering is easy, given that the cp-theory is locally consistent. Section 7 defines a compact computational structure and associated techniques for defining the variable orderings of a cs-tree satisfying the cp-theory; this can be used for confirming consistency, ordering outcomes and constrained optimisation.

Section 8 discusses related work, Section 9 concludes, and the appendix contains most of the proofs.

2 A logic of conditional preferences

In this section, a simple logic of conditional preferences is defined, with a language, semantics and a kind of proof theory. As we will see in Sections 3.1 and 3.2, CP-nets and TCP-nets can be expressed within this language. The logic has a somewhat restrictive language, but the restrictions entail some nice properties, generalising properties of CP-nets.

After giving some basic definitions of ordering relations in Section 2.1, we define cp-theories and their associated preference relations in Section 2.2. A semantics (Section 2.3) and a proof theory (Section 2.4) are defined, with a completeness result (Theorem 1).

2.1 Ordering relations

In this section we give some basic definitions and properties of ordering relations that will be used throughout the paper.

A binary relation $\succ$ on a set Ω is defined to be a subset of $\Omega \times \Omega$; the notations "$(a, b) \in \succ$" and "$a \succ b$" are used interchangeably. Since binary relations are sets, we can talk about the intersection and union of them, and containment of one by another. So, in particular, if $\succ$ and $\succ'$ are (binary) relations on Ω then $\succ \subseteq \succ'$ holds if and only if $a \succ b$ implies $a \succ' b$. We may also say in this case that $\succ'$ *contains* $\succ$, or $\succ'$ *extends* $\succ$.

Let $\succ$ be a binary relation on a set Ω. $\succ$ is said to be *reflexive* if $a \succ a$ for all $a \in \Omega$. It is said to be *irreflexive* if for all $a \in \Omega$ it is not the case that $a \succ a$. Relation $\succ$ is said to be *transitive* if for all $a, b, c \in \Omega$, if $a \succ b$ and $b \succ c$ both hold then $a \succ c$ holds. For any relation $\succ$, there exists a unique (set-wise) minimal transitive relation R containing $\succ$ (which is equal to the intersection of all transitive relations on Ω containing $\succ$). R is known as the *transitive closure of* $\succ$.

Irreflexive relation $\succ$ is said to be *acyclic* if its transitive closure is irreflexive, i.e., if there exists no cycle $a \succ a' \succ a'' \succ \cdots \succ a$. Relation $\succ$ is said to be a *strict partial order* if it is transitive and irreflexive. A strict partial order is therefore acyclic. A *strict total order* $\succ$ is a strict partial order such that for all different $a, b \in \Omega$ either $a \succ b$ holds or $b \succ a$ holds.

We summarise some well-known properties of ordering relations (e.g., part (ii) generalised to infinite sets is Szpilrajn's Extension Theorem (Marczewski, 1930)).

Lemma 1 *Let $\succ$ be a binary relation on finite set Ω. Then the following properties hold.*

(i) Suppose that $\succ$ is a strict partial order on Ω, and that $\alpha, \beta \in \Omega$ are such that it is not the case that $\alpha \succ \beta$. Then $\succ \cup \{(\beta, \alpha)\}$ is acyclic.

(ii) If $\succ$ is a strict partial order then it can be extended to a strict total order, that is, there exists some strict total order $\succ'$ on Ω with $\succ \subseteq \succ'$ (i.e., $\alpha \succ \beta \Rightarrow \alpha \succ' \beta$).

(iii) If $\succ$ is irreflexive and acyclic then it can be extended to a strict total order on Ω.

(iv) Suppose that $\succ$ is irreflexive. Then $\succ$ is acyclic if and only if its transitive closure is irreflexive if and only if there exists a strict total ordering extending it.

(v) If $\succ$ is a strict partial order then $\succ$ is equal to the intersection of all strict total orders extending it.

2.2 cp-theories and their associated preference relations

In this section we define a formalism for compactly expressing comparative preferences. Before defining cp-theories, we first introduce our notation for outcomes and assignments.

Variables, tuples and outcomes. Let V be a set of variables; for each $X \in V$ let $\mathit{Domain}(X)$ be the set of possible values of X. For subset of variables $U \subseteq V$, we use the notation $\underline{U} = \prod_{X \in U} \mathit{Domain}(X)$ to represent the set of possible assignments to U. Formally, $\underline{U}$ is the set of functions on U which, for each $X \in U$, assign a value of X to variable X.[1]

The assignment to the empty set of variables is written $\top$. A *complete tuple/assignment* or *outcome* is an element of $\underline{V}$, i.e., an assignment to all the variables. Let $a \in \underline{A}$ be an assignment to variables A, and let $u \in \underline{U}$ be an assignment to variables $U \subseteq A \subseteq V$. We may write $a \models u$ to mean that a projected to U gives u, which can also be written as $a(U) = u$. We then also say that *a extends u*.

2.2.1 cp-theories

For set of variables V, the language $\mathcal{L}_V$ (abbreviated to $\mathcal{L}$) consists of all statements of the form $u : x > x' \,[W]$, where u is an assignment to a set of variables $U \subseteq V$ (i.e., $u \in \underline{U}$), $x, x' \in \underline{X}$ are different assignments to some variable $X \notin U$ (and so x and x' correspond to different values of X) and W is some subset

[1] For variable $X \in V$, the elements of $\underline{X}$, which can be written in the form $X = a$ for some value a of X, are in one-to-one correspondence with $\mathit{Domain}(X)$. We will usually slightly abuse notation, and refer to an element of $\underline{X}$ as a *value of* X, and refer to $\underline{X}$ as the *domain* of X.

of $V - U - \{X\}$. If φ is the statement $u : x > x' [W]$, we may write $u_\varphi = u$, $U_\varphi = U$, $x_\varphi = x$, $x'_\varphi = x'$, $W_\varphi = W$ and $T_\varphi = V - (\{X\} \cup U \cup W)$.

Subsets of $\mathcal{L}$ are called *conditional preference theories* or *cp-theories* (on V). For $\varphi = u : x > x' [W]$, let φ^* be the set of pairs of outcomes $\{(tuxw, tux'w') : t \in \underline{T_\varphi}, w, w' \in \underline{W}\}$. Such pairs $(\alpha, \beta) \in \varphi^*$ are intended to represent a preference for α over β, and φ is intended as a compact representation of the preference information φ^*. Informally, φ represents that, given u and any t, x is preferred to x', irrespective of the assignments to W. For conditional preference theory $\Gamma \subseteq \mathcal{L}$, define $\Gamma^* = \bigcup_{\varphi \in \Gamma} \varphi^*$, so Γ^* represents a set of preferences. We assume here that preferences are transitive, so it is then natural to define order $>_\Gamma$, induced on $\underline{V}$ by Γ, to be the transitive closure of Γ^*. In Section 3 it is shown that CP-nets can be represented in terms of statements $u : x > x' [W]$ with $W = \emptyset$, and TCP-nets with statements with W containing at most one variable.

Conditional preference theories allow locally partially ordered preferences: we do not need to assume that we can elicit a total order on the values of a variable given each assignment to its parents. This kind of representation of conditional preferences is very flexible as regards elicitation: we can reason with an arbitrary subset Γ of the language $\mathcal{L}$, so we can accept any conditional preference statements (of the appropriate form) that the agent is happy to give us. More statements can be added later, and, because the logic is monotonic (i.e., $\Gamma \subseteq \Delta$ implies $>_\Gamma \subseteq >_\Delta$), all of our previous deductions from Γ will still hold, in particular whether one outcome is preferred to another.

2.2.2 Example A

I'm planning a holiday. I can either go next week ($\mathbf{n}$) or later in the year ($\overline{\mathbf{n}}$). I've decided to go either to Oxford ($\mathbf{o}$) or to Manchester ($\overline{\mathbf{o}}$), and I can either take the plane ($\mathbf{p}$) or drive and take a car ferry ($\overline{\mathbf{p}}$). So, there are three variables, X_1, X_2 and X_3, where $\underline{X_1} = \{\mathbf{n}, \overline{\mathbf{n}}\}$, $\underline{X_2} = \{\mathbf{o}, \overline{\mathbf{o}}\}$ and $\underline{X_3} = \{\mathbf{p}, \overline{\mathbf{p}}\}$. Firstly, I'd prefer to go next week irrespective of the choices of the other variables, as I could do with a break soon. This can be represented by statement φ_1 which equals $\top : \mathbf{n} > \overline{\mathbf{n}}\ [\{X_2, X_3\}]$. This represents a set φ_1^* of pairs of outcomes $(\mathbf{n}\, w_1, \overline{\mathbf{n}}\, w_2)$, where w_1 and w_2 are both arbitrary assignments to the set of variables $\{X_2, X_3\}$; e.g., setting $w_1 = \overline{\mathbf{o}}\,\mathbf{p}$ and $w_2 = \mathbf{o}\,\mathbf{p}$ gives the pair $(\mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}, \overline{\mathbf{n}}\,\mathbf{o}\,\mathbf{p})$ indicating the preference of $\mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}$ over $\overline{\mathbf{n}}\,\mathbf{o}\,\mathbf{p}$. φ_1 is a compact way of representing the 16 pairs in φ_1^*. Secondly, all else being equal (*ceteris paribus*), I'd prefer to go to Oxford rather than Manchester. We represent this by the statement φ_2, which equals $\top : \mathbf{o} > \overline{\mathbf{o}}\ [\emptyset]$. This is an unconditional *ceteris paribus* statement. It represents set φ_2^* of pairs of outcomes $(x_1\mathbf{o}\,x_3, x_1\overline{\mathbf{o}}\,x_3)$ meaning outcome $x_1\mathbf{o}\,x_3$ is preferred to $x_1\overline{\mathbf{o}}\,x_3$, where x_1 is either value of X_1 and x_3 is either value of X_3.

My preferences on variable X_3 are conditional. I'd prefer to fly rather than drive unless I go later in the year to Manchester, when the weather will be warmer and a car would be useful for touring around. This can be represented by conditional preference statements φ_3, φ_4 and φ_5 defined as follows. φ_3 is $\mathbf{n} : \mathbf{p} > \overline{\mathbf{p}}\ [\emptyset]$, and φ_4 is $\mathbf{o} : \mathbf{p} > \overline{\mathbf{p}}\ [\emptyset]$. φ_5 is $\overline{\mathbf{n}}\,\overline{\mathbf{o}} : \overline{\mathbf{p}} > \mathbf{p}\ [\emptyset]$, representing φ_5^* which consists of the single preference of $\overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}$ over $\overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\mathbf{p}$, i.e., $\varphi_5^* = \{(\overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}, \overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\mathbf{p})\}$.

Let $\Gamma = \{\varphi_1, \ldots, \varphi_5\}$. The statement φ_1 cannot be represented in a CP-net on $V = \{X_1, X_2, X_3\}$. The others all can as they involve empty W. The induced partial ordering $>_\Gamma$ on outcomes can be shown to be the transitive closure of: $\mathbf{n\,o\,p} >_\Gamma \{\mathbf{n\,o\,\overline{p}}, \mathbf{n\,\overline{o}\,p}\} >_\Gamma \mathbf{n\,\overline{o}\,\overline{p}} >_\Gamma \mathbf{\overline{n}\,o\,p} >_\Gamma \mathbf{\overline{n}\,o\,\overline{p}} >_\Gamma \mathbf{\overline{n}\,\overline{o}\,\overline{p}} >_\Gamma \mathbf{\overline{n}\,\overline{o}\,p}$, so that $>_\Gamma$ is almost a total order, with only the pair of outcomes $\mathbf{n\,o\,\overline{p}}$ and $\mathbf{n\,\overline{o}\,p}$ not being ordered (see Figure 1).

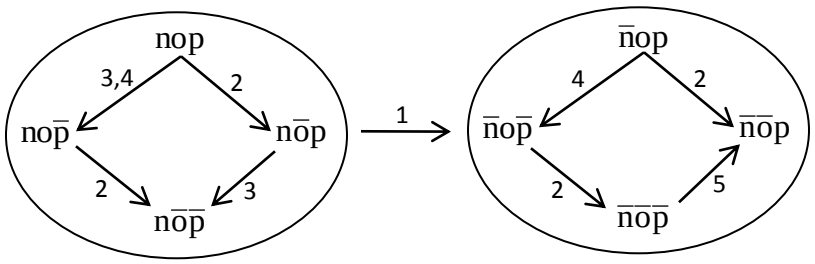

Figure 1: The ordering of outcomes in Example A. For $i = 2, \ldots, 5$, an edge from outcome α to outcome β is labelled with i if and only if preference statement φ_i entails that α is preferred to β i.e., iff $(\alpha, \beta) \in \varphi_i^*$. All four outcomes in the left-hand oval dominate, using φ_1, all the other four outcomes. All the edges together form Γ^*, and $>_\Gamma$ is the transitive closure of this set Γ^* of orderings.

2.2.3 Graphs $H(\Gamma)$ and $G(\Gamma)$ on V, and fully acyclic cp-theories

Directed graphs notation. In the paper we will define several kinds of directed graph on the set of variables, and we introduce the following notation. For $S, T \subseteq V$, define $S \to T$ to be the set of edges $\{(X, Y) : X \in S, Y \in T\}$. In addition, if S or T is a singleton set, then we may omit the set brackets, abbreviating, e.g., $S \to \{X\}$ to $S \to X$.

Let Γ be a cp-theory over set of variables V. We define a pair of binary relations on V, which are considered as directed graphs.

The *dependency graph* $H(\Gamma)$ consists of edges $U_\varphi \to X_\varphi$ for all φ in Γ. In other words, $H(\Gamma)$ consists of all pairs of the form (Y, X_φ), where $\varphi \in \Gamma$ and $Y \in U_\varphi$. $H(\Gamma)$ contains edge (Y, X) if and only if there is some conditional preference statement $\varphi \in \Gamma$ that makes the preferences for X conditional on Y. $H(\Gamma)$ thus encodes dependency information. Given fixed Γ, and variable $X \in V$, we will sometimes write U_X to mean the parents of variable X with respect to $H(\Gamma)$, i.e., the set of variables Y with $(Y, X) \in H(\Gamma)$. U_X is the set of variables that the preference for X can depend on.

Define $G(\Gamma)$ to contain $U_\varphi \to X_\varphi$ and $X_\varphi \to W_\varphi$ for all $\varphi \in \Gamma$, i.e., $G(\Gamma) = \bigcup_{\varphi\in\Gamma}(U_\varphi \to X_\varphi) \cup \bigcup_{\varphi\in\Gamma}(X_\varphi \to W_\varphi)$. $G(\Gamma)$ is $H(\Gamma)$ with extra edges (X, Z) when there is some preference statement $\varphi \in \Gamma$ representing a preference

for values of X irrespective of the value of Z, which implies that X is more important than Z in that context. $G(\Gamma)$ thus represents both the dependency and the relative importance information.

We say that cp-theory Γ is *fully acyclic* if $G(\Gamma)$ is acyclic. Fully acyclic cp-theories will be studied further in Section 5.

In the example it happens to be the case that both $H(\Gamma)$ and $G(\Gamma)$ are acyclic, and so Γ is fully acyclic: $H(\Gamma) = \{(X_1, X_3), (X_2, X_3)\}$ and $G(\Gamma)$ equals the total order on variables, $\{(X_1, X_2), (X_1, X_3), (X_2, X_3)\}$. However, more generally, this need not be so. Suppose that some cp-theory Γ contains the following statements: "If I go later in the year to Oxford, then I'd prefer to drive than fly", and "If I fly next week then I'd prefer to go to Manchester than Oxford"; $H(\Gamma)$ then contains both (X_2, X_3) and (X_3, X_2). There is nothing unreasonable or inconsistent about this (in particular, because the preferences are in disjoint contexts, the first regarding later in the year, the second regarding travelling next week).

2.3 Semantic entailment for cp-theories

In this section, we define a semantics and the notion of consistency for cp-theories. We define models for $\mathcal{L}$ to be strict total orders on $\underline{V}$, i.e., irreflexive transitive binary relations $>$ on $\underline{V}$ such that for all α and β in $\underline{V}$, with $\alpha \neq \beta$, either $\alpha > \beta$ or $\beta > \alpha$. For strict total order $>$, and conditional preference statement $\varphi \in \mathcal{L}$, we say that $>$ *satisfies* φ ($> \models \varphi$) if $> \supseteq \varphi^*$. Therefore, if φ is the statement $u : x > x' [W]$ then $>$ satisfies φ if and only if for all $t \in \underline{T}$ and $w, w' \in \underline{W}$, $tuxw > tux'w'$.

For cp-theory $\Gamma \subseteq \mathcal{L}$ we say that $>$ *satisfies* Γ ($> \models \Gamma$) if $>$ satisfies every element of Γ, which is if and only if $> \supseteq \Gamma^*$. We also then say that $>$ *is a model of* Γ.

Definition 1 (semantic entailment) *For $\Gamma \subseteq \mathcal{L}$ and $\varphi \in \mathcal{L}$, we define the semantic entailment relation by $\Gamma \models \varphi$ if and only if $> \models \varphi$ for all $>$ such that $> \models \Gamma$. For $\alpha, \beta \in \underline{V}$ we also say that $\Gamma \models (\alpha, \beta)$ if $\alpha > \beta$ holds for all models $>$ of Γ.*

Definition 2 (consistency of a cp-theory) *We say that cp-theory Γ* is consistent *if it has a model, i.e., if there exists a strict total order $>$ with $> \models \Gamma$.*

For Example A, consider the model $>$ defined as the transitive closure of $\mathbf{n}\,\mathbf{o}\,\mathbf{p} > \mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}} > \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p} > \mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}} > \overline{\mathbf{n}}\,\mathbf{o}\,\mathbf{p} > \overline{\mathbf{n}}\,\mathbf{o}\,\overline{\mathbf{p}} > \overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}} > \overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\mathbf{p}$. Total order $>$ satisfies $\varphi_1 = \top : \mathbf{n} > \overline{\mathbf{n}}\ [\{X_2, X_3\}]$ because any outcome extending $\mathbf{n}$ is preferred to any outcome extending $\overline{\mathbf{n}}$, i.e., $\alpha > \beta$ for any outcomes α and β with $\alpha(X_1) = \mathbf{n}$ and $\beta(X_1) = \overline{\mathbf{n}}$. It can be checked that $>$ satisfies each statement in Γ, and so satisfies Γ, i.e., $>$ extends Γ^*. This shows that Γ is consistent. Γ has one other model, which differs from $>$ only in how it orders outcomes $\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}}$ and $\mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}$. Hence we have, for example, $\Gamma \models (\mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}, \overline{\mathbf{n}}\,\mathbf{o}\,\mathbf{p})$, but

it is not the case that $\Gamma \models (\mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}, \mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}})$, since there exists a model $>$ of Γ with $\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}} > \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}$, and nor do we have $\Gamma \models (\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}}, \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p})$.

The construction of semantic entailment relation $\models$ ensures that it is monotonic; in particular, if $\Gamma \subseteq \Delta \subseteq \mathcal{L}$ and $\Gamma \models (\alpha, \beta)$ then $\Delta \models (\alpha, \beta)$. The following lemma sums up some basic properties of semantic entailment.

Lemma 2 *Let $\Gamma \subseteq \mathcal{L}$ be a cp-theory over variables V.*

(i) Let $>$ be a strict total order on $\underline{V}$. Then $>$ satisfies Γ if and only if $>$ extends $>_\Gamma$, i.e., $> \models \Gamma \iff > \supseteq >_\Gamma$.

(ii) The following four statements are equivalent: (a) Γ is consistent; (b) Γ^* is acyclic; (c) $>_\Gamma$ is irreflexive; (d) $>_\Gamma$ is a strict partial order.

(iii) If Γ is consistent then $>_\Gamma$ is equal to the intersection of all strict total orders satisfying Γ, i.e., $>_\Gamma$ equals $\bigcap_{> \models \Gamma} >$.

(iv) If Γ is consistent then $\Gamma \models (\alpha, \beta)$ if and only if $\alpha >_\Gamma \beta$.

(v) If Γ is consistent then $\Gamma \models \varphi$ if and only if $>_\Gamma \supseteq \varphi^*$.

2.4 Proof theory

We describe a proof theory for cp-theories, based on *swapping sequences* (which generalise flipping sequences for CP-nets (Boutilier et al., 2004a)), and give a completeness result, relating the proof theory with the semantics.

Definition 3 (swapping sequences) *Let $\alpha, \beta \in \underline{V}$ be two outcomes. We say that* there is a worsening swap from α to β *for cp-theory Γ if $(\alpha, \beta) \in \Gamma^*$, i.e., iff there exists $\varphi = (u : x > x' [W]) \in \Gamma$ such that $\alpha \models u$, $\beta \models u$, $\alpha(X) = x$, $\beta(X) = x'$, and $\alpha(T_\varphi) = \beta(T_\varphi)$. We say that* there is a worsening swapping sequence from α to β *(for Γ) if there exists a sequence $\alpha = \alpha_1, \ldots, \alpha_l = \beta$ such that for each $k = 1, \ldots, l-1$, there is a worsening swap from α_k to α_{k+1}, i.e., $(\alpha_k, \alpha_{k+1}) \in \Gamma^*$.*

For instance, in Example A, there is a worsening swap (and hence a worsening swapping sequence) from $\mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}$ to $\overline{\mathbf{n}}\,\mathbf{o}\,\mathbf{p}$, since $(\mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}, \overline{\mathbf{n}}\,\mathbf{o}\,\mathbf{p}) \in \varphi_1^*$. There is also a worsening swapping sequence from $\overline{\mathbf{n}}\,\mathbf{o}\,\overline{\mathbf{p}}$ to $\overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\mathbf{p}$, since there is a worsening swap from $\overline{\mathbf{n}}\,\mathbf{o}\,\overline{\mathbf{p}}$ to $\overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}$ (using φ_2), and a worsening swap from $\overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}$ to $\overline{\mathbf{n}}\,\overline{\mathbf{o}}\,\mathbf{p}$ (using φ_5).

Clearly, if there is a worsening swapping sequence from α to β then (α, β) is in the transitive closure $>_\Gamma$ of Γ^*. Conversely, if (α', β') is in the transitive closure of Γ^* then there exists a sequence $\alpha' = \alpha_1, \ldots, \alpha_l = \beta'$ with for each $k = 1, \ldots, l-1$, $(\alpha_k, \alpha_{k+1}) \in \Gamma^*$. We therefore have, using Lemma 2(ii) and (iv), the following result which is a soundness and completeness result for worsening swapping sequences.

Theorem 1 (Soundness and completeness for swapping sequences) [2]
Let Γ be a conditional preference theory on V and let $\alpha, \beta \in \underline{V}$ be outcomes. Then

(i) $\alpha >_\Gamma \beta$ if and only if there exists a worsening swapping sequence for Γ from α to β;

(ii) Γ is consistent if and only if the associated preference relation $>_\Gamma$ is irreflexive (which is if and only if Γ^* is acyclic);

(iii) if Γ is consistent then $\Gamma \models (\alpha, \beta) \iff \alpha >_\Gamma \beta$, which is if and only if there exists a worsening swapping sequence for Γ from α to β.

3 Expressiveness, CP-nets, TCP-nets and lexicographic orders

This section considers issues regarding the expressiveness of cp-theories, CP-nets and TCP-nets. It is shown in Section 3.1 how to map a CP-net to a cp-theory with the same associated preference relation; Section 3.2 does the same for TCP-nets. Lexicographic orders are considered in Section 3.3, where it is shown how they can be represented easily using a cp-theory, but not by CP-nets or TCP-nets. Lexicographic orders can be viewed as being composed of a set of a particular kind of strong preference statement where the choice of values of a variable dominate the assignments to a set of other (less important) variables. Some results are presented in Section 3.4 which show that, these statements are not at all natural for CP-nets or TCP-nets. In Section 3.5 we show how hard it is to generate a total order on outcomes with purely *ceteris paribus* preferences; we show that for any number n there is essentially a unique acyclic CP-net on n Boolean variables whose associated preference relation is a total order.

3.1 Expressing CP-nets in the language

In this section we show how CP-nets can be expressed as conditional preference theories, using statements $u : x > x' [W]$ with $W = \emptyset$. It is shown in Section 3.3 and Section 3.4 that the language is a good deal more expressive than CP-nets.

A *CP-net over V* is defined (see (Boutilier et al., 1999) and especially Definitions 1, 2 and 3 of (Boutilier et al., 2004a)) to be a pair $N = (H, \mathrm{CT})$ where H is a (binary) relation on V (which is conventionally thought of as a directed graph) and CT is a function which assigns a *conditional preference table* to each

[2] An alternative to using strict total orders as models is to use total pre-orders (reflexive, transitive and complete binary relations), see Section 3 of (Brafman & Dimopoulos, 2004). We can define total pre-order $\succcurlyeq$ to satisfy Γ if $\succcurlyeq \supseteq \Gamma^*$. Every Γ then has a model, in particular every Γ is satisfied by the total pre-order with $\alpha \succcurlyeq \beta$ for all outcomes α and β. We can define relation $\models'$ as follows: $\Gamma \models' (\alpha, \beta)$ if and only if $\alpha \succcurlyeq \beta$ holds for all total pre-orders $\succcurlyeq$ satisfying Γ. This leads to a fuller completeness result: $\Gamma \models' (\alpha, \beta)$ if and only if $\alpha >_\Gamma \beta$: see Theorem 1 of (Wilson, 2006), and also Theorem 1 of (Brafman & Dimopoulos, 2004).

variable $X \in V$. The conditional preference table $\mathrm{CT}(X)$ is defined to be a function which assigns to each[3] $u \in \mathrm{Pa}_H(X)$ a strict total order[4] $\succ_u^X$ on $\underline{X}$.

Let $>$ be a strict total order on $\underline{V}$. Let X be a variable and let $u \in \mathrm{Pa}_H(X)$ be an assignment to the parents of X. Let $T = V - \mathrm{Pa}_H(X) - \{X\}$. $>$ is said to satisfy $\succ_u^X$ if $tux > tux'$ holds for all $t \in \underline{T}$ and for all $x, x' \in \underline{X}$ such that $x \succ_u^X x'$. We say that $>$ *satisfies* CP-net $N = (H, \mathrm{CT})$ if for all $X \in V$, and all $u \in \mathrm{Pa}_H(X)$, $>$ satisfies $\succ_u^X$ (where $\succ_u^X = \mathrm{CT}(X)(u)$). CP-net N is said to be satisfiable if there exists some $>$ which satisfies N. There is a simple sufficient condition for satisfiability of a CP-net N: that its associated relation H is acyclic. We say that N is *acyclic* if its associated relation H is acyclic.

For CP-net N define relation $\succ_N$ on $\underline{V}$ as follows. For $\alpha, \beta \in \underline{V}$, $\alpha \succ_N \beta$ if and only if $\alpha > \beta$ for all total orders $>$ satisfying N. Therefore, $\succ_N$ is the intersection of all $>$ satisfying N.

Representing a CP-net N as a cp-theory

For variable $X \in V$ and assignment $u \in \mathrm{Pa}_H(X)$ to the parent variables, let $\Gamma_N^{X,u} \subseteq \mathcal{L}$ be the set of statements $\{(u : x > x' [\emptyset]) : x, x' \in \underline{X}, x \succ_u^X x'\}$. Let conditional preference theory Γ_N be the union of sets $\Gamma_N^{X,u}$ over all $X \in V$ and $u \in \mathrm{Pa}_H(X)$. Note that the construction of Γ_N is linear in the size of the conditional preference table. (If the domain of variable X is large, one might represent total order $\succ_u^X$ by a sub-relation whose transitive closure is $\succ_u^X$; the sub-relation could then also be used in the definition of $\Gamma_N^{X,u}$.) Now, for strict total order $>$, we have $> \models \Gamma_N^{X,u}$ if and only if $>$ satisfies $\succ_u^X$. So, $> \models \Gamma_N$ if and only if $>$ satisfies N. Therefore, Γ_N is consistent if and only if N is satisfiable. We have:

Proposition 1 *Let N be a CP-net, and $\Gamma_N \subseteq \mathcal{L}$ (as defined above) be its associated conditional preference theory. Then N is satisfiable if and only if Γ_N is consistent. If N is satisfiable, then $>_{\Gamma_N} = \succ_N$.*

Proof: The first part has already been shown. Now, suppose that N is satisfiable, and so Γ_N is satisfiable. $\succ_N$ is the intersection of all $>$ satisfying N, that is, the intersection of all $>$ satisfying Γ_N, which, by Lemma 2(iii), equals $>_{\Gamma_N}$. □

This shows that a CP-net can be represented within the language $\mathcal{L}$, with the same associated preference order on outcomes.

3.2 Expressing TCP-nets within the language

In this section we show how TCP-nets—a generalisation of CP-nets—can be represented using cp-theories. A TCP-net (Brafman et al., 2006; Brafman & Domshlak, 2002) on set of variables V can be considered as consisting of a

[3] $\mathrm{Pa}_H(X)$, the parents of X with respect to H, is the set of all Y such that $(Y, X) \in H$.

[4] If we relax this assumption by allowing $\succ_u^X$ to be a non-empty strict partial order then the results all still hold.

directed graph H on V, a conditional preference table, a set of *i-arcs*, and a set of *ci-statements*. For $X \in V$, let U_X be $\mathrm{Pa}_H(X)$, the set of parents of X in H, i.e., the set of variables Y such that $(Y, X) \in H$. A conditional preference table assigns to each $X \in V$ and assignment $u \in \underline{U_X}$ a strict partial order $\succ_u^X$ on $\underline{X}$, i.e., it partially orders the values of X. An i-arc is an ordered pair of different variables X and Y, which we write as $X \to Y$. It is intended to represent that X is a much more important variable than Y. A ci-statement consists of an ordered pair of variables X and Y and an assignment s to some set of variables $S_{X,Y} \subseteq V - \{X, Y\}$; such a statement is written here as $X \to_s Y$. It is intended to represent that given s, X is much more important than Y. (It is assumed that $Y \notin U_X$ and $X \notin U_Y$.)

A strict partial order $>$ on outcomes is said to satisfy the conditional preference table if for each $X \in V$ and $u \in \underline{U_X}$ it satisfies the associated ordering $\succ_u^X$; strict partial order $>$ satisfies $\succ_u^X$ if $[x \succ_u^X x'$ implies for all $t \in \underline{T}$ $tux > tux']$, where $T = V - \{X\} - U_X$.

Given a TCP-net N, strict partial order $>$ is said to satisfy an i-arc $X \to Y$ if $rxy > rx'y'$ holds for all x, x' such that $x \succ^X_{r(U_X)} x'$, and for all $y, y' \in \underline{Y}$, and for all assignments r to $V - \{X, Y\}$.

Given a TCP-net N, strict partial order $>$ satisfies ci-statement $X \to_s Y$ if $rsxy > rsx'y'$ holds for all assignments r to $V - S_{X,Y} - \{X, Y\}$, all x, x' such that $x \succ_u^X x'$ and all $y, y' \in \underline{Y}$ where u is rs restricted to U_X.

Strict partial order $>$ on outcomes is said to satisfy a TCP-net if it satisfies the conditional preference table, every i-arc and every ci-statement.

Define the TCP-net order on outcomes as follows: for TCP-net N, define $>_N$ on $\underline{V}$ by: for $\alpha, \beta \in \underline{V}$, $\alpha >_N \beta$ if and only if $\alpha > \beta$ for all strict partial orders $>$ satisfying N.

Therefore, $>_N$ is the intersection of all strict partial orders satisfying N. Lemma 1(v) then implies that $>_N$ is the intersection of elements of J, where J is the set of strict total orders that extend some partial order satisfying N. The definitions immediately imply that if strict partial order $>$ satisfies a TCP-net then any strict total order extending $>$ also satisfies the TCP-net. Hence, J equals the set of strict total orders satisfying N, and so, $>_N$ is equal to the intersection of all strict total orders satisfying N. Therefore, $\alpha >_N \beta$ if and only if $\alpha > \beta$ for all strict total orders $>$ satisfying N.

Representing TCP-nets orderings using cp-theories.

Let N be a TCP-net as defined above. We will define a cp-theory Γ_N that generates the same order on outcomes.

Define $\Gamma_{\mathrm{cp}} \subseteq \mathcal{L}$ to be the set of statements $u : x > x'[\emptyset]$ over all $X \in V$, $u \in \underline{U_X}$, and $x, x' \in \underline{V}$ such that $x \succ_u^X x'$ (where $\succ_u^X$ is part of the conditional preference table of N).

For i-arc $X \to Y$ of N define $\Gamma_{X \to Y} \subseteq \mathcal{L}$ to be the set of statements $u : x > x'\,[Y]$ such that $u \in \underline{U_X}$ and x and x' are such that $x \succ_u^X x'$. Let Γ_{i} be the union of the $\Gamma_{X \to Y}$ over all i-arcs $X \to Y$ of N.

For ci-statement $X \rightarrow_s Y$ define $\Gamma_{X \rightarrow_s Y}$ to be the set of statements $qs : x > x' [Y]$ for all assignments q to $U_X - S_{X,Y}$ and all x, x' such that $x \succ_u^X x'$, where u is qs restricted to U_X. Let $\Gamma_{\rm ci}$ be the union of $\Gamma_{X \rightarrow_s Y}$ over all ci-statements $X \rightarrow_s Y$ of N.

Finally, define the cp-theory Γ_N to be $\Gamma_{\rm cp} \cup \Gamma_{\rm i} \cup \Gamma_{\rm ci}$. These definitions easily lead to the following result.

Proposition 2 *TCP-net N is satisfiable if and only if Γ_N is consistent. If N is satisfiable, then $>_N = >_{\Gamma_N}$.*

This means that cp-theories are more general than TCP-nets, in the sense that any TCP-net can be efficiently converted into a cp-theory which has the same preference relation on outcomes. The TCP-net order $>_N$ only differs from the corresponding cp-theory order $>_{\Gamma_N}$ when N is not satisfiable; but in that case, the TCP-net order becomes trivial: $>_N$ is the complete relation $\underline{V} \times \underline{V}$.

As shown above, TCP-nets represent conditional preference statements φ with $|W_\varphi| = 0$ or 1; they cannot directly represent statements with larger W_φ (and in many situations, one variable will be more important than each of a large set of variables, so W_φ can be large). It is not immediately obvious how much difference this makes: how much is lost by approximating a statement $\varphi = (u : x > x' [W])$ by a set Δ of statements $u : x > x' [\{Y\}]$ over all variables Y in W? One can get a good idea of the answer to this by comparing the sizes of φ^* and Δ^*, which represent the direct consequences of the conditional preference statements. For example, with all binary (two-valued) variables, it can be shown that $|\Delta^*|/|\varphi^*| = (k+1)2^{-k}$, where $k = |W|$, so the TCP-style approximation to a statement $u : x > x' [W]$ will tend to be a very poor one unless W is small.

Example B: This is a variation of the holiday example (Example A) in Section 2.2. To make the relationship between the values and variables clearer we use x_1 instead of **n** for travelling next week, and x_2 instead of **o** for Oxford, and x_3 instead of **p** for travelling by plane. As well as the decision regarding when, where and how I travel, I also have to decide whether to take my expensive camera x_4, or my cheaper one $\overline{x}_4$. This last choice is much less important than the others. So, there are four variables, X_1, X_2, X_3 and X_4, where $\underline{X_1} = \{x_1, \overline{x}_1\}$, $\underline{X_2} = \{x_2, \overline{x}_2\}$, $\underline{X_3} = \{x_3, \overline{x}_3\}$ and $\underline{X_4} = \{x_4, \overline{x}_4\}$.

Firstly, I'd prefer to go next week irrespective of the choices of the other variables. This is represented by the following preference statement: $\top : x_1 > \overline{x}_1 [\{X_2, X_3, X_4\}]$. This implies that outcome α is preferred to β whenever $\alpha(X_1) = x_1$ and $\beta(X_1) = \overline{x}_1$, irrespective of what the other values of α and β are. It represents a strong kind of preference, but one that is natural in many contexts. As we shall see, this cannot be represented in a CP-net or TCP-net.

If I go next week I definitely want to fly, as I can't face the long drive, which is represented by the preference statement $x_1 : x_3 > \overline{x}_3 [\{X_2, X_4\}]$. Also, I'd prefer to go to Oxford in that case: $x_1 : x_2 > \overline{x}_2 [\{X_4\}]$. So, if I go next week, the choice of how I travel (X_3) is more important than the choice of where I go

(X_2). Later in the year my preference for Oxford is irrespective of how I travel: $\overline{x}_1 : x_2 > \overline{x}_2\,[\{X_3, X_4\}]$. If I go later, I'd also prefer to drive than to fly (whether I go to Oxford or Manchester), $\overline{x}_1 : \overline{x}_3 > x_3\,[\{X_4\}]$, as it would then be useful having a car with me. If I fly I'd prefer to take my cheap camera, whereas if I drive I'd rather take the better one: $x_3 : \overline{x}_4 > x_4\,[\emptyset]$ and $\overline{x}_3 : x_4 > \overline{x}_4\,[\emptyset]$.

Let Γ be this set of preference statements. Unlike in Example A, $G(\Gamma)$ is not acyclic, as it contains pairs (X_2, X_3) and (X_3, X_2). Let $\alpha = x_1\overline{x}_2\overline{x}_3\overline{x}_4$ and $\beta = \overline{x}_1 x_2 \overline{x}_3 x_4$, and let φ be the first preference statement, $\top : x_1 > \overline{x}_1\,[\{X_2, X_3, X_4\}]$. Then $(\alpha, \beta) \in \varphi^*$ since $\alpha(X_1) = x_1$ and $\beta(X_1) = \overline{x}_1$. So, there is a worsening swap from α to β, and therefore $\alpha >_\Gamma \beta$. It can be seen that α and β are consecutive in the order $>_\Gamma$, with no outcome γ such that $\alpha >_\Gamma \gamma >_\Gamma \beta$.

Since α and β differ on three variables, it can be seen that there exists no TCP-net N on V with $>_N$ equalling $>_\Gamma$ (see Lemma 3 below). Because $>_\Gamma$ happens in this case to be a total order this further implies that there exists no satisfiable TCP-net N which satisfies these preferences, i.e, with $>_N \supseteq \Gamma^*$. This illustrates the fact that a statement such as φ is strictly stronger than a CP-net statement $\top : x_1 > \overline{x}_1$ along with three i-arcs $X_1 \to X_2$, $X_1 \to X_3$ and $X_1 \to X_4$; with the latter, the outcomes $x_1\overline{x}_2\overline{x}_3\overline{x}_4$ and $\overline{x}_1 x_2 \overline{x}_3 x_4$ would not be ordered by the TCP-net.

3.3 Representing lexicographic orders

We will show how to represent a lexicographic order with a cp-theory. For set of variables V, a lexicographic order on $\underline{V}$ involves an ordering $X_1, \ldots, X_n$ of the variables V, and for each X_i a total order $>_i$ on the set of values $\underline{X_i}$ of X_i. Define relation $>_{lex}$ as follows. For $\alpha, \beta \in \underline{V}$, $\alpha >_{lex} \beta$ if and only if $\alpha \neq \beta$ and $\alpha(X_i) >_i \beta(X_i)$, where X_i is the first variable (i.e., with minimal i) such that $\alpha(X_i) \neq \beta(X_i)$. The lexicographic order $>_{lex}$ is easily seen to be a strict total order on $\underline{V}$.

The following proposition shows that lexicographic orders can be represented by conditional preference theories, i.e., for any lexicographic order $>_{lex}$, there exists cp-theory Γ such that its associated order $>_\Gamma$ equals $>_{lex}$.

Proposition 3 *For each variable X_i, let Γ_i be the set of all statements of the form $\top : x > x'\,[\{X_{i+1}, \ldots, X_n\}]$, where $x, x' \in \underline{X_i}$ are such that $x >_i x'$. Let $\Gamma = \Gamma_1 \cup \cdots \cup \Gamma_n$. Then the associated order $>_\Gamma$ equals $>_{lex}$.*

The following lemma is useful for revealing limits to the expressivity of CP-nets and TCP-nets. We say that α *covers* β with respect to a transitive relation $\succ$ on $\underline{V}$ if $\alpha \succ \beta$ and there does not exist $\gamma \in \underline{V}$ with $\alpha \succ \gamma \succ \beta$.

Lemma 3

(i) Let Γ be a conditional preference theory. Suppose α covers β with respect to $>_\Gamma$. Then there is a worsening swap from α to β.

(ii) Let N be a CP-net. Suppose α covers β with respect to $\succ_N$. Then α and β differ on precisely one variable. In other words, there exists $X \in V$ with $\alpha(X) \neq \beta(X)$ and for all $X' \in V - \{X\}$, $\alpha(X') = \beta(X')$.

(iii) Let M be a TCP-net, with associated relation $\succ_M$ Suppose α covers β with respect to $\succ_M$. Then α and β differ either on one variable or on two variables.

All three parts follow easily from the appropriate completeness theorems for swapping/flipping sequences: Theorem 1, Section 2.4, for (i); Theorem 8 (the CP-nets completeness result for flipping sequences) of (Boutilier et al., 2004a) for (ii); and for (iii), the TCP-nets completeness result: see Theorem 6 of (Brafman et al., 2006).

In Example A (Section 2.2.2), there are a pair of outcomes, $\mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}$ and $\overline{\mathbf{n}}\,\mathbf{o}\,\mathbf{p}$, which are consecutive in the preference order $>_\Gamma$ that differ on all three variables. Lemma 3 then implies that the preferences in this example cannot be represented by a CP-net or TCP-net, i.e., there's no CP-net or TCP-net N on V with $>_N$ equal to $>_\Gamma$.

A consequence of Lemma 3 is that, except in some trivial cases, if N is a CP-net or a TCP-net, then $\succ_N$ is never a lexicographic order. This is because lexicographic orders on n variables include consecutive elements that differ on all n variables (assuming the domain of each variable has more than one element). To illustrate this, consider the case of Boolean variables and the order on complete tuples being just the usual order of binary numbers. Then $(1, 0, 0, \ldots, 0)$ and $(0, 1, 1, \ldots, 1)$ are consecutive in the order, but they differ on all the variables. Therefore, by the lemma, the order cannot be generated by a CP-net if $n > 1$ and the order cannot be generated by a TCP-net if $n > 2$. This leads to the following result (which also appears in Section 3.2 of (Freuder, Heffernan, Wallace, & Wilson, 2010)).

Proposition 4 *Let $>_{lex}$ be a lexicographic order (as defined above) on $\underline{V}$, where the domain of each variables contains more than one element, i.e., for all $X \in V$, $|\underline{X}| > 1$. Then (a) if $|V| > 1$, there exists no CP-net N on V with $\succ_N = >_{lex}$; (b) if $|V| > 2$, there exists no TCP-net M on V with $\succ_M = >_{lex}$.*

3.4 Representing stronger conditional preferences

In this section we show how a strong kind of preference statement, of the sort that holds for a lexicographic order, can be represented with a cp-theory.

Lexicographic orders are a very special type of order, but the kind of statements they represent can be natural. Let $\succ$ be a strict partial order (i.e., a transitive irreflexive relation) on $\underline{V}$. Let $X \in V$ and $W \subseteq V - \{X\}$ and let $T = V - \{X\} - W$, so that $\{X\}$, W and T partition V. Let $>_X$ be a non-empty partial order on $\underline{X}$, the set of assignments to variable X. We say that X (unconditionally) *dominates W with respect to* $(\succ, >_X)$ if the following condition holds: for $\alpha, \beta \in \underline{V}$, $\alpha \succ \beta$ holds whenever α and β are such that: $\alpha(X) >_X \beta(X)$ and

$\alpha(T) = \beta(T)$. In other words, α is preferred to β if α and β agree on T and α is better than β on X.

In particular, if X dominates $W = V - \{X\}$ with respect to $(\succ, >_X)$, then a sufficient condition for $\alpha \succ \beta$ is $\alpha(X) >_X \beta(X)$. This is a stronger form of preference statement than *ceteris paribus* statements. It represents a situation where the value of variable X is much more important than the values of any other variable; we prefer any outcome that does better on variable X.

This kind of condition is naturally represented within the language $\mathcal{L}$. Let Θ be the set of preferences statements $\{(\top : x > x' [W]) : x >_X x'\}$. Then, if cp-theory Γ contains Θ, X dominates W with respect to $(>_\Gamma, >_X)$. Such statements can be used to represent a lexicographic order, as shown above in Proposition 3. In contrast, this type of variable dominance is not at all natural for CP-nets and TCP-nets, as the following results indicate. It is, however, easy to construct a consistent cp-theory Γ that satisfies the hypotheses of the two propositions (e.g., $\Gamma = \Theta$ for the representation Θ above, or extensions of Θ, in particular, representing a lexicographic order).

Proposition 5 *Consider any consistent CP-net N on $V = \{X_1, \ldots, X_n\}$ ($n \geq 2$) such that X_2 has no parents and $|\underline{X_2}| > 1$. Then for no (non-empty) $>_1$ on $\underline{X_1}$ is it the case that X_1 dominates $\{X_2, \ldots, X_n\}$ with respect to $(\succ_N, >_1)$.*

In Example A, X_1 dominates $\{X_2, X_3\}$ with respect to $(>_\Gamma, >_1)$, where $\mathbf{n} >_1 \overline{\mathbf{n}}$; also X_2 has no parents. The proposition then implies (without looking at the level of outcomes) that there's no CP-net N on V with $\succ_N = >_\Gamma$. It also implies that the same would hold if we were to change the preferences on X_3 in any way.

There is a similar result for TCP-nets:

Proposition 6 *Consider any consistent TCP-net M on $V = \{X_1, \ldots, X_n\}$ ($n \geq 3$) with total local orderings and such that X_2 has no parents and X_3 has no parents, $|\underline{X_2}|, |\underline{X_3}| > 1$. Then for no total order $>_1$ on $\underline{X_1}$ is it the case that X_1 dominates $\{X_2, \ldots, X_n\}$ with respect to $(\succ_M, >_1)$.*

3.5 Generating precisely a total order on outcomes

We finish Section 3 with an expressibility result illustrating how unusual it is for a CP-net to generate a total order of outcomes. It shows that once one removes the obvious symmetries concerned with variable and value ordering, there is a unique acyclic CP-net on a set V of Boolean variables that generates a total order of outcomes.

This contrasts with the situation for conditional preference theories, where there are doubly exponential number[5]of total orders $>$ on $\underline{V}$ whose maximum

[5]It follows from Proposition 17 below in Section 5.2 that we can instead count the number of total orders equalling $\succ_{p(\Gamma)}$ for some cp-theory Γ satisfying these properties, since if $>_\Gamma$ is a total order then $\succ_{p(\Gamma)}$ equals $>_\Gamma$, and if $\succ_{p(\Gamma)}$ is a total order then $>_{\overline{\Gamma}}$ equals $\succ_{p(\Gamma)}$. For $\succ_{p(\Gamma)}$ to be a total order we just need that the transitive closure of $G(\Gamma)$ is a total order and

element is $(1, 1, \ldots, 1)$ and which are equal to some $>_\Gamma$, for cp-theory Γ such that $G(\Gamma)$ is consistent with the variable ordering $X_1, \ldots, X_n$.

Theorem 2 *For any given value of $n \geq 1$, there is a unique CP-net N on Boolean variables $V = \{X_1, \ldots, X_n\}$ satisfying the following properties:*

(i) the CP-net order $\succ_N$ is a strict total order of outcomes with maximum element $(1, \ldots, 1)$; and

(ii) the variable ordering $X_1, X_2, \ldots, X_n$ is consistent with the relation H on V associated with N, i.e., $(X_j, X_i) \in H$ implies $j < i$.

We will show, furthermore, that H is maximally large: $H = \{(X_j, X_i) : j < i\}$ so that the parents set $\mathrm{Pa}(X_i)$ of X_i is $\{X_1, \ldots, X_{i-1}\}$. The conditional preference tables (when written out explicitly) are therefore of exponential size. They can be expressed compactly as follows: for each $i = 1, \ldots, n$, and assignment u to $\mathrm{Pa}(X_i)$, $1 \succ_u^{X_i} 0$ holds if and only if u (viewed as a sequence of Boolean values) contains an even number of zeros.

We derive three auxiliary results to help prove this theorem. The first two prove that there is at most one ordering $>$ on outcomes equalling $\succ_N$ for some CP-net satisfying conditions (i) and (ii) in Theorem 2. The third result gives an explicit construction of such a CP-net.

First we consider the case of $n = 3$, to illustrate the ideas behind the results.

Example 1 *Suppose that we'd like to construct an acyclic CP-net N on Boolean variables X_1, X_2 and X_3 such that $\succ_N$ is a total order. We can relabel the values so that $(1, 1, 1)$ is the optimal outcome. Since N is acyclic, we can choose some variable, which we relabel to being X_3, which has no children. We can generate a CP-net N' on variables X_1 and X_2 by deleting from N the conditional preferences of X_3. It can be seen that $\succ_N$ being a total order implies that $\succ_{N'}$ is a total order (by deleting flips involving X_3 from flipping sequences for N). Without loss of generality, let us assume that X_1 has no parents. Then we can see that N' must be the CP-net with preferences $1 \succ_\top^{X_1} 0$, and $1 \succ_{X_1=1}^{X_2} 0$, and $0 \succ_{X_1=0}^{X_2} 1$.*

Now, if outcomes for N, α and β, agree on X_1 and X_2 and differ on X_3 then they must be consecutive in the total order. Otherwise, removing the flips changing X_3 in a flipping sequence between α and β gives a flipping sequence in N' from $\alpha(\{X_1, X_2\})$ to $\beta(\{X_1, X_2\}) = \alpha(\{X_1, X_2\})$, and hence a cycle, contradicting consistency of N'. Let $\alpha_1, \alpha_2, \ldots, \alpha_8$ be the ordering of outcomes, so $\alpha_1 = (1, 1, 1)$. The previous remark implies that $(1, 1, 1)$ and $(1, 1, 0)$ are consecutive, so $\alpha_2 = (1, 1, 0)$. Now, α_3 and α_2 are consecutive, so there exists a worsening flip from α_2 to α_3, so they differ on precisely one variable. This means that $\alpha_2(X_3) = \alpha_3(X_3) = 0$. If we change the value of X_3 in α_3 to 1 then, by the earlier remark, we obtain an outcome consecutive with α_3, which

all the local orderings $\succ_\alpha^X$ are total orderings. X_n can have parents $\{X_1, \ldots, X_{n-1}\}$, and so there are a doubly exponential number ($2^{2^{n-1}-1}$) of valid non-equivalent choices for the set of local orderings for X_n.

must be α_4. *Hence,* $\alpha_4(X_3) = 1$. *Continuing this argument, the values of* X_3 *in* $\alpha_1, \alpha_2, \ldots, \alpha_8$ *are 1, 0, 0, 1, 1, 0, 0, 1. Therefore, the ordering on outcomes is* $(1,1,1)$, $(1,1,0)$, $(1,0,0)$, $(1,0,1)$, $(0,0,1)$, $(0,0,0)$, $(0,1,0)$, $(0,1,1)$.

This kind of reasoning is generalised and formalised in Lemma 4 and its proof.

Lemma 4 *Let* N *be an acyclic CP-nets on Boolean variables* V *such that* $\succ_N$ *is a strict total order on* $\underline{V}$. *Let* Z *be a variable with no children, i.e., the conditional preferences of no variable are conditional on* Z. *List the elements* $\underline{V}$ *in decreasing order with respect to* $\succ_N$ *as* $\alpha_1, \alpha_2, \ldots, \alpha_K$, *where* $K = 2^{|V|}$. *Let* N' *be the CP-net on variables* $V - \{Z\}$ *formed by deleting the conditional preferences of* Z. *Then the following hold.*

(a) $\succ_{N'}$ *is a strict total order on* $\underline{V - \{Z\}}$.

(b) If $\alpha, \beta \in \underline{V}$ *differ on* $V - \{Z\}$ *then* $\alpha \succ_N \beta$ *if and only if* $\alpha(V - \{Z\}) \succ_{N'} \beta(V - \{Z\})$.

(c) If outcomes α *and* β *differ on* Z *but agree on all other variables, then* α *and* β *are consecutive in* $\succ_N$. *Hence, for all* $j = 1, 2, \ldots, \frac{K}{2}$, *outcomes* α_{2j-1} *and* α_{2j} *agree on* $V - \{Z\}$.

(d) Suppose that $\alpha_1(Z) = 1$. *Then the value of* Z *in the sequence* $\alpha_1, \alpha_2, \ldots, \alpha_K$ *follows the pattern: 1, 0, 0, 1, 1, 0, 0, 1,..., so that for* $l \in \{1, 2, 3, 4\}$, *and* $j = 0, 1, \ldots, \frac{K}{4} - 1$, *we have* $\alpha_{4j+l}(Z) = 1$ *if* $l = 1, 4$, *and* $\alpha_{4j+l}(Z) = 0$ *if* $l = 2, 3$.

Proposition 7 shows that there is at most one CP-net ordering satisfying the conditions of Theorem 2. Proposition 8 defines such a CP-net, showing that there exists exactly one. Theorem 2 can then be proved using these two results.

Proposition 7 *Let* n *be any natural number, and let* $V = \{X_1, \ldots, X_n\}$ *be a set of Boolean variables. There is at most one strict total order* $>$ *on* $\underline{V}$ *with* $>$ *equalling* $\succ_N$ *for some CP-net* N *on* V *satisfying the pair of conditions (i) the CP-net order* $\succ_N$ *is a strict total order of outcomes with maximum element* $(1, \ldots, 1)$*; (ii) the variable ordering* $X_1, X_2, \ldots, X_n$ *is consistent with the relation* H *on* V *associated with* N, *i.e.,* $(X_j, X_i) \in H$ *implies* $j < i$.

Sketch of proof: The proof is by induction on n. Suppose that there are CP-nets, N_1 and N_2, on variables $V = \{X_1, \ldots, X_n\}$ satisfying the conditions of the proposition. Eliminating variable X_n gives CP-nets N_1' and N_2' on variables $\{X_1, \ldots, X_{n-1}\}$. Lemma 4(b), (c) and (d) show that $\succ_{N_1'}$ determines the total ordering $\succ_{N_1}$, by extending $\succ_{N_1'}$ with the following sequence of values for variable X_n: 1, 0, 0, 1, 1, 0, 0, 1,.... Similarly, for $\succ_{N_2}$. By Lemma 4(a), $\succ_{N_1'}$ and $\succ_{N_2'}$ are both strict total orders, so, by induction, are equal, and hence $\succ_{N_1}$ and $\succ_{N_2}$ are equal. □

Continuing Example 1, write assignment u to variables X_1 and X_2 as a pair of the Boolean values; for example, the assignment $X_1 = 1$, $X_2 = 0$ is abbreviated to $(1, 0)$. It can be seen that if we use the following conditional preference table for X_3 then we arrive at the total ordering of outcomes: $1 \succ^{X_3}_{(1,1)} 0$, $0 \succ^{X_3}_{(1,0)} 1$, $1 \succ^{X_3}_{(0,0)} 0$, and $0 \succ^{X_3}_{(0,1)} 1$. This argument can be generalised to show that for any n, there exists a CP-net on n Boolean variables which totally orders the outcomes. In fact it can be seen that $1 \succ^{X_i}_u 0$ if and only if the tuple u contains an even number of zeros:

Proposition 8 *Let V be a set of variables, which we label as $\{X_1, \ldots, X_n\}$. Define a CP-net N as follows:*

(a) the graph H is defined to be $(X_j, X_i) : 1 \leq j < i \leq n$, so that the set U_i of parents of variable X_i is equal to $V_{i-1} = \{X_1, \ldots, X_{i-1}\}$;

(b) for $i = 1, \ldots, n$ and $u \in \underline{U_i}$, the relation $\succ^{X_i}_u$ is defined by $1 \succ^{X_i}_u 0$ if and only if the tuple u contains an even number of zeros, i.e., there is an even number of variables X_j in U_i with $u(X_j) = 0$. So, $0 \succ^{X_i}_u 1$ if and only if the tuple u contains an odd number of zeros.

Then $\succ_N$ is a strict total order on $\underline{V}$ with maximum element $(1, \ldots, 1)$.

Proof of Theorem 2: Let n be a natural number. Proposition 8 shows that there exists at least one CP-net N_0 satisfying the conditions (i) and (ii) of the theorem. Suppose that N is any CP-net on V satisfying conditions (i) and (ii). Proposition 7 implies that $\succ_N$ equals $\succ_{N_0}$. For each variable X_i, the set of parents of X_i in N must be the same as the set of parents of X_i in N_0, i.e., $\{X_1, \ldots, X_{i-1}\}$. This is because we can't have more parents of X_i in N than in N_0 without contradicting (ii), and in N_0, the preference over X_i genuinely depends on each of these variables, so if we omit any X_j from V_{i-1} we can't get an equivalent conditional preference table. ($1 \succ^{X_i}_u 0$ holds if and only if the tuple u contains an even number of zeros, so if we were to omit any variable from u, we can't generate equivalent preferences.) Given the choice of parents, the relation $\succ_N$ determines all the local relations $\succ^{X_i}_u$, since we have $x \succ^{X_i}_u x'$ $\iff$ $tux \succ_N tux'$, where t is any assignment to variables $V - (U \cup \{X_i\})$. This shows that N is actually equal to N_0. □

4 Determining consistency, totally ordering outcomes and constrained optimisation

Section 4, as well as Section 5, 6 and 7, are concerned with the three interrelated topics of determining consistency of a cp-theory, totally ordering sets of outcomes, and constrained optimisation; these are described below.

Determining consistency of a cp-theory Γ. Γ is consistent if and only if there exists some strict total order $>$ extending the preference relation $>_\Gamma$, which, by Theorem 1, is if and only if $>_\Gamma$ is acyclic. We focus on a particular kind of strict total order: one generated by a complete search tree (or "cs-tree"), as used in backtracking search for solutions of a constraint satisfaction problem (CSP); the associated strict total order is the order in which outcomes are visited by such a search tree. For the fully acyclic case, when $G(\Gamma)$ is acyclic (see Section 2.2.3), testing consistency is relatively easy (see Section 5.1). More generally, the problem of determining consistency of a cp-theory is extremely hard, indeed PSPACE-complete: see (Goldsmith, Lang, Truszczyński, & Wilson, 2008), Theorem 3. We give a necessary (see Section 4.1) and some sufficient conditions (see Section 6) for consistency, that have much lower complexity (see Proposition 11 in Section 4.1 and Proposition 24 in Section 6.5).

Totally ordering sets of outcomes. It will often be the case that not all complete assignments are available. Suppose that we have a set $\Omega \subseteq \underline{V}$ of possible outcomes which the user needs to choose between, and we have elicited their preferences as a cp-theory Γ. We wish to display the outcomes in some order, showing them the best ones first. A basic requirement is that if α is preferred to β then α appears before β, since the user is more interested in outcome α than outcome β. Thus, we are concerned with the following task: given cp-theory Γ and subset Ω of outcomes, construct a total order on Ω which extends $>_\Gamma$ restricted to Ω.

The set Ω of available outcomes might be very large; in particular it might be expressed implicitly as the set of solutions of a constraint satisfaction problem. Then we won't be able to display all of them, but just, say, some number K of them. This gives rise to the following related problem: Given a number K, generate outcomes $\alpha_1, \ldots, \alpha_K \in \Omega$ such that for all $j = 1, \ldots, K$, if $\beta >_\Gamma \alpha_j$ for some $\beta \in \Omega$ then $\beta = \alpha_i$ for some $i < j$, i.e., if an element of Ω is preferred to α_j then it occurs before α_j in the generated list of outcomes. cs-trees can be used for these total ordering tasks (see Sections 4.4, 5.1, and 6.4).

Optimisation. Given cp-theory Γ, we say that outcome α is optimal if there exists no outcome β that dominates it, i.e., is such that $\beta >_\Gamma \alpha$. It can easily be seen that outcome α is optimal if and only if it is a solution of a particular constraint satisfaction problem. This implies that checking whether an outcome is optimal or not can be performed very efficiently, and finding an optimal outcome can be solved using CSP technology. Let Ω be a set of outcomes. We say that outcome $\alpha \in \Omega$ is optimal in Ω if there exists no outcome $\beta \in \Omega$ such that $\beta >_\Gamma \alpha$. If Ω is represented as the set of solutions of a CSP then we refer to such a task as *constrained optimisation*.

Given cp-theory Γ, we define an upper approximation $\gg$ to be a strict partial order containing the preference relation $>_\Gamma$, and we say that it is a polynomial upper approximation if $\alpha \gg \beta$ can be determined in polynomial time for any outcomes α and β (see Section 4.4). Specific polynomial upper approximations

are defined in Sections 5.2, 6.3 and 7.6. We show how a polynomial upper approximation can be useful for finding a set of optimal outcomes of a constrained optimisation problem: by using a search tree to generate solutions in an order consistent with $>_\Gamma$, and using an upper approximation $\gg$ to eliminate outcomes which could be non-optimal in Ω. They can also be used to totally order a small set Ω of outcomes: for each $\alpha, \beta \in \Omega$, we determine if $\alpha \gg \beta$ holds, thus determining $\gg$ restricted to Ω, and choose a total order on Ω compatible with this (see Section 4.5).

This section (Section 4) describes the basic mathematical notions and approaches: we define a necessary condition for consistency, called local consistency; we define cs-trees (complete search trees)—which can be used for showing consistency of a cp-theory—and give some basic properties of them; we consider the problem of constrained optimisation and describe our general approach, based on cs-trees and upper approximations. Section 5 below considers the problems of determining consistency, and ordering outcomes for the fully acyclic case, and shows how a polynomial upper approximation can be defined. In Sections 6 and 7 approaches for these tasks are derived for more general cp-theories.

Section 4.1 defines the "local consistency" property; a cp-theory is not locally consistent if and only if there exists some outcome α and some variable X such that there exists a worsening swapping (flipping) sequence from α to itself that just changes variable X. Local consistency is thus a necessary condition for a cp-theory to be consistent, which can often be determined efficiently.

Section 4.2 defines cs-trees and their associated strict total orders, and Section 4.3 gives a precise characterisation for when a cs-tree order satisfies a cp-theory. The cs-tree is a kind of lexicographic order where both the value and the variable orderings are conditional on the values of more important variables. Search trees have previously been used in the context of CP-nets and TCP-nets, in particular, in (Boutilier, Brafman, Domshlak, Hoos, & Poole, 2004b) and (Brafman et al., 2006). Section 4.4 considers the problem of totally ordering a set of outcomes according to the preference ordering $>_\Gamma$, which can be solved if one can construct a compact representation of a satisfying cs-tree (see Sections 5.1, 6.4 and 7 below for such constructions) or, if the set is small, using an upper approximation of $>_\Gamma$, i.e., a strict partial order containing $>_\Gamma$ (see Sections 5.2, 6.3 and 7.6, below for definitions of polynomial upper approximations in different situations). Section 4.5 considers the problems of optimisation and constrained optimisation, showing that one can use a cs-tree and an upper approximation to generate some of the optimal solutions of a set of constraints.

4.1 Local consistency

In this section we consider a necessary condition for consistency, called *local consistency*. In certain cases, it's clear that a cp-theory Γ is not consistent, by just looking at local conditions: if there's a sequence of worsening swaps from some outcome α to itself, which just changes the values of a single variable X.

If this does not hold, then we say that Γ is locally consistent. As well as defining local consistency, the definition below introduces the local ordering $\succ_a^X$ induced by the cp-theory on the domain of variable X given assignment a.

Definition 4 (local consistency and local ordering $\succ_a^X$) *Fix conditional preference theory Γ on V, and consider some variable $X \in V$, set of variables $A \subseteq V$ and assignment $a \in \underline{A}$ to A. Say that ordered pair (x, x') of values of X* is validated by *a if there exists some statement $(u : x > x' [W]) \in \Gamma$ such that a extends u (i.e., u is a projection of a). Define the local ordering $\succ_a^X(\Gamma)$ (abbreviated to $\succ_a^X$) on $\underline{X}$ to be the transitive closure of the set of all pairs (x, x') validated by a. We say that Γ is* locally consistent *if $\succ_\alpha^X$ is irreflexive for all variables X and outcomes α.*

If Γ is not locally consistent then there exists outcome α, variable X and a sequence $x_1, \ldots, x_k$ of values of X with associated statements in Γ, $(u_i : x_i > x_{i+1} [W_i])$, such that $\alpha \models u_i$, and $\alpha(X) = x_1 = x_k$. This gives a worsening swapping sequence from α to α (only involving changing variable X), thus implying that Γ is not consistent, by Theorem 1. Therefore, local consistency is a necessary condition for consistency:

Proposition 9 *If cp-theory Γ is consistent then it is locally consistent.*

The set of statements Γ in Example A in Section 2.2.2 (and also Example B in Section 3.2) is easily seen to be locally consistent. However, if φ_5 were changed to φ_5' equalling $\overline{\mathbf{o}} : \overline{\mathbf{p}} > \mathbf{p} [\emptyset]$ then Γ would no longer be locally consistent as φ_5' and $\varphi_3 = (\mathbf{n} : \mathbf{p} > \overline{\mathbf{p}} [\emptyset])$ would give conflicting preferences for X_3 under the conditions $\mathbf{n}\,\overline{\mathbf{o}}$. Let $\alpha = \mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}$. Then $\succ_\alpha^{X_3}$ is not irreflexive since $\overline{\mathbf{p}} \succ_\alpha^{X_3} \mathbf{p}$ (since $(\overline{\mathbf{p}}, \mathbf{p})$ is validated by α using φ_5'), and $\mathbf{p} \succ_\alpha^{X_3} \overline{\mathbf{p}}$ using φ_3, so $\overline{\mathbf{p}} \succ_\alpha^{X_3} \overline{\mathbf{p}}$. Γ would no longer be consistent as $>_\Gamma$ is no longer irreflexive: we have $\mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}} >_\Gamma \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p} >_\Gamma \mathbf{n}\,\overline{\mathbf{o}}\,\overline{\mathbf{p}}$ so $\alpha >_\Gamma \alpha$.

Local ordering $\succ_a^X$ is monotonic with respect to a, i.e., if partial tuple b extends a then $\succ_b^X$ extends $\succ_a^X$, i.e., if $x \succ_a^X x'$ holds then $x \succ_b^X x'$ holds.

Clearly, relation $\succ_a^X$ can be generated in polynomial time for a given tuple a and variable X, by selecting all φ in Γ with $X_\varphi = X$ and such that $a(U_\varphi) = u_\varphi$, recording the associated pairs (x_φ, x'_φ), and computing the transitive closure of this set of pairs of values of X.

Unsurprisingly, the converse of Proposition 9 does not hold:

Example 2 *Let $V = \{X_1, X_2\}$ with $\underline{X_1} = \{x_1, \overline{x}_1\}$, and $\underline{X_2} = \{x_2, \overline{x}_2\}$. Let Γ be the pair of statements $\top : x_1 > \overline{x_1} [\{X_2\}]$ and $\top : x_2 > \overline{x}_2 [\{X_1\}]$, which is easily seen to be locally consistent (since there is no statement preferring $\overline{x}_1$ over x_1, nor $\overline{x}_2$ over x_2). However, we have $x_1\overline{x}_2 >_\Gamma \overline{x}_1 x_2$ because of the first statement, and $\overline{x}_1 x_2 >_\Gamma x_1\overline{x}_2$ because of the second statement, so $x_1\overline{x}_2 >_\Gamma x_1\overline{x}_2$ and Γ is therefore inconsistent. For another example, see Example 1 of (Goldsmith et al., 2008).*

The following lemma, which is important in Section 6.1, follows easily from the definitions (since (x, x') is validated by $\alpha(A)$ if and only if (x, x') is validated

by α). It states that the local ordering $\succ_\alpha^X$ is unchanged if one eliminates irrelevant variables from α.

Lemma 5 *Let α be an outcome, let $X \in V$ be a variable, and let A be a set of variables satisfying the following property: for all $\varphi \in \Gamma$ such that $X_\varphi = X$, if $\alpha \models u_\varphi$ then $A \supseteq U_\varphi$. It follows that $\succ_\alpha^X = \succ_{\alpha(A)}^X$.*

For $X \in V$, recall that $U_X = \mathrm{Pa}_{H(\Gamma)}(X)$ is the set of parents of X with respect to $H(\Gamma)$ (see Section 2.2.3). Hence, $Y \in U_X$ if and only if there exists $\varphi \in \Gamma$ with $X_\varphi = X$ and $U_\varphi \ni Y$, so that $U_X = \bigcup_{\varphi \in \Gamma, X_\varphi = X} U_\varphi$. Lemma 5 implies the following result, which shows that local consistency can be determined using just the local orderings based on the set of parents of each variable.

Proposition 10 *Γ is locally consistent if and only if for all $X \in V$ and $u \in \underline{U_X}$, $\succ_u^X$ is irreflexive.*

Proof: Suppose that Γ is locally consistent, and consider any $X \in V$ and $u \in \underline{U_X}$. Choose any α extending u, so we have $\succ_\alpha^X \supseteq \succ_u^X$. Local consistency implies that $\succ_\alpha^X$ is irreflexive and hence $\succ_u^X$ is irreflexive.

Conversely, suppose that for all $X \in V$ and $u \in \underline{U_X}$, $\succ_u^X$ is irreflexive. Consider any variable X and outcome α. Then $\succ_\alpha^X$ is irreflexive since, by Lemma 5, it equals $\succ_u^X$, where $u = \alpha(U_X)$. □

Proposition 10 shows that local consistency can be checked efficiently if all the parents sets U_X are small: for each X and for each $u \in \underline{U_X}$, we compute $\succ_u^X$, by taking the transitive closure of all the pairs (x, x') of values of X validated by u. Hence, if the sizes of the parents sets and the sizes of the domains are bounded by a constant, then determining local consistency is polynomial.

Example 3 *Consider the set $\Gamma = \{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5\}$ of preference statements on variables $\{X_1, X_2, X_3\}$, which are those of Example B with variable X_4 deleted. φ_1 equals $\top : x_1 > \overline{x}_1\,[\{X_2, X_3\}]$. $\varphi_2 = x_1 : x_3 > \overline{x}_3\,[\{X_2\}]$; φ_3 equals $x_1 : x_2 > \overline{x}_2\,[\emptyset]$. $\varphi_4 = \overline{x}_1 : x_2 > \overline{x}_2\,[\{X_3\}]$, and $\varphi_5 = \overline{x}_1 : \overline{x}_3 > x_3\,[\emptyset]$.*

It is clear, for any outcome α, that $\succ_\alpha^{X_1}$ is irreflexive, since there is no preference statement φ in Γ that involves a preference for $\overline{x}_1$ over x_1, i.e. which is such that $x_\varphi = \overline{x}_1$ and $x'_\varphi = x_1$. Similarly, for X_2. Regarding X_3, we have $x_3 \succ_\alpha^{X_3} \overline{x}_3$ if and only if $\alpha(X_1) = x_1$, and $\overline{x}_3 \succ_\alpha^{X_3} x_3$ if and only if $\alpha(X_1) = \overline{x}_1$, and so, for no outcome α do we have both preferences, which proves that for all outcomes α, $\succ_\alpha^{X_3}$ is irreflexive, and hence Γ is locally consistent.

An alternative way of proving local consistency is to use Proposition 10. For X_3 this involves considering the two possible assignments to the set $\{X_1\}$ of parents of X_3. We have $\succ_{x_1}^{X_3} = \{(x_3, \overline{x}_3)\}$, and $\succ_{\overline{x}_1}^{X_3} = \{(\overline{x}_3, x_3)\}$.

In general, determining local consistency is coNP-complete:

Proposition 11 *The problem of deciding whether a cp-theory Γ is locally consistent is coNP-complete.*

However, often checking local consistency will be easy; in particular, as discussed above, when the sets U_X are small (where U_X is the set of variables that variable X depends on), as in intended applications of CP-nets and TCP-nets, one can efficiently construct each local ordering $\succ_u^X$ explicitly, thus determining whether local consistency holds or not, using Proposition 10. (For CP-nets and TCP-nets, it is assumed that these local orderings $\succ_u^X$ have already been computed, or directly elicited; they are also assumed to be strict partial orders, so local consistency is guaranteed.) To give another example, when all the variables are binary (i.e., two-valued), local consistency can be determined in time proportional to $|\Gamma|^2|V|$, (assuming that the domain sizes of variables are bounded by a constant).

4.2 cs-trees

In this section we describe complete search trees (cs-trees), and their associated total orderings over outcomes. In Sections 4.3, 5, 6, and 7 we will show how under certain conditions a search tree ordering will satisfy a cp-theory, which leads to methods for proving consistency of a cp-theory.

A cs-tree (or "complete search tree") is a rooted directed tree with its $|\underline{V}|$ leaves corresponding to outcomes (see Figure 2). Associated with each non-leaf node r is a variable Y_r, which is instantiated with a different value in each of the node's $|\underline{Y_r}|$ children, and also an ordering $\succ_r$ of the values of Y_r. So, a directed edge in the tree corresponds to an instantiation of one of the variables. Paths in the tree from the root down to a leaf node correspond to sequential instantiations of all the variables V.

Definition 5 (cs-tree ("complete search tree")) *A* cs-tree *over variables V is defined to be a rooted directed tree, where nodes and edges have associated labels as defined below.*

Each directed edge e from node r to node r' is associated with a variable Y_e and a value y_e of Y_e (corresponding to the assignment $Y_e = y_e$). We say that r' is a child of r.

Each node r has the following associated labels:

(a) a set of variables $A_r \subseteq V$ (the assigned *variables, i.e., the variables assigned above that node in the cs-tree);*

(b) an assignment a_r to variables A_r (corresponding to the assignments made above that node).

If a node has no children then we say that it is a leaf node*; otherwise we say that the node is a* body node*. A node is a leaf node if and only if its associated set of variables A_r is equal to the whole set of variables V. If r is a body node then we also associate the following two labels with it.*

(c) a variable $Y_r \in V - A_r$ (the next variable to be instantiated);

(d) an ordering $\succ_r$ on the domain $\underline{Y_r}$ of Y_r (the "value ordering" at that node).

For leaf node r we define the associated leaf tuple *to be $\langle A_r, a_r \rangle$. For body node r we define the associated* body tuple *to be $\langle A_r, a_r, Y_r, \succ_r \rangle$.*

The (unique) root node we write as r^*, and define A_{r^*} to be the empty set, and hence we have a_{r^*} equals $\top$, the assignment to the empty set.

Body node r with associated variable Y_r has $|\underline{Y_r}|$ children, so has $|\underline{Y_r}|$ edges coming from it. Each such edge e has associated variable $Y_e = Y_r$ and a different associated value y_r. If e goes from node r to r' then $A_{r'} = A_r \cup \{Y_r\}$, and $a_{r'}$ is the tuple formed by extending a_r with the assignment $Y_e = y_e$.

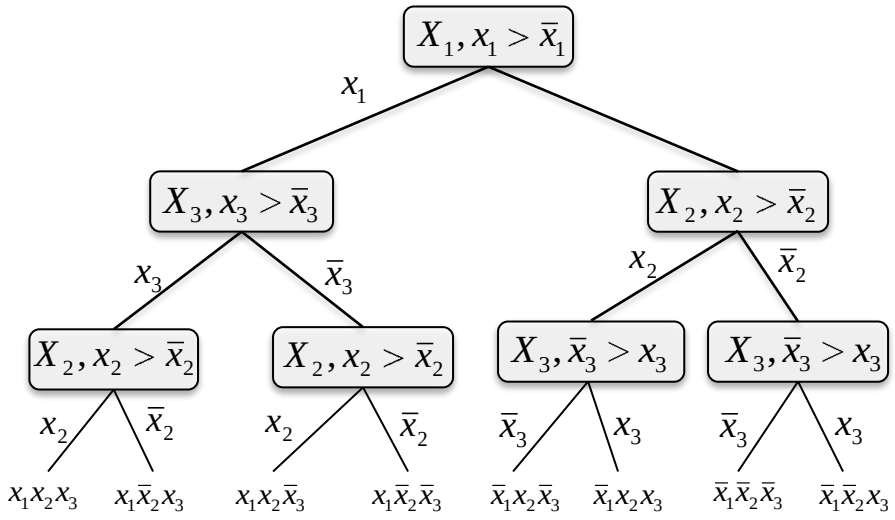

Figure 2: A cs-tree σ over binary variables $\{X_1, X_2, X_3\}$. For each body node r we include its associated variable Y_r and the ordering $\succ_r$. The associated ordering $>_\sigma$ on outcomes is given by the ordering of leaf nodes (at the bottom), starting from the left, i.e., $x_1x_2x_3 >_\sigma x_1\overline{x}_2x_3 >_\sigma x_1x_2\overline{x}_3 >_\sigma \cdots >_\sigma \overline{x}_1\overline{x}_2x_3$.

Example 4 *We define a search tree σ over binary-valued variables $V = \{X_1, X_2, X_3\}$. This is illustrated in Figure 2. The root node is at the top, and the eight leaf nodes are at the bottom. In the figure, we show, for each node r, the associated variable Y_r, and, for body nodes, the local ordering $\succ_r$ on the values of Y_r (omitting the components A_r and a_r).*

The root node has associated tuple $\langle \emptyset, \top, X_1, (x_1, \overline{x}_1) \rangle$, so that X_1 is the variable which is assigned at the root node, with ordering $x_1 \succ \overline{x}_1$. Below it, following the edge associated with the assignment $X_1 = x_1$, is the node with tuple $\langle \{X_1\}, x_1, X_3, (x_3, \overline{x}_3) \rangle$. The first component is the set of variables assigned above that node, i.e., $\{X_1\}$.

The bottom left node is the leaf node $\langle \{X_1, X_2, X_3\}, x_1x_2x_3 \rangle$. The first component of this node is the set of variables assigned in the path from the root to

that node, which is always equal to V for a leaf node. The second component is the assignment to V along that path. Notice that the variable orderings vary within the cs-tree. For example, the leftmost path in the tree has associated variable ordering X_1, X_3, X_2, whereas the rightmost path has variable ordering X_1, X_2, X_3. Also the local (value) orderings of a variable can be different in different nodes. For example, one node has ordering $x_3 \succ \overline{x}_3$, whilst two nodes have ordering $\overline{x}_3 \succ x_3$.

We have the following properties of cs-trees:

Lemma 6 *Let σ be a cs-tree over variables V.*

- *For each node r, a_r is the set of assignments $Y_e = y_e$ made in edges e on the path from the root to r.*
- *For each outcome α there exists exactly one leaf node r with $a_r = \alpha$, so we can associate leaf nodes with outcomes. The set of leaf nodes is therefore in one-to-one correspondence with the set $\underline{V}$ of outcomes.*

Definition 6 (path to outcome) *Let σ be a cs-tree over variables V, and let α be an element of $\underline{V}$. The* path (in σ) to α *is defined to be the sequence of nodes along the directed path from the root node to the leaf node corresponding to α.*

The set of nodes in the path to α thus consists of all nodes r such that α extends a_r. Write the path to α as $r_0, r_1, \ldots, r_n$, where $r_0 = r^*$, the root node. Then for each $i = 1, \ldots, n$, we have $A_{r_i} = \{Y_{r_0}, \ldots, Y_{r_{i-1}}\}$, the set of variables instantiated in nodes between r_i and the root node. We also have $a_{r_i} = \alpha(A_{r_i})$. The path to α has an associated ordering of variables, namely, $Y_{r_0}, \ldots, Y_{r_{n-1}}$. A cs-tree σ thus associates an ordering of V with any outcome α.

Definition 7 (cs-tree node divides outcomes) *Let σ be a cs-tree over variables V, and let α and β be elements of $\underline{V}$. We say that node r divides α and β if r is the last node in the path to α which is also in the path to β.*

Clearly, for any pair of different outcomes α and β, there exists a unique node in σ which divides α and β. We associate a strict total order $>_\sigma$ with cs-tree σ as follows:

Definition 8 (cs-tree order on outcomes) *Let σ be a cs-tree over variables V, and let α and β be elements of $\underline{V}$. Define $\alpha >_\sigma \beta$ if and only if $\alpha \neq \beta$ and $\alpha(Y_r) \succ_r \beta(Y_r)$, where r is the node that divides α and β.*

In other words, we compare two outcomes by considering the lowest (deepest) node r that is above both of them, and use the ordering $\succ_r$ to compare them. If, as in Figure 2, we draw the cs-tree σ with the directed edges pointing downwards from the root, and the edges from a node r in the order $\succ_r$, with the best being leftmost, then $\alpha >_\sigma \beta$ if and only if leaf node $\langle V, \alpha \rangle$ appears to the left of $\langle V, \beta \rangle$.

Similarly, suppose, in a depth-first search, we instantiate at each node r the best value according to $\succ_r$ first, and on backtracking, the values in the order $\succ_r$; then α is reached before β if and only if $\alpha >_\sigma \beta$. This means that it is very easy to generate the first K elements in the cs-tree order.

Example 5 *Consider the cs-tree σ in Figure 2. To determine how σ orders outcomes $x_1\overline{x}_2x_3$ and $x_1x_2\overline{x}_3$, we start from the root, and follow corresponding edges until we find the node that divides them. The root does not divide them because they agree on its associated variable X_1, each taking value x_1. So, we then consider the node at the end of the x_1 edge. This has associated variable X_3. Since the two outcomes differ on X_3, this is the node that divides them. Since the value ordering at this node prefers x_3 to $\overline{x}_3$ we have $x_1\overline{x}_2x_3 >_\sigma x_1x_2\overline{x}_3$.*

The ordering $>_\sigma$ on outcomes is given by the ordering of leaf nodes. $>_\sigma$ is thus the transitive closure of the following preference comparisons: $x_1x_2\overline{x}_3 >_\sigma x_1\overline{x}_2x_3 >_\sigma x_1x_2\overline{x}_3 >_\sigma x_1\overline{x}_2\overline{x}_3 >_\sigma \overline{x}_1x_2\overline{x}_3 >_\sigma \overline{x}_1x_2x_3 >_\sigma \overline{x}_1\overline{x}_2\overline{x}_3 >_\sigma \overline{x}_1\overline{x}_2x_3$.

Compact representations of cs-trees. To construct a cs-tree, for each node (starting at the root node) we have to choose an associated variable Y and a total ordering $\succ$ on its domain $\underline{Y}$. (Similarly, when using a complete search for finding a solution of a constraint satisfaction problem, at each node of the explored search tree, we have to choose a variable to instantiate next, and a value ordering over the domain of that variable.) This determines the components A_r and a_r for each child r of this node, and we choose associated variable $Y_r \in V - A_r$ and total ordering $\succ_r$ on $\underline{Y_r}$. Applying this iteratively from the root to the leaf nodes generates a cs-tree.

A cs-tree σ is an exponentially large object, so we will often not be able to generate it explicitly. However, to define a cs-tree implicitly, all we have to do is to define a function g that takes as input appropriate tuples $a \in A$ and returns a pair $(Y, \succ)$ where $Y \in V - A$ and $\succ$ is a total ordering on $\underline{Y}$. The domain D_g of the function could be the set of all possible tuples a, but need not be: it is sufficient that it contain the assignment $\top$ to the empty set, and satisfy the condition that if D_g contains a and $g(a) = (Y, \succ)$ then D_g contains tuple ay for every assignment y to Y. Such a function g therefore specifies a compact representation of a search tree σ_g (given that g is specified in a compact way). For any search tree, there exists g such that σ equals σ_g.

Given that any value of the function g can be computed efficiently, we can efficiently perform important operations with σ_g (without constructing σ_g explicitly). In particular, for any two outcomes α and β, we can determine if $\alpha >_{\sigma_g} \beta$: we generate the nodes in the path to α until we find the node that divides α and β. In addition, given any number K, we can efficiently generate the best K outcomes according to $>_{\sigma_g}$, by generating the nodes in σ_g in a depth first search manner, as explained below, until K outcomes have been generated.

Using a cs-tree σ for backtracking search. As alluded to above, cs-trees are essentially the same as search trees used for solving CSPs, and they can be used to control the depth-first search for solutions. Any node of the search

tree corresponds to a node r of the cs-tree σ. A_r is the set of variables already assigned, and a_r is their assignment. We start the search at the root node. At each node r, we instantiate variable Y_r next. Relation $\succ_r$ gives the value ordering: we instantiate first the best value according to $\succ_r$. On backtracking to this search node, we remove the previously tried value of Y_r from the domain of Y_r, and instantiate Y_r with the best remaining value according to $\succ_r$. If the domain of Y_r is now empty, we backtrack to the parent node of r. When we reach a leaf node r, i.e., when all the variables are instantiated, we record the associated outcome a_r, and backtrack to its parent node.

Given a set C of constraints[6]on variables V we can use this backtracking search method in the usual way to generate solutions of C, i.e., outcomes satisfying all the constraints in C. Standard CSP techniques such as maintaining arc consistency can be used to reduce the search. The solutions of C will be generated in decreasing order of $>_\sigma$.

4.3 cs-trees satisfying cp-theories

In this section, sufficient conditions are given for a cs-tree ordering to satisfy a cp-theory. These conditions will be used in Sections 5, 6 and 7 to give sufficient conditions for a cp-theory to be consistent.

A cs-tree σ is said to satisfy a cp-theory Γ if its associated total order $>_\sigma$ extends the preference relation $>_\Gamma$ (see Definition 9, below). A cp-theory is *conditionally acyclic* if there exists some cs-tree satisfying it. This immediately implies (Proposition 12) that conditional acyclicity implies consistency (although the converse does not hold: see Example 6). Proposition 13 gives a pair of necessary and sufficient conditions for a cs-tree to satisfy a cp-theory, the first based on the variable orderings in paths in the cs-tree (or, equivalently, on the variable Y_r that can be chosen for a node r given the current assignment a); the second condition is on the value ordering that is chosen for a node r in the cs-tree. Proposition 14 then gives a stronger pair of sufficient conditions (which corresponds to "strong satisfaction", developed in Section 6.1); the first condition, on the variable orderings, is that for any preference statement $\varphi \in \Gamma$, set of variables U_φ must appear before X_φ which must appear before variables W_φ, on the path from the root node of the cs-tree to any outcome α which extends u_φ. This condition enables a simpler form for the second, local ordering condition, namely, that the total ordering $\succ_r$ for node r in the cs-tree extends the appropriate local ordering, which is a strict partial order, given that the cp-theory is locally consistent (this implies that the second condition is easy to satisfy, since a partial order can always be extended to a total order).

Definition 9 *Let Γ be a cp-theory. We say that cs-tree σ satisfies Γ if the associated total order $>_\sigma$ satisfies Γ, i.e., if $>_\sigma \supseteq >_\Gamma$ (by Lemma 2(i)).*

[6]A constraint c on variables V is a relation on S_c for some $S_c \subseteq V$, so is interpreted as a subset of $\underline{S_c}$. An outcome $\alpha \in \underline{V}$ satisfies a constraint c if $\alpha(S_c) \in c$. Outcome α is said to be a solution of set of constraints C on V if it satisfies each constraint in C.

We say that Γ *is* conditionally acyclic[7] *if there exists a cs-tree satisfying* Γ.

If a cp-theory Γ is conditionally acyclic then there exists a cs-tree σ with $>_\sigma$ satisfying Γ, implying that Γ is consistent:

Proposition 12 *Let* Γ *be a cp-theory. If* Γ *is conditionally acyclic then it is consistent.*

The following example shows that the converse does not hold: there exist consistent cp-theories that are not conditionally acyclic.

Example 6 *Let cp-theory* Γ *on variables* $\{X_1, X_2\}$ *consist of the four preference statements* $x_1 : x_2 > \overline{x}_2\,[\emptyset]$; $x_2 : x_1 > \overline{x}_1\,[\emptyset]$; $\overline{x}_1 : \overline{x}_2 > x_2\,[\emptyset]$, *and* $\overline{x}_2 : \overline{x}_1 > x_1\,[\emptyset]$. *These statements imply that* x_1x_2 *and* $\overline{x}_1\overline{x}_2$ *are both preferred to* $x_1\overline{x}_2$ *and* $\overline{x}_1x_2$. Γ *is consistent, but is not conditionally acyclic. To prove a contradiction, assume that there exists a cs-tree* σ *which satisfies* Γ. *Suppose that* X_1 *is the variable associated with the root node* r^* *(i.e., the top, or, most important variable for* σ*). Then the node value ordering* $\succ$ *is either* $x_1 \succ \overline{x}_1$, *or* $\overline{x}_1 \succ x_1$. *If the former, then* $x_1\overline{x}_2 >_\sigma \overline{x}_1\overline{x}_2$; *if the latter then* $\overline{x}_1x_2 >_\sigma x_1x_2$. *Both of these are incompatible with* Γ, *so* X_1 *cannot be the top variable. Similarly,* X_2 *cannot be the top variable, so there exists no cs-tree satisfying* Γ.

The following result gives equivalent conditions for a search tree to satisfy a cp-theory. Condition (1) states that a variable precedes less important variables on relevant paths of the cs-tree, and condition (2) states that the node value ordering must be compatible with the preference statements in the cp-theory. More precisely, consider some statement $u : x > x'\,[W]$ in the cp-theory Γ, where x and x' are values of variable X. (1) requires that on paths to outcomes that extend u, X appears before each element of W. (2) requires that for any node r whose context a_r is compatible with u and where X is chosen, x is preferred to x' in the node value ordering.

Proposition 13 *Let* Γ *be a cp-theory, and let* σ *be a cs-tree. The following pair of conditions is sufficient for* σ *to satisfy* Γ. *If the domain of each variable has at least two elements then the pair of conditions is also necessary for* σ *to satisfy* Γ.

(1) For any $\varphi \in \Gamma$ *and any outcome* α *such that* $\alpha \models u_\varphi$*: on the path from the root to* α, X_φ *appears before each element of* W_φ*;*

(2) for any body node r *and any* $\varphi \in \Gamma$ *such that* $X_\varphi = Y_r$ *and for any* u_φ *compatible with* a_r, *we have* $x_\varphi \succ_r x'_\varphi$.

[7] Our terminology differs here from that used for TCP-nets (and also the terminology used in (Wilson, 2004a)); conditional acyclicity for TCP-nets (Brafman et al., 2006) corresponds with our property "context-uniform conditional acyclicity" (see Section 6.2), which is a very much stronger condition, assuming, in particular, that $H(\Gamma)$ is acyclic.

(The condition that the domain of each variable has at least two elements is not restrictive, since any variable with a singleton domain can be eliminated.)

One interesting aspect of the above result is that it shows that it is not necessary for parents of a variable to precede the variable in the paths in a cs-tree, i.e., we can have, for $\varphi \in \Gamma$, elements of U_φ further from the root (i.e., less important) than X_φ. However, in this paper we are mainly concerned with cs-trees where parents precede children (at least in relevant contexts). For this case we have the following version of Proposition 13, which will be used in Section 6. It again consists of a condition on the variable orderings, and a condition on the node value orderings. Condition (1′) is a stronger form of (1) of Proposition 13, requiring that, for any statement $u : x > x' [W]$ in Γ, on paths where U is instantiated as u, it is instantiated before X which is instantiated before W. Given (1′), condition (2) of Proposition 13 can be expressed as (2′).

Proposition 14 *Let Γ be a cp-theory, and let σ be a cs-tree. Then σ satisfies Γ if the following pair of conditions hold:*

(1′) For any $\varphi \in \Gamma$ and any outcome α such that $\alpha \models u_\varphi$: on the path from the root to α, each element of U_φ appears before X_φ, which appears before each element of W_φ;

(2′) for any body node $r = \langle A, a, Y, \succ \rangle$, relation $\succ$ extends the local ordering $\succ_a^Y$ (see Definition 4).

Example 7 *Consider again the cp-theory Γ from Example 3, and the cs-tree σ from Example 4 and Figure 2. In particular, consider the preference statement $\varphi = x_1 : x_3 > \overline{x}_3 [\{X_2\}]$ in Γ, and any outcome α such that $\alpha \models x_1$, i.e., $\alpha(X_1) = x_1$. In the path to such an outcome, the variables appear in the order X_1, X_3, X_2, and so $U_\varphi = \{X_1\}$ appears before $X_\varphi = X_3$ which appears before $W_\varphi = \{X_2\}$. Thus, condition (1′) of Proposition 14 is satisfied, for this φ, and it can be easily confirmed also for other $\varphi \in \Gamma$. To illustrate condition (2′), consider the node associated with assignment x_1, which has value ordering $\succ$ given by $x_3 \succ \overline{x}_3$, and so $\succ$ is equal to $\succ_{x_1}^{X_3}$ (see Example 3). Condition (2′) can be confirmed for all other nodes also, so by Proposition 14, cs-tree σ satisfies Γ.*

4.4 Total ordering tasks and upper approximations of the preference relation

Suppose that we have a set Ω of possible outcomes which the user needs to choose between, and we have elicited their preferences as a cp-theory Γ. We wish to display the outcomes in some order $>$, showing them the best ones first. A basic requirement is that for $\alpha, \beta \in \Omega$, if α is preferred to β then α appears before β, i.e., if $\alpha >_\Gamma \beta$ then $\alpha > \beta$, since they are more likely to be interested in outcome α than outcome β. Thus, we are concerned with the following task: given cp-theory Γ and subset Ω of outcomes, construct a strict total order $>$ on Ω which extends $>_\Gamma$ restricted to Ω.

However, if Ω is very large, in particular if it is expressed implicitly as the set of solutions of a constraint satisfaction problem, then we won't be able to display all of them, but just, say, K of them. This gives rise to the following related problem: Given K, generate outcomes $\alpha_1, \ldots, \alpha_K \in \Omega$ such that for all $j = 1, \ldots, K$, if $\beta >_\Gamma \alpha_j$ for some $\beta \in \Omega$ then $\beta = \alpha_i$ for some $i < j$.

These ordering tasks can be solved if we can define a cs-tree satisfying Γ, since as shown in Section 4.2, we can then efficiently compare outcomes, and generate outcomes in the cs-tree order. Another approach is to use an upper approximation of the preference relation, as defined below.

Definition 10 (upper approximation) *Binary relation $\gg$ on $\underline{V}$ is said to be an upper approximation of the preference relation $>_\Gamma$ if $\gg$ is a strict partial order extending $>_\Gamma$. We say that $\gg$ is a polynomial upper approximation if for any outcomes α and β, whether or not $\alpha \gg \beta$ holds can be determined in polynomial time.*

Note that an upper approximation of $>_\Gamma$ exists if and only if $>_\Gamma$ is irreflexive, i.e., if and only if (by Theorem 1) Γ is consistent.

The result below, which follows easily from the definitions, describes a general way of generating upper approximations, which will be used below in sections 5.2, 6.3 and 7.6, for proving that certain relations are polynomial upper approximations. It states that the intersection of a non-empty set of cs-tree orderings satisfying a cp-theory is an upper approximation.

Proposition 15 *Let $\mathcal{R}$ be some non-empty set of cs-trees satisfying cp-theory Γ. Define relation $\gg^{\mathcal{R}}$ to be the intersection of $>_\sigma$ over σ in $\mathcal{R}$, so that, for outcomes α and β, $\alpha \gg^{\mathcal{R}} \beta$ if and only if $\alpha >_\sigma \beta$ holds for all $\sigma \in \mathcal{R}$. Then relation $\gg^{\mathcal{R}}$ is an upper approximation of $>_\Gamma$, i.e., a strict partial order containing $>_\Gamma$.*

Proof: Each relation $>_\sigma$ is transitive, and intersections of transitive relations are also transitive, so $\gg^{\mathcal{R}}$ is transitive; it is also irreflexive since any $>_\sigma$ is irreflexive, and so $\gg^{\mathcal{R}}$ is a strict partial order. For each $\sigma \in \mathcal{R}$, $>_\sigma \supseteq >_\Gamma$, which implies that their intersection, $\gg^{\mathcal{R}}$, contains $>_\Gamma$. □

For Ω consisting of just a few outcomes, a polynomial upper approximation $\gg$ can be used to order Ω in a way that is compatible with $>_\Gamma$: for each $\alpha, \beta \in \Omega$ we test if $\alpha \gg \beta$; this generates a strict partial order over Ω, i.e., the restriction of $\gg$ to $\Omega \times \Omega$. (This procedure can be speeded up if we can efficiently find a $\gg$-undominated element in Ω, i.e., an element $\alpha \in \Omega$ such that for all $\beta \in \Omega$ it is not the case that $\beta \gg \alpha$, since we can find such an element and then remove it from Ω and iterate.)

For larger Ω, or Ω defined implicitly as the set of solutions of a constraint satisfaction problem on V, our approach is to use an implicit representation of a cs-tree σ satisfying Γ: see Section 5.1, Section 6.4, and Section 7. This can also be used to generate the top K elements according to σ.

4.5 Constrained optimisation using polynomial upper approximation

This section first considers the task of finding optimal outcomes for a cp-theory Γ, i.e., outcomes that are not dominated by any outcome according to the preference relation $>_\Gamma$. It then considers the much harder problem of finding optimal solutions to a set of constraints, and shows how an upper approximation can be used to help find *some* optimal solutions.

Definition 11 *Given cp-theory Γ, we say that outcome α is ($>_\Gamma$-)*optimal *if there exists no outcome β such that $\beta >_\Gamma \alpha$. Let Ω be a set of outcomes. We say that outcome $\alpha \in \Omega$ is ($>_\Gamma$-)*optimal in Ω *if there exists no outcome $\beta \in \Omega$ such that $\beta >_\Gamma \alpha$. If C is a set of constraints on V we say that α is an ($>_\Gamma$-)*optimal solution of C *if α is $>_\Gamma$-optimal in the set of solutions of C.*

The optimal outcomes with respect to cp-theory Γ are precisely the solutions of a particular constraint satisfaction problem (CSP) C_Γ on V (cf. (Brafman & Dimopoulos, 2004) and Theorem 2 of (Domshlak, Prestwich, Rossi, Venable, & Walsh, 2006)). The point is that if outcome α does not satisfy some c_φ then it's not optimal since there's an improving swap just changing the value of X_φ from x'_φ to x_φ.

Proposition 16 *Let Γ be a cp-theory. Define C_Γ to be the set of constraints $\{c_\varphi : \varphi \in \Gamma\}$, where constraint c_φ on variables $U_\varphi \cup \{X_\varphi\}$ is $(U_\varphi = u_\varphi) \Rightarrow (X_\varphi \neq x'_\varphi)$. Then outcome $\alpha \in \underline{V}$ is $>_\Gamma$-optimal if and only if α is a solution of C_Γ.*

Furthermore, as observed in (Wilson, 2004b), if $H(\Gamma)$ is acyclic and (e.g.,) if Γ is locally consistent, then it's very easy to find $>_\Gamma$-optimal outcomes; by instantiating the variables in an order compatible with $H(\Gamma)$, one can reach a (any) solution (without having to backtrack).

The situation is much trickier when we have a set of constraints C on V, and we wish to find optimal solutions of C. If one is only interested in finding *some* outcomes which are optimal, then one can try to solve the CSP with constraints $C_\Gamma \cup C$. Any solution of this CSP will be an $>_\Gamma$-optimal solution of C since it is $>_\Gamma$-optimal in $\underline{V}$. However, $C_\Gamma \cup C$ may very well have no solutions, so we need some more general methods.

Finding all optimal solutions: Suppose one can find a cs-tree σ satisfying Γ. This can be used to generate the solutions of C in the order $>_\sigma$, by using the natural backtracking algorithm associated with σ (see Section 4.2). The first solution, α, that it generates will be $>_\Gamma$-optimal since $\beta >_\Gamma \alpha$ implies $\beta >_\sigma \alpha$ (and so β cannot be a solution of C). At each point in the search we have a set Ω^* of $>_\Gamma$-optimal solutions already found. When we find the next solution α we need to determine if there exists any $\beta \in \Omega^*$ with $\beta >_\Gamma \alpha$. If not, then α is a $>_\Gamma$-optimal solution and we add it to Ω^* (since α is not dominated by any solution γ found later, because $\alpha >_\sigma \gamma$, and so $\gamma \not>_\sigma \alpha$ and hence $\gamma \not>_\Gamma \alpha$.)

This algorithm, which is based on the approach used for CP-nets in (Boutilier et al., 2004b), is complete, with the final Ω^* being the set of all $>_\Gamma$-optimal solutions, and at each point the set Ω^* contains only $>_\Gamma$-optimal solutions. The problem with this algorithm (and the similar algorithms in (Boutilier et al., 2004b; Brafman et al., 2006)) is that determining if $\beta >_\Gamma \alpha$ (or not) will often be infeasible (Boutilier et al., 2004a; Domshlak & Brafman, 2002; Goldsmith et al., 2008) unless the problem is small, since it involves searching for swapping sequences, which generalise flipping sequences.

Finding some optimal solutions: However, suppose now we apply exactly the same form of algorithm, but replacing tests of the form $\beta >_\Gamma \alpha$ by $\beta \gg \alpha$, where $\gg$ is an upper approximation of $>_\Gamma$. The generated outcomes will be precisely the $\gg$-optimal solutions of C. However, from the definition of an upper approximation it immediately follows that if α is $\gg$-optimal then it is $>_\Gamma$-optimal. This algorithm will therefore generate some (but not usually all) the $>_\Gamma$-optimal solutions; if the tests $\beta \gg \alpha$ can be performed efficiently then generating $>_\Gamma$-optimal solutions in this way should be quite feasible.

There will often be a very large number of optimal solutions, and we may well only wish to report a small fraction of them; it is not necessarily important that the upper approximation is a close approximation, just that $\gg$ is sufficiently far from being a total order that there are still liable to be a good number of solutions which are $\gg$-optimal.

5 Consistency, and polynomial upper bound of preference relation: the fully acyclic case

In this section we consider fully acyclic cp-theories Γ, that is, those such that $G(\Gamma)$ is acyclic. Recall from Section 2.2.3 that $G(\Gamma)$ contains sets of edges $U_\varphi \rightarrow X_\varphi$ and $X_\varphi \rightarrow W_\varphi$ for all $\varphi \in \Gamma$; Γ is thus fully acyclic if and only if the set of variables V can be labelled as $\{Z_1, Z_2, \ldots, Z_n\}$ in such a way that for all $\varphi \in \Gamma$, if $Z_i \in U_\varphi$ then $i < j$ where $Z_j = X_\varphi$, and if $Z_k \in W_\varphi$ then $k > j$. It is shown in Section 5.1 how to construct a satisfying cs-tree for Γ if it is locally consistent. This implies (Theorem 3) that when $G(\Gamma)$ is acyclic, Γ is consistent if and only if it is locally consistent. In Section 5.2 we define an upper approximation for the preference relation $>_\Gamma$, which, as shown in Section 4.4, can be used for totally ordering sets of outcomes, and for constrained optimisation.

5.1 Generating a cs-tree satisfying fully acyclic Γ

Suppose that Γ is a locally consistent and fully acyclic cp-theory. Therefore, there exists a total order on the set of variables V that extends $G(\Gamma)$. Let us enumerate V as $Z_1, Z_2, \ldots, Z_n$ in a way which is compatible with $G(\Gamma)$, i.e., such that $(Z_i, Z_j) \in G(\Gamma)$ implies $i < j$. We can iteratively define the nodes of a cs-tree from the root to the leaves as follows, instantiating the variables in the

order $Z_1, Z_2, \ldots, Z_n$. For each node r, we have $A_r = \{Z_1, \ldots, Z_{k-1}\}$ for some k. We define Y_r to be Z_k, and choose $\succ_r$ to be some strict total order extending $\succ_{a_r}^{Z_k}$, which is possible since $\succ_{a_r}^{Z_k}$ is acyclic, by local consistency. Let us call this cs-tree $\sigma[\Gamma]$. Proposition 14 of Section 4.3 immediately implies that this cs-tree satisfies Γ:

Lemma 7 *Let Γ be a locally consistent and fully acyclic cp-theory. Then the cs-tree $\sigma[\Gamma]$ satisfies Γ.*

Lemma 7 implies that a locally consistent fully acyclic cp-theory Γ is consistent, since $\succ_{\sigma[\Gamma]}$ satisfies Γ. As observed earlier, local consistency is a necessary condition for consistency (Proposition 9, Section 4.1). We therefore have that local consistency and consistency are equivalent for fully acyclic cp-theories:

Theorem 3 *Let Γ be a fully acyclic cp-theory. Then Γ is consistent if and only if Γ is locally consistent.*

5.2 An upper approximation for fully acyclic Γ

There is a simple way of defining a polynomial upper approximation (see Definition 10 in Section 4.4) for a fully acyclic cp-theory. It is a kind of generalised lexicographic order, and is strongly related to the 'ordering queries' and the relation $\gg$ used in the proof of Theorem 6 in (Boutilier et al., 2004a). Two outcomes are compared by comparing, using the appropriate local ordering, their value on each of the most important variables on which they differ, where importance is defined (here) according to relation $G(\Gamma)$. The section finishes with a result, Proposition 17, which shows that this polynomial upper approximation for Γ is equal to the preference relation associated with a strengthened form of Γ.

Definition 12 *Let Γ be a locally consistent and fully acyclic cp-theory. Define relation $\succ_{p(\Gamma)}$ on outcomes as follows. Let α and β be outcomes in $\underline{V}$. Define $\Delta(\alpha, \beta)$ to be the set of variables on which α and β differ, that is, the set $\{Y \in V : \alpha(Y) \neq \beta(Y)\}$. If $\alpha \neq \beta$, define $\Theta(\alpha, \beta)$ to be the set of G°-maximal elements of $\Delta(\alpha, \beta)$, where G° is the transitive closure of $G(\Gamma)$, i.e., variables $Y \in \Delta(\alpha, \beta)$ such that there exists no $Z \in \Delta(\alpha, \beta)$ such that $(Z, Y) \in G^\circ$. In other words, $\Theta(\alpha, \beta)$ is the set of variables Y on which α and β differ such that every ancestor of Y agrees on α and β. For $\alpha, \beta \in \underline{V}$, we define $\alpha \succ_{p(\Gamma)} \beta$ if and only if $\alpha \neq \beta$ and $\alpha(Y) \succ_\alpha^Y \beta(Y)$ for all $Y \in \Theta(\alpha, \beta)$.*

For given outcomes α and β, determining if $\alpha \succ_{p(\Gamma)} \beta$ can clearly be done in polynomial time. This ordering is similar to a lexicographic ordering. It views $G(\Gamma)$ as expressing relative importance of variables: if $(Y, Z) \in G(\Gamma)$ then variable Y is considered as more important than Z. The idea is that $\Theta(\alpha, \beta)$ is the set of most important variables where α and β differ. α is preferred to β if α is better than β on each of these variables. A standard lexicographic order compares two outcomes α and β by considering the most important variable X

on which α and β differ, and preferring α to β if $\alpha(X)$ is preferred to $\beta(X)$. This order differs in that (i) there can be more than one best variable on which α and β differ, because G° is only a partial order; and (ii) the local preference of $\alpha(X)$ over $\beta(X)$ can be partial, and conditional on more important variables.

With Γ as in Example A (Section 2.2.2), consider outcomes $\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}}$ and $\mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}$. We have $\Delta(\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}}, \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}) = \{X_2, X_3\}$, and $\Theta(\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}}, \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}) = \{X_2\}$, because variable X_2 is more important than X_3 according to $G(\Gamma)$. Also, $\mathbf{o} \succ^{X_2}_{\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}}} \overline{\mathbf{o}}$ holds because of φ_2, and so, $\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}} \succ_{p(\Gamma)} \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}$. In fact, the relation $\succ_{p(\Gamma)}$ is in this case a total order, which extends $>_\Gamma$ with the additional preference $\mathbf{n}\,\mathbf{o}\,\overline{\mathbf{p}} \succ_{p(\Gamma)} \mathbf{n}\,\overline{\mathbf{o}}\,\mathbf{p}$.

The following lemma shows that relation $\succ_{p(\Gamma)}$ is equal to the intersection of a particular set of cs-tree orderings, where the ordering of variables in the cs-trees involved is always compatible with $G(\Gamma)$.

Lemma 8 *Let Γ be a locally consistent and fully acyclic cp-theory. Let $\mathcal{S}$ be the set of cs-trees σ satisfying the two conditions:*

(a) for any pair $(Y, Z) \in G(\Gamma)$ and any outcome α, variable Y appears before Z on the path to α (i.e., for any $\varphi \in \Gamma$, variables U_φ appear before variable X_φ, which appears before variables W_φ on any path to any outcome);

(b) for any body node r in σ, $\succ_r \supseteq \succ^{Y_r}_u$, where $u = a_r(U_{Y_r})$, and U_{Y_r} is the set of the parents of Y_r in $H(\Gamma)$ (see Section 2.2.3). (Note that $U_{Y_r} \subseteq A_r$ by (a).)

For outcomes α and β, $\alpha \succ_{p(\Gamma)} \beta$ if and only if for all cs-trees σ in $\mathcal{S}$, we have $\alpha >_\sigma \beta$. In other words, $\succ_{p(\Gamma)} = \gg^{\mathcal{S}}$, using the notation of Proposition 15, Section 4.4.

This lemma and Proposition 15 (Section 4.4) immediately imply the following result, showing that $\succ_{p(\Gamma)}$ is a polynomial upper approximation of the preference relation. (This result can also be proved more directly.)

Theorem 4 *Let Γ be a locally consistent and fully acyclic cp-theory. Then $\succ_{p(\Gamma)}$ is a strict partial order containing $>_\Gamma$, so is an upper approximation of the preference relation $>_\Gamma$.*

As shown by Theorem 4, $\succ_{p(\Gamma)}$ extends $>_\Gamma$. In fact, it turns out that if one strengthens the preference statements in Γ, by expanding the sets W_φ, to form a cp-theory $\overline{\Gamma}$, we obtain equality between the relations: $\succ_{p(\Gamma)} = \succ_{p(\overline{\Gamma})} = >_{\overline{\Gamma}}$. This result is used in the footnote in Section 3.5.

Proposition 17 *Let $\Gamma \subseteq \mathcal{L}$ be a cp-theory. For $\varphi = (u : x > x'\,[W]) \in \Gamma$ let $\overline{\varphi}$ be $(u : x > x'\,[W'])$ where W' is the set of descendants of X in $G(\Gamma)$ (i.e., the set of variables Y such that (X, Y) is in the transitive closure of $G(\Gamma)$). Define $\overline{\Gamma} = \{\overline{\varphi} : \varphi \in \Gamma\}$. If Γ is locally consistent and fully acyclic then $\succ_{p(\Gamma)} = \succ_{p(\overline{\Gamma})} = >_{\overline{\Gamma}} \supseteq >_\Gamma$.*

6 Consistency and polynomial upper bound of preference relation: more general case

This section considers the problems of determining consistency, and generating a polynomial upper approximation for more general kinds of cp-theories than the fully acyclic ones considered in Section 5.

We first consider two conditions on cp-theories that are sufficient conditions for the cp-theory to be consistent. They are weaker forms of acyclicity, though stronger than conditional acyclicity: strong conditional acyclicity in Section 6.1, and context-uniform conditional acyclicity (cuc-acyclicity) in Section 6.2. (The latter is very similar to the notion of "conditional acyclicity" in (Brafman et al., 2006).) An important aspect of these conditions is that the complexity of determining them is only coNP-complete (see Proposition 24), which is much less than the complexity (PSPACE-complete) of determining consistency for general cp-theories. For cuc-acyclic cp-theories we define a way of generating an upper approximation in Section 6.3. Section 6.4 shows how to generate a total order satisfying the cp-theory. Theorem 6 in Section 6.5 summarises the results on the strength of different forms of consistency, specifically, that the following are progressively weaker conditions on a cp-theory: being locally consistent and fully acyclic, cuc-acyclicity, strong conditional acyclicity, conditional acyclicity, consistency, local consistency. Section 6.5 also discusses how to use the results to confirm that a cp-theory is consistent.

6.1 Strong conditional acyclicity and strong satisfaction

We consider a condition, called *strong satisfaction*, which is sufficient for a cs-tree to satisfy a cp-theory (see Definition 13 below). This is expressed as a pair of conditions on each node r of the cs-tree. The first condition is on the choice of the new variable Y to be branched on, given the assignment a to variables made above that node: if this condition is satisfied, we say that Y is *strongly a-undominated*. The second condition is that the node value ordering extends the appropriate local ordering. Proposition 18 gives an equivalent definition for a cs-tree to strongly satisfy a cp-theory, which leads to Proposition 19, which states that strong satisfaction implies satisfaction. The definition of strong satisfaction suggests the definition of a stronger form of conditional acyclicity, Definition 14, that the cp-theory is locally consistent, and that given any assignment a to any proper subset of the set of variables, there exists a strongly a-undominated variable. This easily implies that if a cp-theory is strongly conditionally acyclic then one can iteratively construct, starting with the root node, a cs-tree strongly satisfying Γ, hence implying the consistency of Γ (see Proposition 21). In addition, a result, Proposition 20, is given that is useful for helping prove that a particular cp-theory is strongly conditionally acyclic.

We are interested in cs-trees whose associated orders satisfy cp-theory Γ. For this we need to make sure that for any path and any relevant $\varphi \in \Gamma$, X_φ appears before all the variables in W_φ, as X_φ is a more important variable

(see Proposition 13, in Section 4.3). Furthermore, we require here that the conditioning variables U_φ to all appear before X_φ, so that we know the values of relevant parents of X_φ before we can decide which values of X_φ are preferred (cf. Proposition 14).

Definition 13 *Let Γ be a cp-theory over variables V. Let A be a subset of V, and let $a \in \underline{A}$ be an assignment to the variables A.*

- *We say that $Y \in V - A$ is strongly a-undominated (with respect to Γ) if for all $\varphi \in \Gamma$ such that u_φ is compatible with a, (i) if $X_\varphi = Y$ then $U_\varphi \subseteq A$; (ii) if $W_\varphi \ni Y$ then $U_\varphi \cup \{X_\varphi\} \subseteq A$.*
- *A body tuple $\langle A, a, Y, \succ \rangle$ is said to strongly satisfy Γ if (I) Y is strongly a-undominated and (II) the ordering $\succ$ on $\underline{Y}$ extends $\succ_a^Y$ (i.e., if $y \succ_a^Y y'$ then $y \succ y'$).*
- *A body node is said to strongly satisfy Γ if its associated body tuple strongly satisfies Γ;*
- *A cs-tree is said to strongly satisfy Γ if each body node in the cs-tree strongly satisfies Γ.*

The following result is an immediately consequence of the first part of the previous definition, since any u_φ is compatible with $\top$, the assignment to the empty set of variables.

Lemma 9 *Let Γ be a cp-theory over variables V, and let Y be a variable in V. Then variable Y is strongly $\top$-undominated if and only if for all $\varphi \in \Gamma$, (a) $Y \notin W_\varphi$, and (b) if $X_\varphi = Y$ then $U_\varphi = \emptyset$.*

In other words, Y is strongly $\top$-undominated if and only if Y is never less important than any other variable, and any preferences regarding the values of Y are unconditional. This is also if and only if Y is undominated with respect to $G(\Gamma)$ (see Section 2.2.3), i.e., there does not exist $Z \in V$ with $(Z, Y) \in G(\Gamma)$.

If a cs-tree strongly satisfies Γ, the conditions ensure that if $\varphi \in \Gamma$ and outcome α is such that $\alpha \models u_\varphi$ then on the path from the root to the leaf node $\langle V, \alpha \rangle$, variables U_φ appear before X_φ which appears before variables W_φ; this leads to the following result expressing an equivalent form for a cs-tree to strongly satisfy a cp-theory.

Proposition 18 *Let Γ be a cp-theory, and let σ be a cs-tree. Then σ strongly satisfies Γ if and only if the following pair of conditions hold:*

(1) For any $\varphi \in \Gamma$ and any outcome α such that $\alpha \models u_\varphi$: on the path from the root to α, each element of U_φ appears before X_φ, which appears before each element of W_φ;

(2) for any body node r of σ with associated tuple $\langle A, a, Y, \succ \rangle$, relation $\succ$ extends the local ordering $\succ_a^Y$.

Proposition 18, and Proposition 14 from Section 4.3, immediately imply the following result.

Proposition 19 *If a cs-tree σ strongly satisfies cp-theory Γ, then it satisfies Γ, i.e., its associated order $>_\sigma$ satisfies Γ.*

Here we define a stronger form of conditional acyclicity of a cp-theory (as shown by Proposition 21 below).

Definition 14 (strongly conditionally acyclic cp-theory) *We say that Γ is* strongly conditionally acyclic *if it is locally consistent and for all proper subsets A of V, and all $a \in \underline{A}$, there exists a strongly a-undominated variable (see Definition 13).*

Below we derive a result, Proposition 20, that can make it easier to prove strong conditional acyclicity, and hence consistency. First we show, in Lemma 10, how being strongly a-undominated is the same as being undominated with respect to a particular relation, F_a on $V - A$. Let $A \subseteq V$ be a set of variables, and let $a \in \underline{A}$ be an assignment to A. Define relation F_a on V (using the notation of Section 2.2.3) to consist of all edges $U_\varphi \rightarrow X_\varphi$ and $U_\varphi \cup \{X_\varphi\} \rightarrow W_\varphi$ for all $\varphi \in \Gamma$ such that u_φ is compatible with a, i.e., F_a is the union of $(U_\varphi \rightarrow X_\varphi) \cup (U_\varphi \cup \{X_\varphi\} \rightarrow W_\varphi)$ over all $\varphi \in \Gamma$ such that u_φ is compatible with a. (Thus, $(Y, Z) \in F_a$ if and only if there exists some $\varphi \in \Gamma$ such that u_φ is compatible with a and either $Y \in U_\varphi$ and $Z = X_\varphi$ or $Y \in U_\varphi \cup \{X_\varphi\}$ and $Z \in W_\varphi$.)

Lemma 10 *Let $A \subseteq V$, and let $a \in \underline{A}$, and let Y be a variable in $V - A$. Then Y is strongly a-undominated if and only if Y is not F_a-dominated by any element of $V - A$, i.e., there does not exist $Z \in V - A$ with $(Z, Y) \in F_a$.*

Proposition 20 *Let $A, B \subseteq V$ be a sets of variables with $A \subseteq B$, and let $a \in \underline{A}$ be an assignment to A, and suppose that $b \in \underline{B}$ extends a. Then*

(i) $F_b \subseteq F_a$, i.e., if $(Y, Z) \in F_b$ then $(Y, Z) \in F_a$;

(ii) if $Y \in V - B$ is strongly a-undominated then Y is strongly b-undominated;

(iii) if F_a restricted to $V - A$ is acyclic then there exists a strongly b-undominated variable in $V - B$.

Example 8 *Consider again the cp-theory Γ from Example 3 in Section 4.1, consisting of the following preference statements: $\varphi_1 = \top : x_1 > \overline{x}_1\,[\{X_2, X_3\}]$, $\varphi_2 = x_1 : x_3 > \overline{x}_3\,[\{X_2\}]$; $\varphi_3 = x_1 : x_2 > \overline{x}_2\,[\emptyset]$. $\varphi_4 = \overline{x}_1 : x_2 > \overline{x}_2\,[\{X_3\}]$, and $\varphi_5 = \overline{x}_1 : \overline{x}_3 > x_3\,[\emptyset]$. By Lemma 9, X_1 is strongly $\top$-undominated, so, by Proposition 20(ii), X_1 is strongly b-undominated for any assignment b to set of variables B such that $B \not\ni X_1$. F_{x_1} restricted to $\{X_2, X_3\}$ consists just of the edge $X_3 \rightarrow X_2$ (because of φ_2) and is hence acyclic. Then, by Proposition 20(iii), there exists a strongly b-undominated variable for any b extending x_1.*

Similarly, $F_{\overline{x}_1}$ restricted to $\{X_2, X_3\}$ is acyclic, so by Proposition 20(iii), there exists a strongly b-undominated variable for any b extending $\overline{x}_1$. This implies that, for any assignment b to a proper subset of V, there exists a strongly b-undominated variable, and hence Γ is strongly conditionally acyclic. A similar argument works for Example B in Section 3.2.

This example illustrates that strong conditional acyclicity, and hence consistency, of a cp-theory Γ can sometimes be easily shown, if Γ is close to being acyclic.

The following result follows easily from the definitions.

Lemma 11 *Let Γ be a strongly conditionally acyclic cp-theory. Given any proper subset A of V and $a \in \underline{V}$, there exists Y and total order $\succ$ on $\underline{Y}$ such that $\langle A, a, Y, \succ \rangle$ is a body tuple strongly satisfying Γ.*

Proof: Since Γ is strongly conditionally acyclic, there exists a strongly a-undominated variable, Y. Choose any outcome α extending a, and pick some strict total order $\succ$ extending $\succ_\alpha^Y$. (This is possible since, by local consistency, $\succ_\alpha^Y$ is a strict partial order.) Then $\succ$ extends $\succ_a^Y$, so $\langle A, a, Y, \succ \rangle$ strongly satisfies Γ. □

This means that it is easy to construct a cs-tree satisfying a strongly conditionally acyclic Γ: we start by picking a root node strongly satisfying Γ, and we proceed inductively, choosing children strongly satisfying Γ for each node already chosen. Lemma 11 ensures that the cs-tree generated will strongly satisfy Γ, and hence satisfies Γ, by Proposition 19. This implies the following result:

Proposition 21 *Suppose Γ is a strongly conditionally acyclic cp-theory. Then Γ is conditionally acyclic and hence consistent.*

The example below shows that the converse fails.

Example 9 *Let $V = \{X_1, X_2\}$. Let Γ be the pair of statements $x_1 : x_2 > \overline{x}_2\,[\emptyset]$ and $x_2 : x_1 > \overline{x}_1\,[\emptyset]$. Γ^* is just equal to $\{(x_1x_2, x_1\overline{x}_2), (x_1x_2, \overline{x}_1x_2)\}$, and so is acyclic (with $>_\Gamma$ equal to Γ^*) and hence, by Theorem 1(ii), Γ is consistent. Furthermore, Γ is conditionally acyclic: we can define a cs-tree σ satisfying Γ as follows. σ includes two other body nodes apart from the root node. The root node is equal to $\langle \emptyset, \top, X_1, x_1 > \overline{x}_1 \rangle$. The first body node is $\langle \{X_1\}, x_1, X_2, x_2 > \overline{x}_2 \rangle$, the second is $\langle \{X_1\}, \overline{x}_1, X_2, x_2 > \overline{x}_2 \rangle$. The associated ordering $>_\sigma$ on outcomes is the transitive closure of: $x_1x_2 >_\sigma x_1\overline{x}_2 >_\sigma \overline{x}_1x_2 >_\sigma \overline{x}_1\overline{x}_2$, so, $>_\sigma$ contains Γ^*.*

However, Γ is not strongly conditionally acyclic. By Lemma 9, X_1 is not strongly $\top$-undominated, because of the preference statement $x_2 : x_1 > \overline{x}_1\,[\emptyset]$. Similarly, X_2 is not strongly $\top$-undominated, so there exists no strongly $\top$-undominated variable.

6.2 Context-uniform conditional acyclicity

We consider here *cuc-acyclicity*, a yet stronger form of conditional acyclicity of cp-theories, which requires, in particular, that the dependency graph $H(\Gamma)$ is acyclic (see Section 2.2.3). It is closely connected with the notion of conditional acyclicity for TCP-nets (Brafman et al., 2006). A major motivation for considering cuc-acyclicity is that it allows a polynomial upper approximation of the preference relation $>_\Gamma$ to be defined (see Section 6.3 below), which, as shown above in Section 4.4 and Section 4.5, can be used for ordering tasks and for constrained optimisation.

Definition 15 (cuc-acyclic cp-theory) *Let Γ be a cp-theory. Let $a \in \underline{A}$ be an assignment to a set of variables A. Define directed graph $J_a(\Gamma)$ on the set of variables V, using the notation from Section 2.2.3, to consist of the set $U_\varphi \to \{X_\varphi\} \cup W_\varphi$ of edges for all $\varphi \in \Gamma$, and also the set $X_\varphi \to W_\varphi$ of edges for all $\varphi \in \Gamma$ such that $U_\varphi \subseteq A$ and a extends u_φ. (In other words, $(Y, Z) \in J_a(\Gamma)$ if and only if there exists some $\varphi \in \Gamma$ such that either (i) $Y \in U_\varphi$ and $Z \in \{X_\varphi\} \cup W_\varphi$ or (ii) a extends u_φ, and $Y = X_\varphi$ and $Z \in W_\varphi$.) Define order $\rhd_a(\Gamma)$ on V (abbreviated to $\rhd_a$) to be the transitive closure of $J_a(\Gamma)$.*

We say that cp-theory Γ is context-uniformly conditionally acyclic *(abbreviated to* cuc-acyclic*) if it is locally consistent and for each outcome $\alpha \in \underline{V}$, $J_\alpha(\Gamma)$ is acyclic, i.e., $\rhd_\alpha$ is irreflexive.*

The reason for the terminology "context-uniform" is that the relations $U_\varphi \to \{X_\varphi\} \cup W_\varphi$ involving the context sets U_φ are required to hold for any $\varphi \in \Gamma$, in contrast with the relations in Section 6.1 (see e.g., the definition of F_a) which require a condition on φ. In particular, cuc-acyclicity of Γ requires that the dependency graph $H(\Gamma)$ is acyclic.

This property is very similar to the notion of "conditional acyclicity" for TCP-nets: see Definitions 8 and 9 in (Brafman et al., 2006); (the property we call "conditional acyclicity" is a much weaker condition). The methods in Section 5 of (Brafman et al., 2006) could therefore be used for checking whether certain kinds of cp-theories are cuc-acyclic.

Note that if Γ is context-uniformly conditionally acyclic then for any $A \subseteq V$ and $a \in \underline{A}$, J_a is acyclic, since $J_a \subseteq J_\alpha$ for any α extending a (i.e., such that $\alpha(A) = a$).

cuc-acyclicity is a still stronger condition than being strongly conditionally acyclic:

Proposition 22 *If cp-theory Γ is context-uniformly conditionally acyclic then it is strongly conditionally acyclic, and hence conditionally acyclic and consistent.*

Example 10 *Consider again the cp-theory Γ from Example 8 (Section 6.1) and Example 3 (Section 4.1). For any α extending x_1, $\rhd_\alpha$ is consistent with (in fact is equal to) the ordering X_1, X_3, X_2 (partly because of the preference statement φ_2 which equals $x_1 : x_3 > \overline{x}_3\,[\{X_2\}]$). For any α extending $\overline{x}_1$, $\rhd_\alpha$ is equal to*

the ordering X_1, X_2, X_3. Thus, for any α, $\rhd_\alpha$ is irreflexive, so Γ is cuc-acyclic. Similarly, for Example B (Section 3.2), for α extending x_1, $\rhd_\alpha$ equals the the ordering X_1, X_3, X_2, X_4, and for α extending $\overline{x}_1$, $\rhd_\alpha$ equals the the ordering X_1, X_2, X_3, X_4, and so Γ is cuc-acyclic for Example B as well.

As the following example illustrates, context-uniform conditional acyclicity is a strictly stronger condition than strong conditional acyclicity.

Example 11 *Let cp-theory Γ consist of the following statements: $\top : x_1 > \overline{x}_1 \, [\emptyset]$, $x_1x_3 : x_2 > \overline{x}_2 \, [\emptyset]$, and $\overline{x}_1x_2 : x_3 > \overline{x}_3 \, [\emptyset]$. $H(\Gamma)$ is not acyclic, since X_2 is a parent of X_3, and vice versa. This implies that, for any outcome α, $J_\alpha(\Gamma)$ is not acyclic; hence, Γ is not cuc-acyclic. On the other hand, Γ is strongly conditionally acyclic. Consider any assignment $a \in \underline{A}$, where A is a proper subset of V. To show strong conditional acyclicity we need to show that there exists some $Y \in V - A$ which is strongly a-undominated. If A does not contain X_1, then X_1 is strongly a-undominated. If a equals x_1, or x_1x_3 or $x_1\overline{x}_3$ then X_2 is strongly a-undominated. If a equals $\overline{x}_1$, or x_1x_2 or $x_1\overline{x}_2$ then X_3 is strongly a-undominated. This covers all cases.*

6.3 Generating a polynomial upper approximation of the preference relation

In this section, we develop an approach for generating an upper approximation for the preference relation $>_\Gamma$. As explained in Sections 4.4 and 4.5, this can be used for constrained optimisation and ordering queries. This particular upper approximation, although not requiring full acyclicity, still requires a strong condition on the cp-theory, that it be cuc-acyclic (see Definition 15, Section 6.2).

We define a relation $\gg_\Gamma$ on outcomes associated with a cuc-acyclic Γ. This is a modified form of Definition 12 in Section 5.2, which is for the fully acyclic case. We will show (Theorem 5) that $\gg_\Gamma$ is an upper approximation of the preference relation $>_\Gamma$.

To compare two outcomes with relation $\gg_\Gamma$, we first consider the set $\Delta(\alpha, \beta)$ of variables on which they differ. Among this set we eliminate any of the variables which are dominated by any other with respect to relation $\rhd_\alpha$ (see Definition 15), to obtain the set of variables $\Theta'(\alpha, \beta)$. Then $\alpha \gg_\Gamma \beta$ is defined to hold if and only if for all $X \in \Theta'(\alpha, \beta)$, $\alpha(X)$ is better than $\beta(X)$ according to the local ordering $\succ_\alpha^X$ (see Definition 4).

Let A be a subset of V, and let $a \in \underline{A}$ be an assignment to the variables A, and let B be a subset of V. We say that $Y \in B$ is *$\rhd_a$-undominated in B* if Y is undominated in B with respect to $\rhd_a$, i.e., there does not exist $Z \in B$ with $Z \rhd_a Y$.

Definition 16 (upper approximation $\gg_\Gamma$) *Consider fixed cuc-acyclic Γ. For $\alpha, \beta \in \underline{V}$, let $\Delta(\alpha, \beta)$ be the set of variables where α and β differ, i.e., $\{X \in V : \alpha(X) \neq \beta(X)\}$. Let $\Theta'(\alpha, \beta)$ be the $\rhd_\alpha$-undominated variables in $\Delta(\alpha, \beta)$*

(note that the definition of cuc-acyclicity ensures that $\Theta'(\alpha,\beta)$ is non-empty given that $\alpha \neq \beta$). The binary relation $\gg_\Gamma$ on outcomes is defined by: $\alpha \gg_\Gamma \beta$ if and only if for all $X \in \Theta'(\alpha,\beta)$, $\alpha(X) \succ_\alpha^X \beta(X)$.

We will prove the following result in the next section.

Theorem 5 *Let Γ be a cuc-acyclic cp-theory. Then $\gg_\Gamma$ is a strict partial order containing $>_\Gamma$ and so is an upper approximation for the preference relation $>_\Gamma$.*

Example 12 *Continuing Example 10, let α be the outcome $\overline{x}_1 x_2 x_3$, and let β be the outcome $\overline{x}_1\overline{x}_2\overline{x}_3$. The two outcomes differ on variables X_2 and X_3 so $\Delta(\alpha,\beta) = \{X_2, X_3\}$. $\rhd_\alpha$ is the ordering X_1, X_2, X_3 (see Example 10) so $\Theta'(\alpha,\beta) = \{X_2\}$. We have $\alpha(X_2) \succ_\alpha^{X_2} \beta(X_2)$, i.e., $x_2 \succ_\alpha^{X_2} \overline{x}_2$, because of the preference statement $\varphi_4 = \overline{x}_1 : x_2 > \overline{x}_2\,[\{X_3\}]$. Hence, $\alpha \gg_\Gamma \beta$.*

Dominance testing with $\gg_\Gamma$, i.e., determining if $\alpha \gg_\Gamma \beta$ or not, can be done in polynomial time (and is often very easy), and so $\gg_\Gamma$ is a polynomial upper approximation. If Γ is fully acyclic then $\gg_\Gamma$ can be compared with relation $\succ_{p(\Gamma)}$ defined in Definition 12 in Section 6.3. Let α and β be outcomes. We have $J_\alpha(\Gamma) \subseteq G(\Gamma)$, and so $\rhd_\alpha \subseteq G^\circ$. Hence, $\Theta'(\alpha,\beta) \supseteq \Theta(\alpha,\beta)$. Therefore, $\alpha \gg_\Gamma \beta$ implies $\alpha \succ_{p(\Gamma)} \beta$, so that $\gg_\Gamma$ is a closer approximation of $>_\Gamma$ than $\succ_{p(\Gamma)}$.

Proving Theorem 5

The way we shall prove Theorem 5 is to show, using Lemma 14 and Lemma 15 below, that $\gg_\Gamma$ is the intersection of total orders $>_\sigma$ over a particular set of cs-trees, those which *cu-satisfy* Γ (see Definition 17 below). This property ensures that the cs-tree satisfies Γ. The theorem then follows using Proposition 15 of Section 4.4. (A more direct proof is also possible.)

Definition 17 (cu-satisfaction) *Let $\Gamma \subseteq \mathcal{L}$ be cuc-acyclic. A body node with associated tuple $\langle A, a, Y, \succ\rangle$ is said to context-uniformly satisfy Γ (abbreviated to* cu-satisfy Γ*) if (i) Y is $\rhd_a$-undominated in $V - A$ (i.e., there does not exist $Z \in V - A$ with $Z \rhd_a Y$), and (ii) the total ordering $\succ$ on $\underline{Y}$ extends $\succ_a^Y$ (i.e., if $y \succ_a^Y y'$ then $y \succ y'$). A cs-tree is said to* cu-satisfy Γ *if each body node in the cs-tree cu-satisfies Γ.*

If Y is $\rhd_a$-undominated in $V - A$ then it can be seen that Y is strongly a-undominated in $V - A$ (see Definition 13), which immediately implies the following result, using Proposition 19, Section 6.1.

Lemma 12 *If a cs-tree cu-satisfies Γ then it strongly satisfies Γ, and hence satisfies Γ.*

If Γ is cuc-acyclic (see Definition 15), then for any $A \subseteq V$ and $a \in \underline{A}$, we can define a body node cu-satisfying Γ by choosing any Y which is $\rhd_a$-undominated in $V - A$ ($\rhd_a$ is acyclic, since $\rhd_\alpha$ is acyclic for any $\alpha \models a$, ensuring we can

pick such a Y) and choosing any total order $\succ$ on $\underline{Y}$ extending $\succ_a^Y$ (which is possible since $\succ_a^Y$ is a partial order by local consistency). This means that we can generate a cs-tree cu-satisfying Γ by choosing a root node and iteratively generating children of nodes already created. This proves the following lemma:

Lemma 13 *If Γ is cuc-acyclic then there exists a cs-tree cu-satisfying Γ.*

The following result shows that $\gg_\Gamma$ is contained in $>_\sigma$ for any cs-tree σ cu-satisfying Γ.

Lemma 14 *Let Γ be a cuc-acyclic cp-theory, and let σ be any cs-tree that cu-satisfies Γ. Then $>_\sigma$ contains $\gg_\Gamma$.*

The next lemma supplies the final piece we need to prove Theorem 5.

Lemma 15 *Let Γ be a cuc-acyclic cp-theory, and suppose outcomes $\alpha, \beta \in \underline{V}$ are such that it is not the case that $\alpha \gg_\Gamma \beta$. Then there exists a cs-tree σ cu-satisfying Γ such that it is not the case that $\alpha >_\sigma \beta$.*

Putting the parts together, we have the following result, which immediately entails Theorem 5.

Proposition 23 *Let Γ be a cuc-acyclic cp-theory. Then $\gg_\Gamma$ is the intersection of $>_\sigma$ over all cs-trees σ cu-satisfying Γ. Furthermore, $\gg_\Gamma$ is a strict partial order containing $>_\Gamma$ and so is an upper approximation for the preference relation $>_\Gamma$.*

Proof: By Lemma 14 and Lemma 15, $\alpha \gg_\Gamma \beta$ if and only if $\alpha >_\sigma \beta$ for all cs-trees σ cu-satisfying Γ. This implies that $\gg_\Gamma$ is the intersection of $>_\sigma$ over all cs-trees σ cu-satisfying Γ. If σ cu-satisfies Γ then it satisfies Γ, by Lemma 12. Proposition 15 (Section 4.4) then implies that $\gg_\Gamma$ is a strict partial order containing $>_\Gamma$. □

6.4 Generating a total order satisfying cp-theory Γ

The procedure described above—after Lemma 11 in Section 6.1—for generating a cs-tree from strongly conditionally acyclic Γ, could usually, depending on the choices made, generate many different cs-trees satisfying Γ. It can be useful to have a way of pinning down which choice is made, and thus a way of defining (implicitly) a particular total order that satisfies Γ. We assume a listing $X_1, \ldots, X_n$ of V, and for each i, a total ordering $x_i^1, \ldots, x_i^{m_i}$ (where $m_i = |\underline{X_i}|$) of the values of X_i. We use these to define a particular cs-tree $\sigma(\Gamma)$ which satisfies Γ. This is constructed from the root down. Whenever we have a set of choices of a-undominated variable at node r we choose Y_r to be X_i with minimal i (among the choices). Similarly we define $\succ_r$ by generating the values of X_i from the best to the worst, at each point choosing a $\succ_a^{X_i}$-maximal value

amongst the remaining values, where ties are broken by choosing value x_i^j with largest j.

Suppose we want to be able to generate outcomes in an order compatible with the preferences Γ (i.e., compatible with $>_\Gamma$). If Γ is conditionally acyclic then $>_{\sigma(\Gamma)}$ is such an order. This order was defined implicitly: we do not have to explicitly construct cs-tree $\sigma(\Gamma)$ to use it. In particular, we can generate in polynomial time the best K outcomes according to $>_{\sigma(\Gamma)}$, by generating just the first K leaf nodes of cs-tree $\sigma(\Gamma)$. Also, given any two different outcomes α and β, we can efficiently determine which is better according to this order $>_{\sigma(\Gamma)}$, by constructing just the nodes that are above both leaf node $\langle V, \alpha\rangle$ and leaf node $\langle V, \beta\rangle$.

6.5 Summary and discussion on forms of consistency

The following result sums up the relationships between the different conditions of cp-theories that we have explored.

Theorem 6 *Let Γ be a cp-theory. Recall that Γ is*

- consistent *if there exists a strict total order satisfying Γ (Definition 2, Section 2.3);*
- locally consistent *if $\succ_\alpha^X$ is irreflexive for all variables X and outcomes α (Definition 4, Section 4.1);*
- fully acyclic *if $G(\Gamma)$ is acyclic (see Section 2.2.3);*
- conditionally acyclic *if there exists a cs-tree satisfying Γ (Definition 9, Section 4.3);*
- strongly conditionally acyclic *if it is locally consistent and for all $A \subseteq V$ and $a \in \underline{A}$, there exists a strongly a-undominated variable (Definition 14, Section 6.1);*
- context-uniformly conditionally acyclic (cuc-acyclic) *if it is locally consistent and for each outcome $\alpha \in \underline{V}$, $J_\alpha(\Gamma)$ is acyclic (Definition 15, Section 6.2).*

Then, Γ is locally consistent and fully acyclic

$\Rightarrow$ *Γ is context-uniformly conditionally acyclic*

$\Rightarrow$ *Γ is strongly conditionally acyclic*

$\Rightarrow$ *Γ is conditionally acyclic*

$\Rightarrow$ *Γ is consistent*

$\Rightarrow$ *Γ is locally consistent.*

Moreover, none of the implications are equivalences.

Proof: If Γ is fully acyclic, i.e., $G(\Gamma)$ is acyclic, then $J_\alpha(\Gamma)$ is acyclic for any outcome α, since $J_\alpha(\Gamma) \subseteq G(\Gamma)$. This shows that if Γ is locally consistent and $G(\Gamma)$ is acyclic then Γ is context-uniformly conditionally acyclic. The other implications are from Propositions 22, 21, 12 and 9, respectively.

The last part follows from Example 10 (where the cp-theory Γ is context-uniformly conditionally acyclic but not fully acyclic), Example 11, Example 9, Example 6, and Example 2, respectively. □

Determining if Γ is strongly conditionally acyclic is a coNP-complete problem, as is determining if Γ is context-uniformly conditionally acyclic. It is straight-forward to see that these problems are in coNP. coNP-hardness can be shown using a reduction from 3-SAT, with a similar construction to that used in the proof of Theorem 2 of (Brafman et al., 2006), and for coNP-hardness of local consistency (Proposition 11).

Proposition 24 *The problem of determining if cp-theory Γ is strongly conditionally acyclic is* coNP*-complete, as is the problem of determining if Γ is cuc-acyclic.*

Determining consistency of a cp-theory is PSPACE-complete (Goldsmith, Lang, Truszczyński, & Wilson, 2005; Goldsmith et al., 2008) We thus have both a necessary condition for consistency (local consistency) and a sufficient condition (strong conditional acyclicity) of much lower complexity.

One might adapt the methods of Section 5 of (Brafman et al., 2006) to check the context-uniformly conditionally acyclicity–and hence to show the consistency—of certain forms of cp-theory Γ. It may also often be easy to confirm consistency of a cp-theory Γ in a somewhat *ad hoc* manner, assuming that Γ is locally consistent (which will often be easy to confirm). We can define a simple set of rules that determine which variable gets picked at each point, in generating a cs-tree σ satisfying Γ. For example, suppose $V = \{X_1, \ldots, X_8\}$; consider the following rules for the variable ordering, where e.g., $X_1 < X_2$ means that X_1 must always be picked earlier than X_2:

- $X_1 < X_2 < \{X_3, X_4, X_5\} < X_6 < \{X_7, X_8\}$;
- if x_1 then $X_3 < X_4 < X_5$;
- if $x_1' x_2$ then $X_5 < X_4 < X_3$; else $X_5 < X_3 < X_4$;
- if $x_4 x_6'$ then $X_7 < X_8$, else $X_8 < X_7$.

Given tuple $a \in A$, these rules determine a minimal variable in $V - A$, say Y_a; for example, if a is the assignment $x_1' x_2'$ then $Y_a = X_5$; we can also define the value ordering $\succ$ at any node in a simple way. This defines a compact representation function g with associated cs-tree σ_g (see Section 4.2). It is straight-forward to check if σ_g strongly satisfies Γ (that is, if Y_a is always a-undominated). If so, then this proves that Γ is conditionally acyclic and hence consistent; we can also use σ_g to totally order a set of outcomes.

In the next section a more formal way is developed of compactly representing the variable orderings for a search tree, and hence determining consistency, and enabling one to totally order outcomes in a way that is compatible with a cp-theory.

7 Variable Ordering Networks as Compact Representations of Satisfying cs-trees

It was shown in Section 4 how complete search trees can be used for the important tasks of confirming consistency of a cp-theory Γ, and totally ordering a set of outcomes. In particular, if Γ happens to be strongly conditionally acyclic then it is easy to define a cs-tree that (strongly) satisfies Γ (see Section 6.4). However, confirming the consistency of Γ by constructing a cs-tree explicitly will usually not be feasible since the number of nodes is of the order of the number of outcomes $|\underline{V}|$. Here we describe an often much more compact representation of certain cs-trees satisfying Γ. The idea is to turn the cs-tree into a directed acyclic graph (a *decision diagram*) by merging nodes where we can take the same decisions regarding variables orderings from that point.

The key to generating a cs-tree strongly satisfying cp-theory Γ is choosing an a_r-undominated variable Y_r for each body node r, where a_r is the tuple of assignments made to instantiated variables (see Definition 13, Section 6.1); if there is such a variable for each body node, then we have shown that Γ is consistent, assuming local consistency. (Given local consistency (4, Section 4.1), choosing the node value ordering $\succ_r$ is never a problem, since we just choose any strict total order extending the local ordering $\succ_a^Y$, which is a strict partial order by local consistency.) Consider two nodes q and r with associated tuples a and a', where a and a' are both assignments to some set of variables A. Suppose that a and a' are equivalent in the following sense: for any assignment b to any subset B of the remaining variables $V - A$, variable Y is ab-undominated if and only if Y is $a'b$-undominated. This means that we can make the same choices of a-undominated variables in nodes below q, as for nodes below r, or, more neatly, we can *merge nodes q and r*. We will define a graphical structure that enables us to assert such equivalences.

In Section 7.1 we define a variable ordering network, which is a compact representation of the variable orderings in the different branches of a cs-tree. In Section 7.2, variable ordering triples are defined, which are restrictions on variable orderings in a cs-tree or variable ordering network. In Section 7.3, sufficient conditions are defined for a variable ordering network to satisfy a set of variable ordering triples, and in Section 7.4, we show how to generate a set of variable ordering triples $[\Gamma]$ from a cp-theory Γ in such a way that a compatible variable ordering network will, given local consistency, generate a cs-tree satisfying Γ. Thus, if we can construct such a variable ordering network, we have proved consistency of locally consistent Γ, and we can use the implicitly defined cs-tree to efficiently answer total ordering queries. Section 7.5 gives a

method of constructing a variable ordering network from [Γ]. Section 7.6 shows how one can easily also generate an upper approximation of the preference relation from a variable ordering network.

7.1 Variable ordering networks

This section describes variable ordering networks, which are intended as compact representations of the variable orderings in a cs-tree which (strongly) satisfies a given cp-theory. Let σ be a cs-tree and let α be an outcome. Write $\mathcal{O}_\sigma(\alpha)$ for the ordering of variables on the path to α. The idea is to develop a compact representation for this variable ordering function $\mathcal{O}_\sigma$. A variable ordering network is very similar to a cs-tree; however, we are only interested in representing the variable orderings, so we do not include any value ordering information; we use a directed acyclic graph as a potentially much more compact representation than a tree.

Variable ordering networks (VONs) are defined first (Definition 18), and a VON is defined (Definition 19) to be compatible with a cs-tree if they induce the same variable ordering for any outcome. Proposition 25 shows that variable ordering networks and cs-trees represent the same set of variable ordering functions.

Definition 18 (Variable ordering network (VON)) *A* variable ordering network *over variables V is defined to be a directed acyclic graph with two distinguished nodes, a root node r^* and a sink node r_*, where nodes and edges have associated labels as defined below.*

Each directed edge e from node r to node r' is associated with a variable Y_e and a value y_e of Y_e (corresponding to the assignment $Y_e = y_e$). We say that r' is a child of r, and that r is a parent of r'.

Every node except the root node has a parent, and every node except the sink node has a child.

Each node r has the following associated labels:

(a) a set of variables $A_r \subseteq V$;

(b) an assignment a_r to variables A_r (corresponding to the assignments in one of the paths to that node);

(c) with the exception of the sink node: a variable $Y_r \in V - A_r$ (the next variable to be instantiated).

For node r which is not the sink node we define its associated tuple to be $\langle A_r, a_r, Y_r \rangle$. It has $|\underline{Y_r}|$ children, so has $|\underline{Y_r}|$ edges coming from it. Each such edge e has associated variable $Y_e = Y_r$ and a different associated value y_e.

If e goes from node r to r' then $A_{r'} = A_r \cup \{Y_r\}$.

For any node r' which is not equal to the root, there exists some parent node r of r' such that $a_{r'}$ is equal to the tuple formed by extending a_r with the assignment $Y_e = y_e$, where e is the edge from node r to r'.

For root node r^*, we define A_{r^*} to be the empty set, and hence we have a_{r^*} equals $\top$, the assignment to the empty set.

The sink node r_* has associated tuple $\langle A_{r_*}, \alpha \rangle$, where α is some outcome and $A_{r_*} = V$.

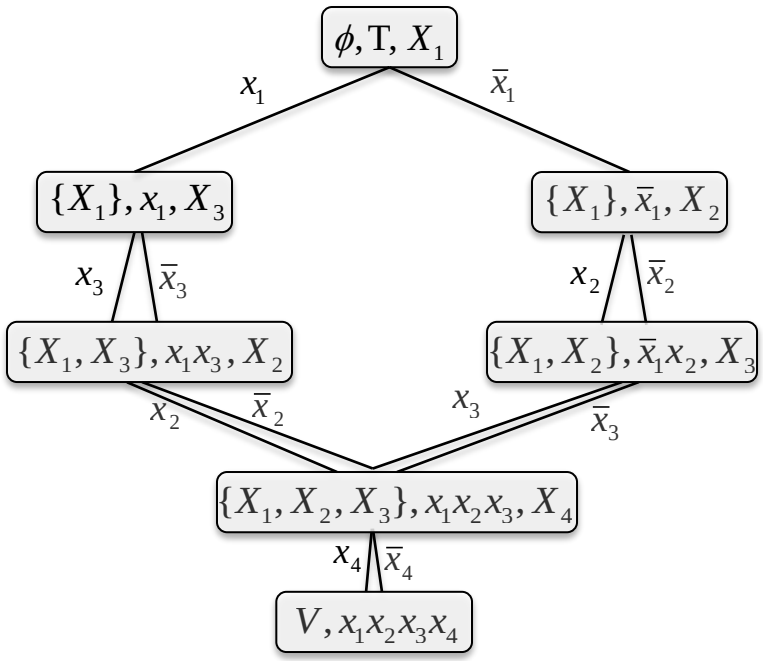

Figure 3: A variable ordering network τ. The root node is at the top, and the sink node is at the bottom. The set of paths from the root to the sink are in one-to-one correspondence with the set of outcomes.

Example 13 *Figure 7.1 illustrates a variable ordering network over the variables $V = \{X_1, X_2, X_3, X_4\}$ in Example B in Section 3.2. The top node is the root node r^* with associated tuple $\langle \emptyset, \top, X_1 \rangle$. The bottom node is the sink node r_* with associated pair $\langle V, x_1x_2x_3x_4 \rangle$. Note that, in contrast with cs-trees, some nodes have more than one parent.*

Let τ be a variable ordering network. For each outcome α, we can define a path from the root node to the sink node by following at each node r the edge corresponding to assignment $Y_r = \alpha(Y_r)$. In particular, each outcome generates an ordering of the variables V. Conversely, each (directed) path from the root node to the sink node corresponds to an outcome. The set of paths from the root node to the sink node are in one-to-one correspondence with the set of outcomes.

Let τ be a variable ordering network over variables V. Let α be any outcome. A variable ordering network τ generates a total ordering $\mathcal{O}_\tau(\alpha)$ of the variables V, by following the assignments to variables made in α. For instance, if τ is the variable ordering network in Figure 7.1, then $\mathcal{O}_\tau(x_1x_2x_3x_4)$ equals the ordering X_1, X_3, X_2, X_4 (following the left-hand edges from the root node), and $\mathcal{O}_\tau(\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4)$ equals X_1, X_2, X_3, X_4.

A variable ordering network can be viewed as a representation for the variable orderings used in a cs-tree.

Definition 19 *cs-tree σ and VON τ are said to be compatible if for all outcomes α, the ordering of variables in τ associated with α is the same as the ordering of variables in the path to α in cs-tree σ. In order words, if and only if $\mathcal{O}_\tau = \mathcal{O}_\sigma$.*

The following result implies that variable ordering networks and cs-trees represent the same set of variable ordering functions.

Proposition 25 *Let σ be a cs-tree. There exists a variable ordering network compatible with σ. Conversely, if τ is a variable ordering network then there exists some cs-tree compatible with τ.*

7.2 Variable ordering triples

A variable ordering triple $\langle u, Y, Z\rangle$ is intended as a restriction on the orderings in a cs-tree: it means that variable Y should appear before variable Z on the path from the root to any outcome α which extends assignment u. (Variable ordering triples are similar to ci-statements in TCP-nets—see Section 3.2.) A set of variable ordering triples is used to represent the conditions on variable orderings required for a cs-tree to strongly satisfy a cp-theory—see Section 7.4 below.

Definition 20 *A variable ordering triple $\langle u, Y, Z\rangle$ is an ordered triple where Y and Z are variables and u is an assignment to a set of variables.*

Given a set of variable ordering triples $\mathcal{T}$ and a partial tuple $a \in \underline{A}$ (for some $A \subseteq V$), we define $\mathcal{T}_a$ to be the set of variable ordering triples which are still relevant given a: we consider only triples $\langle u, Y, Z\rangle$ in $\mathcal{T}$ with u compatible with a, and we restrict such triples to $V - A$.

Definition 21 *Let $\mathcal{T}$ be a set of variable ordering triples, let $a \in \underline{A}$ be an assignment to a subset of V. Define $\mathcal{T}_a$ to be the set of all triples $\langle u', Y, Z\rangle$ such that there exists triple $\langle u, Y, Z\rangle$ in $\mathcal{T}$ with $Y, Z \in V - A$, tuple $u \in \underline{U}$ is compatible with a, and $u(U - A) = u'$.*

The first lemma is used to prove Lemma 17, which is a modularity property that we use in Section 7.3.

Lemma 16 *Let A and B be disjoint subsets of V, and let $a \in \underline{A}$ and $b \in \underline{B}$. Let t be an assignment to some set of variables $T \subseteq V - (A \cup B)$, and let Y and Z be variables in $V - (A \cup B)$. Let $\mathcal{T}$ be a set of variable ordering triples. Then $\langle t, Z, Y \rangle \in \mathcal{T}_{ab}$ if and only if there exists assignment u to some set of variables $U \subseteq V - A$ such that (i) $U - B = T$, $u(T) = t$, and u is compatible with b, and (ii) $\langle u, Z, Y \rangle$ is in $\mathcal{T}_a$.*

Lemma 17 *Let $\mathcal{T}$ be a set of variable ordering triples. Let A and B be disjoint subsets of V, and let $a, a' \in \underline{A}$ and $b \in \underline{B}$. If $\mathcal{T}_a = \mathcal{T}_{a'}$ then $\mathcal{T}_{ab} = \mathcal{T}_{a'b}$.*

Proof: This follows immediately from the previous lemma. □

7.3 Sufficient conditions for a variable ordering network to satisfy a set of ordering triples

In this section, a sufficient condition (see Definition 23 and Proposition 26) is given for a VON to satisfy the variable orderings conditions stipulated by a set of triples. A result, Proposition 27, is also included which will imply a limit on the size of the VONs required to satisfy a cp-theory.

A variable ordering network respects a variable ordering triple if the associated restriction on variable orderings is respected.

Definition 22 *A VON τ respects a variable ordering triple $\langle u, Y, Z \rangle$ if Y appears before Z in $\mathcal{O}_\tau(\alpha)$ for all outcomes α extending u.*

We define a condition which is sufficient (see Proposition 26) for a variable ordering network to respect a set of variable ordering triples; this condition can be enforced more easily.

Definition 23 *Let us say that VON τ is strongly compatible with set of variable ordering triples $\mathcal{T}$ if the following pair of properties holds:*

(i) for any node r of τ, if $\langle u, Y, Y_r \rangle$ is in $\mathcal{T}$ and u is compatible with a_r then $Y \in A_r$.

(ii) Merging: suppose that r is a child of node r', with the edge between them labelled with the assignment $Y_{r'} = y$. Extend assignment $a_{r'}$ with assignment $Y_{r'} = y$ to form assignment a to variables A_r. We have $\mathcal{T}_a = \mathcal{T}_{a_r}$.

The following lemma is proved using Lemma 17, and is used to prove Proposition 26, showing that a variable ordering network τ being strongly compatible with a set of ordering triples $\mathcal{T}$ implies that τ respects $\mathcal{T}$.

Lemma 18 *Suppose that τ is strongly compatible with set of triples $\mathcal{T}$ and let r be any node of τ. Consider any path from the root to r with associated assignment a. Then $\mathcal{T}_a = \mathcal{T}_{a_r}$.*

Proposition 26 *If variable ordering network τ is strongly compatible with set of ordering triples $\mathcal{T}$ then τ respects $\mathcal{T}$.*

We finish this section with a result which limits the size of the variable ordering networks that we will need to use for a cp-theory Γ (see Section 7.5).

For set $\mathcal{T}$ of ordering triples, define $Q_{\mathcal{T}}(A)$ to consist of all variables $X \in A$ such that there exists some $\langle u, Y, Z\rangle$ in $\mathcal{T}$ with $Y, Z \in V - A$ and $U \ni X$, where u is an assignment to variables U. $Q_{\mathcal{T}}(A)$ can be considered as the set of variables in A which are relevant for determining variable orderings outside of A.

Proposition 27 *Let $\mathcal{T}$ be a set of variable ordering triples on variables V, and let A be a subset of V. Suppose that a and a' agree on $Q_{\mathcal{T}}(A)$. Then $\mathcal{T}_a = \mathcal{T}_{a'}$.*

7.4 cp-theories and sets of variable ordering triples

For a cp-theory Γ, we generate a set of variable ordering triples $[\Gamma]$ corresponding to Γ. We will show (Theorem 7) that if Γ is locally consistent and we can construct a variable ordering network which respects $[\Gamma]$ then Γ is consistent. This therefore gives an approach for showing consistency of cp-theories. A method for constructing $[\Gamma]$ is given below in Section 7.5.

The idea is that we want to enforce the following property on a variable ordering network: for any $\varphi \in \Gamma$, and for any outcome α extending u_φ, variables U_φ appear before X_φ, and X_φ appears before variables W_φ on the path to α (cf Proposition 18(1), Section 6.1). $[\Gamma]$ encodes this ordering information.

Definition 24 *Define $[\Gamma]$ to be the set of all triples $\langle u, Y, Z\rangle$ such that there exists $\varphi \in \Gamma$ with $u_\varphi = u$ and either (i) $Y \in U_\varphi$ and $Z \in \{X_\varphi\} \cup W_\varphi$ or (ii) $Y = X_\varphi$ and $Z \in W_\varphi$.*

Example 14 *Consider Γ in Example B in Section 3.2 (or see Example 15 below). Because Γ contains the statement $\varphi = x_1 : x_2 > \overline{x}_2\,[\{X_4\}]$, (and $x_1 \in \underline{X_1}$, and $x_2, \overline{x}_2 \in \underline{X_2}$), $[\Gamma]$ contains, among others, the triples $\langle x_1, X_1, X_2\rangle$, and $\langle x_1, X_1, X_4\rangle$ (using case (i) of Definition 24), and also $\langle x_1, X_2, X_4\rangle$ (using case (ii) of Definition 24). This ties in with the fact that if σ is any cs-tree which strongly satisfies Γ then, because of φ, X_1 must come before X_2 which must come before X_4 on the path to any outcome α which extends x_1 (see Proposition 18(1)).*

Lemma 19 shows that $[\Gamma]$ determines which variables are strongly a-undominated. This is used in the proof of Lemma 20, which gives a sufficient condition using $[\Gamma]$ for a cs-tree to strongly satisfy cp-theory Γ, and which is used in the proof of Theorem 7.

Lemma 19 *For $a \in \underline{A}$ and $Y \in V - A$, Y is strongly a-undominated (with respect to Γ) if and only if there does not exist $\langle u, Z, Y\rangle$ in $[\Gamma]$ with $Z \in V - A$ and u compatible with a. This is also if and only if there does not exist any triple of the form $\langle u', Z, Y\rangle$ in $[\Gamma]_a$.*

Lemma 20 *Let Γ be locally consistent cp-theory, suppose that τ is a variable ordering network which respects $[\Gamma]$, and let σ be any cs-tree compatible with τ. Suppose also that for all body nodes $r = \langle A, a, Y, \succ \rangle$ in σ, the associated strict total ordering $\succ$ extends local ordering $\succ_a^Y$ (see Definition 4). Then σ strongly satisfies Γ.*

Theorem 7 *Let Γ be a cp-theory over variables V, and suppose there exists a variable ordering network τ which respects $[\Gamma]$. If Γ is locally consistent then it is conditionally acyclic. Therefore, Γ is consistent if and only if it is locally consistent.*

Proof: By Proposition 25 (Section 7.1) there exists some cs-tree σ compatible with τ. Let $r = \langle A, a, Y, \succ \rangle$ be any body node of σ. Local consistency of Γ implies that the local ordering $\succ_a^Y$ is a strict partial order. Redefine $\succ$ to be any strict total ordering extending $\succ_a^Y$. By Lemma 20, cs-tree σ now strongly satisfies Γ, showing that Γ is conditionally acyclic and hence consistent, by Proposition 12 (Section 4.3). The last part follows because local consistency is a necessary condition for consistency (Proposition 9, Section 4.1). □

Theorem 7, which is a kind of counterpart of Theorem 3 (Section 5.1), shows that if we can construct a variable ordering network which respects $[\Gamma]$ for locally consistent cp-theory Γ, then we have proved that Γ is consistent. In Section 7.5 we show one way of attempting to construct a variable ordering network which is strongly compatible with $[\Gamma]$ and hence respects $[\Gamma]$, by Proposition 26.

7.5 Generating a variable ordering network which respects $[\Gamma]$

In this section, we describe a method for constructing a variable ordering network $\tau(\Gamma)$ that respects $[\Gamma]$, and hence, if it succeeds, will prove that Γ is consistent if Γ is locally consistent (see Theorem 7).

Label V as $\{X_1, \ldots, X_n\}$. To construct $\tau(\Gamma)$ we first construct the (single) root node; then we iteratively construct the children of a node already constructed. As for cs-trees, for each node r we generate $|\underline{Y_r}|$ directed edges from r, each associated with some value of variable Y_r. Let a be a_r extended with assignment y to Y_r, and let $A = A_r \cup \{Y_r\}$. We check if there is any node q already constructed with $A_q = A$ and $[\Gamma]_{a_q} = [\Gamma]_a$ (this corresponds to the merging condition of Definition 23(ii)). If there is such a node q then we add a directed edge from r to q with associated value y of Y_r.

Suppose that there is no such node. If there is no strongly a-undominated variable in $V - A$ then we say that the construction of $\tau(\Gamma)$ fails, and we proceed no further; else we create a new node $\langle A, a, X_i \rangle$ (or the leaf node $\langle A, a \rangle$ if $A = V$) with X_i chosen with minimal i among the a-undominated variables in $V - A$.

If the construction for $\tau(\Gamma)$ does not fail, then we say that $\tau(\Gamma)$ exists.

Example 15 *The algorithm will be applied for Γ in Example B in Section 3.2, which consists of the following preference statements: $\top : x_1 > \overline{x}_1$ $[\{X_2, X_3, X_4\}]$;*

$x_1 : x_3 > \overline{x}_3\,[\{X_2, X_4\}]$; $x_1 : x_2 > \overline{x}_2\,[\{X_4\}]$; $\overline{x}_1 : x_2 > \overline{x}_2\,[\{X_3, X_4\}]$; $\overline{x}_1 : \overline{x}_3 > x_3\,[\{X_4\}]$; $x_3 : \overline{x}_4 > x_4\,[\emptyset]$; *and* $\overline{x}_3 : x_4 > \overline{x}_4\,[\emptyset]$. *We first generate the root node. To do this we need to choose a strongly* $\top$*-undominated variable. The only choice for this is* X_1*, since the preference statement* $\top : x_1 > \overline{x}_1\,[\{X_2, X_3, X_4\}]$ *means that* X_2*,* X_3 *and* X_4 *are not strongly* $\top$*-undominated, by Lemma 9. Thus, the root node is* $\langle \emptyset, \top, X_1 \rangle$.

We then create edges corresponding to choices x_1 *and* $\overline{x}_1$ *for* X_1*. Following the* x_1 *edge, we need to choose an* x_1*-undominated variable among* $\{X_2, X_3, X_4\}$*. The only one is* X_3*, because the preference statement* $x_1 : x_3 > \overline{x}_3\,[\{X_2, X_4\}]$ *means that* X_2 *and* X_4 *are not* x_1*-undominated (given* x_1*,* X_3 *needs to come before* X_2 *and* X_4*). Thus, we create a node* r *with tuple* $\langle \{X_1\}, x_1, X_3 \rangle$.

Following the edge from r *associated with the choice* x_3 *for* X_3 *leads to the node* q *with tuple* $\langle \{X_1, X_3\}, x_1x_3, X_2 \rangle$*, since* X_2 *is the only* x_1x_3*-undominated variable among* $\{X_2, X_4\}$*, because of the preference statement* $x_1 : x_2 > \overline{x}_2\,[\{X_4\}]$. *Following the edge from* r *associated with the choice* $\overline{x}_3$ *for* X_3*, let* a *be* a_r *extended with assignment* $\overline{x}_3$ *to* Y_r *(*$= X_3$*), and let* $A = A_r \cup \{Y_r\}$*, so that* $a = x_1\overline{x}_3$ *and* $A = \{X_1, X_3\}$*; we find that* $A = A_q$ *and* $[\Gamma]_a = [\Gamma]_{a_q}$*, which consists of the single triple* $\langle \top, X_2, X_4 \rangle$*. This means that the* $\overline{x}_3$*-edge from node* r *leads also to node* q*. (This relates to the merging condition in Definition 23.)*

Continuing this process (and using the value ordering z *before* $\overline{z}$*, for* z *equalling each of* x_1*,* x_2*,* x_3 *and* x_4*) we generate the variable ordering network in Figure 7.1.*

Proposition 27 (Section 7.3) limits the possible size of the generated variable ordering network $\tau(\Gamma)$. $Q_{[\Gamma]}(A)$ consists of all variables $X \in A$ such that there exists some $\varphi \in \Gamma$ such that $U_\varphi \ni X$ and either (i) $U_\varphi \not\subseteq A$ and $\{X_\varphi\} \cup W_\varphi \not\subseteq A$ or (ii) $X_\varphi \notin A$ and $W_\varphi \not\subseteq A$. For any given set of variables A the number of nodes r with $A_r = A$ is at most exponential in $|Q_{[\Gamma]}(A)|$ (rather than exponential in $|A|$).

The following result shows that if Γ is strongly conditionally acyclic then the construction given above is bound to succeed.

Proposition 28 *If* Γ *is a strongly conditionally acyclic cp-theory then* $\tau(\Gamma)$ *exists.*

Proof: The definition of strongly conditionally acyclic cp-theory ensures that for any $a \in \underline{A}$ there exists an a-undominated variable in $V - A$. This implies that the construction does not fail at any point. □

Proposition 29 below shows that if we find that $\tau(\Gamma)$ exists for locally consistent Γ then we have proved the consistency of Γ. Furthermore, Proposition 28 shows that if Γ happens to be strongly conditionally acyclic cp-theory, then the construction given above is bound to succeed, and hence will correctly determine (by Theorem 7) that Γ is consistent.

Proposition 29 *Let* Γ *be a cp-theory, and suppose that* $\tau(\Gamma)$ *exists. Then* $\tau(\Gamma)$ *respects* $[\Gamma]$*. Also,* Γ *is consistent if and only if it is locally consistent.*

Proof: The construction of $\tau(\Gamma)$ ensures that it is strongly compatible with $[\Gamma]$: Lemma 19 ensures that condition (i) of Definition 23 holds, and condition (ii) clearly holds. Hence, by Proposition 26, $\tau(\Gamma)$ respects $[\Gamma]$. By Theorem 7 (Section 7.4), Γ is consistent if and only if it is locally consistent. □

The variable ordering network $\tau(\Gamma)$ for Example B (in Section 3.2) has only seven nodes (see Example 15 and Figure 7.1) as opposed to the 31 nodes in the corresponding cs-tree $\sigma(\Gamma)$. More generally, one would expect when there are only a few variations in importance orderings, that the variable ordering network would be compact. Also the number of variable orderings associated with a variable ordering network can be exponential in the number of nodes, as the network 'factorises' the variable orderings, so even if we need many different orderings in different paths in a cs-tree, the variable ordering network may still be small, thus enabling consistency to be efficiently checked.

7.6 Generating an upper approximation from a variable ordering network

From locally consistent cp-theory Γ and variable ordering network τ which respects $[\Gamma]$ we can generate in a simple way a polynomial upper approximation for the preference relation $>_\Gamma$, which can be used for ordering and optimisation tasks as shown in Section 4.

Define irreflexive relation $\gg_\tau^\Gamma$ as follows. Consider two different outcomes α and β. Follow the assignment of α along the edges in τ from the root until a node r is reached such that α and β differ on Y_r, so that $\alpha(A_r) = \beta(A_r)$ and $\alpha(Y_r) \neq \beta(Y_r)$. As for cs-trees, we call this node, *the node that divides α and β*. Define $\alpha \gg_\tau^\Gamma \beta$ to hold if and only if $\alpha(Y_r) \succ_{a_r}^{Y_r} \beta(Y_r)$ holds (see Definition 4).

Theorem 8 below shows that $\gg_\tau^\Gamma$ is indeed an upper approximation for $>_\Gamma$. Note that if the local orderings $\succ_a^Y$ are all total orders then $\gg_\tau^\Gamma$ is a strict total order, so would then not be useful for generating more than one optimal solution of a constrained optimisation problem. $\gg_\tau^\Gamma$ will tend to be a cruder approximation of $>_\Gamma$ than the other two upper approximations defined in the paper (in Sections 5.2 and 6.3). To prove Theorem 8 we use a similar technique as we used for proving Theorems 4 and 5, by showing that $\gg_\tau^\Gamma$ is the intersection of a set of cs-tree orders.

Lemma 21 *Let Γ be locally consistent cp-theory, and let τ be a variable ordering network which respects $[\Gamma]$. Let $\mathcal{Q}$ be the set of cs-trees σ which are compatible with τ and such that for all body nodes $r = \langle A, a, Y, \succ \rangle$ in σ, the associated strict total ordering $\succ$ extends local ordering $\succ_a^Y$. Then $\alpha \gg_\tau^\Gamma \beta$ if and only if $\alpha >_\sigma \beta$ for all $\sigma \in \mathcal{Q}$.*

This implies that $\gg_\tau^\Gamma$ is an upper approximation of the preference relation $>_\Gamma$:

Theorem 8 *Let Γ be a locally consistent cp-theory, and let τ be a variable ordering network that respects $[\Gamma]$. Then $\gg_\tau^\Gamma$ is a strict partial order, and $\gg_\tau^\Gamma$*

contains $>_\Gamma$*, i.e.,* $\alpha \gg_\tau^\Gamma \beta$ *holds for any outcomes* α *and* β *such that* $\alpha >_\Gamma \beta$*. Hence,* $\gg_\tau^\Gamma$ *is an upper approximation of the preference relation* $>_\Gamma$*.*

Proof: Lemma 21 implies that $\gg_\tau^\Gamma$ is the intersection of relations $>_\sigma$ over cs-trees σ in $\mathcal{Q}$. By Lemma 20, each such σ strongly satisfies Γ, so σ satisfies Γ, by Proposition 19, Section 6.1. Proposition 15 (Section 4.4) then implies that $\gg_\tau^\Gamma$ is a strict partial order containing $>_\Gamma$. □

8 Related Work

The work described here—which is based on and develops the papers (Wilson, 2004b, 2004a)—builds very much on the fundamental work on CP-nets, as described especially in (Boutilier et al., 1999) and (Boutilier et al., 2004a), as well as on other work by the authors of these papers, such as (Domshlak & Brafman, 2002; Brafman & Domshlak, 2002; Domshlak, 2002; Boutilier et al., 2004b; Brafman & Dimopoulos, 2004; Brafman, Domshlak, & Shimony, 2004; Brafman et al., 2006); indeed one of the main initial motivations of the current work was to show how CP-nets approaches could be generalised to a richer language.

Although most work has focused on acyclic CP-nets, this is a strong restriction, limiting their potential applicability. For analysis and discussion of non-acyclic CP-nets see Chapter 6 of (Domshlak, 2002), Section 4 of (Domshlak & Brafman, 2002), (Brafman & Dimopoulos, 2004) (Section 4), (Goldsmith et al., 2005) and (Xia, Conitzer, & Lang, 2008). One of the motivations for the work in this paper on determining consistency of a cp-theory is the hardness complexity results for this task for CP-nets: see (Domshlak, 2002; Domshlak & Brafman, 2002; Goldsmith et al., 2005, 2008).

Logic-based formalisms for comparative preferences, that also emphasise the importance of *ceteris paribus* interpretations, include (von Wright, 1963, 1972; Hansson, 1996, 2001a, 2001b; van Benthem, Girard, & Roy, 2009) from the Philosophy literature, as well as (Doyle & Wellman, 1994; McGeachie & Doyle, 2002, 2004; Lang, 2002, 2004; Bienvenu, Lang, & Wilson, 2010) in the AI literature. The preference statements in this paper are of the form $u : x > x' [W]$, focusing on the values x and x' of a single (not-necessarily Boolean) variable X, conditional on a partial assignment u, and irrespective of variables W, where the other variables are treated in a *ceteris paribus* manner. Other logic-based formalisms allow preferences between arbitrary propositional formulae, and vary on the interpretation of *ceteris paribus*. Formalisms defined in (Lang, 2004) (Definition 4) and (van Benthem et al., 2009; Bienvenu et al., 2010) also allow irrespective statements, with the latter formalisms being especially expressive.

One of the main tasks that this paper is concerned with is totally ordering the outcomes in a way that is compatible with the cp-theory's preference ordering. The work here is most closely related to that on ordering queries in (Boutilier et al., 2004a), and the use of search trees in (Boutilier et al., 2004a, 2004b; Brafman et al., 2006) for CP-nets and TCP-nets. However, another kind of

approach to this problem is to construct a value function or utility function on the set of outcomes (i.e., a function that assigns a number to each outcome) whose ordering is compatible with the preference relation. Work of this kind for acyclic CP-nets includes UCP-nets (Boutilier, Bacchus, & Brafman, 2001), soft constraints methods (Domshlak, Rossi, Venable, & Walsh, 2003; Domshlak et al., 2006), and (Brafman & Domshlak, 2008). An advantage of the search tree approaches over value function approaches is that it's easier to generate outcomes in decreasing preference order.

The upper approximation for the case of fully acyclic cp-theories (see Section 5.2) generalises the relation $\gg$ defined for ordering queries in (Boutilier et al., 2004a) for acyclic CP-nets, and this kind of approximation has also been considered in (Kaci & Prade, 2007). Another upper approximation, for cp-theories, is defined in (Wilson, 2006), and is generalised to more expressive preference languages in (Wilson, 2009).

9 Summary and Discussion

This paper has defined a formalism, cp-theories, that is a simple logic of comparative preferences, and basic formal properties are shown (Section 2). The relationship with CP-nets and TCP-nets has been analysed, regarding what can and cannot be expressed (Section 3). It is shown that cp-theories are, in a particular precise sense, more general than CP-nets and TCP-nets, and that they can represent natural statements that cannot be expressed by CP-nets and TCP-nets, such as those used in a lexicographic order. A result is also included showing that, for each n, there is essentially a unique acyclic CP-net on n Boolean variables that totally orders the outcomes.

Much of the paper (Sections 4, 5, 6 and 7) is dedicated to the related problems of determining consistency of a cp-theory, and totally ordering a set of outcomes in a way that is compatible with the cp-theory preference relation. The latter is a key basic task that can be used, in particular, for choosing the outcomes that are displayed first to the user. The approaches developed cover non-acyclic as well as acyclic cases of cp-theories, which means that the methods can be applied to non-acyclic CP-nets and TCP-nets; this is important since non-acyclic sets of preference statements can easily arise—there's nothing irrational about them.

Inconsistency indicates incoherence within the input preference statements, which could, for example, be because of the elicited statements not reflecting the user's preferences; hence the importance of being able to check consistency. However, determining consistency of a cp-theory (or a CP-net) is an extremely hard problem, so it is desirable to find incomplete methods that can sometimes prove consistency or prove inconsistency. A simple necessary condition for consistency is defined, called local consistency (Section 4.1); testing local consistency will often be easy (in particular if the parents set U_X of each variable X is small); inconsistency can thus sometimes be proved by showing local inconsistency. If the cp-theory is fully acyclic then consistency holds if and only

if local consistency holds (Section 5). In Section 6, sufficient conditions are derived for consistency which have much lower complexity, specifically, strong conditional acyclicity (Section 6.1) and cuc-acyclicity (Section 6.2).

The approaches for confirming consistency and for ordering tasks are based on complete search trees (cs-trees), similar to those used for depth-first search for a solution of a constraint satisfaction problem. A sufficient condition for consistency of a cp-theory Γ is that there is a cs-tree satisfying Γ. If one is looking to show that there exists a cs-tree satisfying locally consistent Γ, it is sufficient to satisfy certain constraints that Γ imposes on the variable orderings that appear in branches of the cs-tree (see e.g., Proposition 18, Section 6.1).

Explicitly constructing a cs-tree will not be possible, unless the number of variables is small, since its size is linear in the number of outcomes. Different compact representations of a satisfying cs-tree are derived, for expressing the variable orderings of a search tree. Of particular note is the variable ordering network, defined in Section 7, which uses a decision diagram representation for the variable orderings. These compact representations of a cs-tree thus can allow consistency of a cp-theory to be determined, and ordering tasks to be performed in an efficient way.

Unconstrained optimisation reduces (just like for the case of CP-nets) to solving a constraint satisfaction problem. Polynomial upper approximations of the preference relation have been derived (in Sections 5.2, 6.3 and 7.6), and we show how they can be useful for generating, in a relatively efficient way, a set of optimal solutions of a constraint satisfaction problem (see Section 4.5).

There are many directions in which this work might usefully be extended and developed; we list some of these.

- The language of conditional preferences only allows preferences of a single variable (conditional on other variables); some natural preference statements involve preferences over more than one variable, so it would be desirable to consider more general languages, in particular, on the lines of the languages considered in (Lang, 2004; McGeachie & Doyle, 2004) and (Wilson, 2009), and to see to what extent the methods of this paper can be extended.

- The idea of a cs-tree can be extended to include partially completed search trees. An upper approximation can be defined from this that is a closer approximation of the preference relation $>_\Gamma$ than ones defined in this paper (Wilson, 2006); this might be used, in conjunction with the cs-tree methods developed here, for constrained optimisation. In addition, propagation methods for improving the efficiency of constrained optimisation might be developed, in order to prune subtrees of the search tree which only contain outcomes that are dominated—with respect to the appropriate upper approximation—by a solution we've already found.

- The compact representation of variable orderings developed in Section 7 could be made yet more compact if weaker conditions for merging (see Section 7.5) are used when generating the variable ordering network.

- This paper hasn't addressed the important (but very hard (Boutilier et al., 2004a; Domshlak & Brafman, 2002; Goldsmith et al., 2008)) problem of dominance testing; approaches in Section 5 of (Boutilier et al., 2004a) and Proposition 8 of (Wilson, 2004b) can be developed for more general cp-theories.

Acknowledgements

This material is based upon works supported by the Science Foundation Ireland under Grant No. 05/IN/I886 and Grant No. 08/PI/I1912. Thanks also for the referees for their constructive comments which helped improve the paper.

Appendix: Proofs

Proof of Lemma 1 (i): If there exists a cycle in $\succ \cup \{(\beta, \alpha)\}$ then, since $\succ$ is acyclic, β, α must appear in the cycle. We thus have a sequence $\alpha \succ \cdots \succ \beta$ and so $\alpha \succ \beta$ by transitivity, which contradicts the hypothesis.

(ii): If strict partial order $\succ$ is not already a (strict) total order, there exists some pair α, β of different elements such that neither $\alpha \succ \beta$ nor $\beta \succ \alpha$. Arbitrarily choose such a pair. Let $\succ'$ be the transitive closure of $\succ \cup \{(\beta, \alpha)\}$. By (i), $\succ'$ is a strict partial order strictly containing $\succ$. Iterating this we eventually generate a strict total order containing $\succ$. (iii) follows from (ii) by first taking the transitive closure to generate a strict partial order. (iv) easily follows from (iii).

(v) Clearly, $\succ$ is a subset of the intersection of all strict total orders containing it. To prove the converse, it is sufficient to show that if it is not the case that $\alpha \succ \beta$ then (α, β) is not in the intersection of all strict total orders extending $\succ$, i.e., there exists some strict total order $>$ extending $\succ$ with $\alpha \not> \beta$. So, suppose that it is not the case that $\alpha \succ \beta$. If $\alpha = \beta$ then the implication follows immediately, so let us assume that $\alpha \neq \beta$. By (i), $\succ \cup \{(\beta, \alpha)\}$ is irreflexive and acyclic, so by (iii), there exists strict total order $>$ extending $\succ \cup \{(\beta, \alpha)\}$, so $\beta > \alpha$, and hence $\alpha \not> \beta$, as required.

Proof of Lemma 2 (i) We have that $> \models \Gamma$ if and only if $> \supseteq \Gamma^*$, which, since $>$ is transitive, is if and only if $>$ contains $>_\Gamma$, the transitive closure of Γ^*.

(ii) Γ is consistent if and only if there exists some strict total order $>$ extending Γ^*, which, by Lemma 1(iv) is if and only if Γ^* is acyclic. Γ^* is acyclic if and only if its transitive closure $>_\Gamma$ is irreflexive, and hence is a strict partial order.

(iii) If Γ is consistent then, by (ii), $>_\Gamma$ is a strict partial order, and so, by Lemma 1(v), is equal to the intersection of all strict total orders extending it, i.e., by (i), the intersection of all strict total orders satisfying Γ.

(iv) Suppose that Γ is consistent. $\Gamma \models (\alpha, \beta)$ if and only if $\alpha > \beta$ holds for all strict total orders $>$ satisfying Γ, which is if and only if the intersection of

all strict orders satisfying Γ contains the pair (α, β), which, by (iii), is if and only if $\alpha >_\Gamma \beta$.

(v) Suppose that Γ is consistent. $\Gamma \models \varphi$ holds if and only if $>$ extends φ^* for all strict total orders $>$ satisfying Γ, which is if and only if the intersection of all strict total orders satisfying Γ is a superset of φ^*, which, by (iii), is if and only if $>_\Gamma \supseteq \varphi^*$.

Proof of Proposition 2 Let $>$ be an arbitrary strict total order on the set $\underline{V}$ of outcomes.

- Let $X \in V$, $u \in \underline{U_X}$, and $x, x' \in \underline{V}$ be such that $x \succ_u^X x'$. $>$ satisfies the cp-theory statement $u : x > x'[\emptyset]$ if and only if for all $t \in \underline{T}$ $tux > tux'$, where $T = V - \{X\} - U_X$, which is if and only if $>$ satisfies $\succ_u^X$. Therefore, $>$ satisfies the conditional preference table if and only if $>$ satisfies the cp-theory Γ_{cp}.

- $>$ satisfies $\Gamma_{X \to Y}$ if and only if for all $u \in \underline{U_X}$ and x and x' such that $x \succ_u^X x'$ holds, we have $>$ satisfies $u : x > x' [Y]$, i.e., for all $y, y' \in \underline{Y}$ and for all assignments r to $V - \{X, Y\}$ extending u, $rxy > rx'y'$. This is if and only if $>$ satisfies i-arc $X \to Y$. Hence, $>$ satisfies all the i-arcs of N if and only if $>$ satisfies Γ_{i}.

- $>$ satisfies $\Gamma_{X \to_s Y}$ if and only if $>$ satisfies $qs : x > x' [Y]$ for all assignments q to $U_X - S_{X,Y}$ and all x, x' such that $x \succ_u^X x'$ holds, where u is qs restricted to U_X. This holds if and only if $rsxy > rsx'y'$ holds for all $y, y' \in \underline{Y}$, for all assignments r to $V - S_{X,Y} - \{X, Y\}$, for all x, x' such that $x \succ_u^X x'$, where u is rs restricted to U_X. This is if and only if $>$ satisfies the ci-statement $X \to_s Y$. Hence, $>$ satisfies all the ci-statements in N if and only if $>$ satisfies Γ_{ci}.

Putting these together, $>$ satisfies the TCP-net$N$ if and only if $>$ satisfies the cp-theory Γ_N. Therefore, N is satisfiable if and only if Γ_N is consistent.

Furthermore, if N is satisfiable (and so Γ_N is consistent), $>_N$ is the intersection of all strict total orders satisfying N, i.e., the intersection of all strict total orders satisfying Γ_N, which equals $>_{\Gamma_N}$ by Lemma 2(iii).

Proof of Proposition 3 We'll show that $>_\Gamma$ equals $>_{lex}$ by showing that Γ^* equals $>_{lex}$. Because $>_{lex}$ is transitive we then have $>_\Gamma$, the transitive closure of Γ^* is equal to $>_{lex}$.

To show $\Gamma^* \subseteq >_{lex}$: suppose $(\alpha, \beta) \in \Gamma^*$. Then for some i, $(\alpha, \beta) \in (\Gamma_i)^*$, so there exists statement $\top : x > x' [\{X_{i+1}, \ldots, X_n\}]$ in Γ_i with $\alpha(X_i) = x$, $\beta(X_i) = x'$ and for all $j < i$, $\alpha(X_j) = \beta(X_j)$. We have $\alpha(X_i) >_i \beta(X_i)$ so $\alpha >_{lex} \beta$ as required.

To prove the converse, suppose $\alpha >_{lex} \beta$. Then for some $i \in \{1, \ldots, m\}$ we have $\alpha(X_i) >_i \beta(X_i)$ and for all $j < i$, $\alpha(X_j) = \beta(X_j)$. But then $(\alpha, \beta) \in \varphi^*$ where φ is the statement $\top : x > x' [\{X_{i+1}, \ldots, X_n\}]$, with $x = \alpha(X_i)$ and $x' = \beta(X_i)$. Since $\varphi \in \Gamma$, we have $(\alpha, \beta) \in \Gamma^*$.

Proof of Lemma 3 All three parts follow easily from the appropriate completeness theorems for swapping/flipping sequences. However, to make the presentation more self-contained, we instead use Propositions 1 and 2 in Section 3.1 and Section 3.2 for (ii) and (iii), respectively.

The first part follows immediately from Theorem 1: $\alpha >_\Gamma \beta$ implies that there exists a worsening swapping sequence from α to β; but since α covers β, there can be no element in the sequence between α and β, so there is a worsening swap from α to β.

(ii) If a CP-net N is unsatisfiable then $\succ_N$ degenerates to $\underline{V} \times \underline{V}$, which implies that no outcome covers any other outcome. (The same applies to TCP-nets.) Suppose α covers β with respect to $\succ_N$. CP-net N is then satisfiable, so $\succ_N$ equals $>_{\Gamma_N}$ by Proposition 1. Part (i) implies that there exists a worsening swap from α to β for Γ_N. Let $u : x > x' [W]$ be the relevant element of Γ_N used for this swap. The form of Γ_N implies that $W = \emptyset$, and so α and β differ on X and agree on all other variables.

The proof of (iii) is almost the same as that for (ii). Let N be a TCP-net, and suppose that α covers β with respect to $\succ_N$. N is then satisfiable, so $\succ_N$ equals $>_{\Gamma_N}$, by Proposition 2. Part (i) implies that there is a worsening swap from α to β for Γ_N. Let $u : x > x' [W]$ be the relevant element of Γ_N used for this swap. The form of Γ_N implies that W is empty or a singleton, so α and β differ on at least one variable (X) and at most two variables.

Proof of Proposition 5 Suppose to the contrary that there exists order $>_1$ on $\underline{X_1}$ such that X_1 dominates $\{X_2, \ldots, X_n\}$ with respect to $(\succ_N, >_1)$. Write $>_2$ for the total order on $\underline{X_2}$ given in the specification of the CP-net; (it is unconditional, as X_2 has an empty set of parents).

Because $>_1$ is non-empty we can choose $\alpha, \beta \in \underline{V}$ such that $\alpha(X_1) >_1 \beta(X_1)$ and $\beta(X_2) >_2 \alpha(X_2)$. Because $\alpha(X_1) >_1 \beta(X_1)$ we have $\alpha \succ \beta$, by the dominance of X_1. Therefore, there exists a sequence $\alpha = \alpha_1, \ldots, \alpha_k = \beta$ of outcomes such that, for $i = 1, \ldots, k-1$, there is a worsening flip from α_i to α_{i+1}. For each i we must either have $\alpha_i(X_2) = \alpha_{i+1}(X_2)$ or $\alpha_i(X_2) >_2 \alpha_{i+1}(X_2)$. This implies that either $\alpha_1(X_2) = \alpha_k(X_2)$ or $\alpha_1(X_2) >_2 \alpha_k(X_2)$; but neither of these are possible as $\alpha_k(X_2) = \beta(X_2) >_2 \alpha(X_2) = \alpha_1(X_2)$.

Proof of Proposition 6 Let x_1 be the $>_1$-maximal element in $\underline{X_1}$. Let α be a $\succ_M$-minimal element in $A = \{\alpha : \alpha(X_1) = x_1\}$ (this exists because $\succ_M$ is acyclic), and let β be a $\succ_M$-maximal element in $B = \underline{V} - A = \{\beta : x_1 >_1 \beta(X_1)\}$. Since $\alpha(X_1) >_1 \beta(X_1)$, by the dominance of X_1 we have $\alpha \succ \beta$. Furthermore, if $\gamma \neq \alpha, \beta$ then either $\gamma \in A$ and so it is not the case that $\alpha \succ \gamma$, or $\gamma \in B$ and it is not the case that $\gamma \succ \beta$. This shows that there does not exist γ with $\alpha \succ \gamma \succ \beta$, proving that α covers β with respect to $\succ_M$. By Lemma 3(iii), it must then be the case that α and β differ on at most two variables; we will obtain the required contradiction by showing that α and β differ on at least three variables.

We have $\alpha(X_1) \neq \beta(X_1)$. Suppose $\alpha(X_2) = \beta(X_2)$; call this element x_2, and let x_2' be any other element of $\underline{X_2}$. Since X_2 has no parents, we either (a) have some CP statement $\top : x_2 > x_2'$; or (b) have some CP statement $\top : x_2' > x_2$. If (a) there is a worsening flip from α to some element $\alpha' \in A$, and so $\alpha \succ \alpha'$, contradicting the definition of α. If (b) we can perform an improving flip on β to produce $\beta' \in B$ with $\beta' \succ \beta$, contradicting the definition of β. Therefore, $\alpha(X_2) \neq \beta(X_2)$.

We can use exactly the same argument to show that $\alpha(X_3) \neq \beta(X_3)$, so α and β differ on at least three variables, which is the contradiction required.

Proof of Lemma 4 (a) $\succ_{N'}$ is transitive by its construction. We just need to show that it is irreflexive and complete (i.e., for all outcomes α and β, either $\alpha = \beta$ or $\alpha \succ_{N'} \beta$ or $\beta \succ_{N'} \alpha$).

Suppose that $\delta \succ_{N'} \delta$ for some $\delta \in \underline{V - \{Z\}}$. Then there exists a worsening flipping sequence from δ to δ in N'. Let δ^0 be δ extended with the assignment $Z = 0$. Applying the same flips yields a worsening flipping sequence for N from δ^0 to δ^0, showing that $\delta^0 \succ_N \delta^0$, contradicting the assumption that $\succ_N$ is a strict total order.

Consider any two different elements δ and ϵ of $\underline{V - \{Z\}}$. Extend each of these to an element of $\underline{V}$ by assigning $Z = 0$, leading to outcomes δ^0 and ϵ^0, respectively. Since $\succ_N$ is a strict total order, we either have $\delta^0 \succ_N \epsilon^0$ or $\epsilon^0 \succ_N \delta^0$. Without loss of generality, assume that $\delta^0 \succ_N \epsilon^0$. There exists a worsening flipping sequence for N from δ^0 to ϵ^0. Removing any flips of Z generates a worsening flipping sequence for N' from δ to ϵ, proving $\delta \succ_{N'} \epsilon$ and hence completeness.

(b) First suppose that $\alpha \succ_N \beta$, so that there exists a worsening flipping sequence for N from α to β. By ignoring flips of Z we can generate a worsening flipping sequence for N' from $\alpha(V - \{Z\})$ to $\beta(V - \{Z\})$, showing that $\alpha(V - \{Z\}) \succ_{N'} \beta(V - \{Z\})$. The converse then follows immediately from the fact that $\succ_N$ and $\succ_{N'}$ are strict total orders: for if $\alpha \succ_N \beta$ does not hold then $\beta \succ_N \alpha$ holds, implying $\beta(V-\{Z\}) \succ_{N'} \alpha(V-\{Z\})$ and hence $\alpha(V-\{Z\}) \succ_{N'} \beta(V - \{Z\})$ does not hold.

(c) Suppose that α and β differ on Z and agree on all other variables, but that α and β are not consecutive with respect to $\succ_N$. Since $\succ_N$ is a total order, there exists $\gamma \in \underline{V}$, different from α and β, with either $\alpha \succ_N \gamma \succ_N \beta$ or $\beta \succ_N \gamma \succ_N \alpha$. Without loss of generality, assume $\alpha \succ_N \gamma \succ_N \beta$. Since Z has only two values, γ must differ with α and β on $V - \{Z\}$. There exists a worsening flipping sequence for N from α to β passing through γ. By ignoring flips of Z we can generate a worsening flipping sequence for N' from $\alpha(V-\{Z\})$ to $\beta(V - \{Z\}) = \alpha(V - \{Z\})$, which contradicts the irreflexivity of $\succ_{N'}$ shown in (a).

If outcomes δ and γ differ on Z but agree on all other variables, write $\delta = \overline{\gamma}$ and $\gamma = \overline{\delta}$ (recall that Z has only two values). The first part implies that α_1 and $\overline{\alpha_1}$ are consecutive, so $\alpha_2 = \overline{\alpha_1}$. Similarly, α_3 and $\overline{\alpha_3}$ are consecutive, so $\overline{\alpha_3}$ equals α_2 or α_4. But $\overline{\alpha_3} = \alpha_2$ would imply $\alpha_3 = \alpha_1$ (since Z has only

two values), so we must have $\overline{\alpha_3} = \alpha_4$. Continuing this shows that for all $j = 1, 2, \ldots, \frac{K}{2}$, $\alpha_{2j} = \overline{\alpha_{2j-1}}$. and so α_{2j-1} and α_{2j} agree on $V - \{Z\}$.

(d) Suppose that $\alpha_1(Z) = 1$. Since Z takes two values, for each element α of $\underline{V}$, there exists exactly one other element of $\underline{V}$ which agrees with α on $V - \{Z\}$. By part (c), α_1 and α_2 agree on $V - \{Z\}$ and so differ on Z (otherwise they would be the same outcome). Similarly, for all $j = 1, 2, \ldots, K/2$, outcomes α_{2j-1} and α_{2j} agree on $V - \{Z\}$ and so differ on Z. α_2 and α_3 differ on $V - \{Z\}$; they are consecutive so there exists a worsening flip from α_2 to α_3; thus, they differ on exactly one variable, and hence agree on Z. Similarly, for all $j = 1, 2, \ldots$, outcomes α_{2j} and α_{2j+1} agree on Z. We thus have the sequence of values of Z being $1, 0, 0, 1, 1, 0, 0, 1, 1$, and so on.

Proof of Proposition 7 We show this by induction on n. This is clearly true for $n = 1$, since there is only one strict total order satisfying property (i). Suppose it is true for $n = k$. We will show that it holds also for $n = k+1$, hence implying that it is true for all natural numbers n. Suppose that $>_1$ and $>_2$ are both strict total orders on $\underline{V}$ corresponding to CP-nets N_1 and N_2, respectively, satisfying properties (i) and (ii). Let N_1' and N_2' be the associated CP-nets on $\{X_1, \ldots, X_k\}$ defined in Lemma 4. By Lemma 4(a), $\succ_{N_1'}$ and $\succ_{N_2'}$ are both strict total orders, so, by induction, are equal.

Lemma 4(b), (c) and (d) then imply that $\succ_{N_1}$ and $\succ_{N_2}$ are equal: by Lemma 4(b), if $\alpha, \beta \in \underline{V}$ differ on $V - \{Z\}$ then $\alpha \succ_{N_1} \beta$ if and only if $\alpha(V - \{Z\}) \succ_{N_1'} \beta(V - \{Z\})$ if and only if $\alpha(V - \{Z\}) \succ_{N_2'} \beta(V - \{Z\})$ if and only if $\alpha \succ_{N_2} \beta$. If different outcomes $\alpha, \beta \in \underline{V}$ agree on $V - \{Z\}$ then by Lemma 4(c) they are consecutive in the orderings $\succ_{N_1}$ and $\succ_{N_2}$. Lemma 4(b) and the fact that $\succ_{N_1'}$ and $\succ_{N_2'}$ are equal implies that the ordering of $\{\alpha, \beta\}$ relative to the other outcomes is the same in $\succ_{N_1}$ and $\succ_{N_2}$. Lemma 4(d) then implies that both $\succ_{N_1}$ and $\succ_{N_2}$ order α and β the same way, i.e., $\alpha \succ_{N_1} \beta \iff \alpha \succ_{N_2} \beta$. Therefore, $>_1$ equals $>_2$, completing the inductive step.

Proof of Proposition 8 This CP-net N can be restricted to a CP-net N_i on V_i for $1 \leq i \leq n$, where $N_n = N$. We shall prove by induction that $\succ_{N_i}$ is a strict total order on $\underline{V_i}$ with maximum element $(1, \ldots, 1)$ for each $i = 1, \ldots, n$, proving that $\succ_N$ (i.e., $\succ_{N_n}$) is a strict total order.

The base case: $i = 1$. Then $1 \succ_{\top}^{X_1} 0$ since $\top$ contains an even number of zeros, as it contains no zeros. Hence, $1 \succ_{N_i} 0$ and it is not the case that $0 \succ_{N_i} 1$, so $\succ_{N_i}$ is a strict total order.

Suppose, by induction, that $\succ_{N_{i-1}}$ is a strict total order with maximum element $(1, \ldots, 1)$; we need to show that $\succ_{N_i}$ is a strict total order with maximum element $(1, \ldots, 1)$. Consider any $v, v' \in \underline{V_i}$ with $v \neq v'$. We will prove that either $v \succ_{N_i} v'$ or $v' \succ_{N_i} v$. This implies that there is at most one model satisfying N_i. Acyclicity implies that N_i has at least one model (see Theorem 1 of (Boutilier et al., 2004a)), so has exactly one model, which is then equal to $\succ_{N_i}$. This implies that $\succ_{N_i}$ is a strict total order; Lemma 4(b),(c),(d) and the inductive hypothesis can then be used to show that the maximum element is

$(1, \ldots, 1)$, completing the inductive step.

Case (i) $v(V_{i-1}) = v'(V_{i-1})$ which equals u, say. Then v and v' differ on only one variable, X_i, and $v(X_i) = 1$ and $v'(X_i) = 0$ or vice versa. We either have $1 \succ_u^{X_i} 0$ or $0 \succ_u^{X_i} 1$, which implies that either $u1 \succ_{N_i} u0$ or $u0 \succ_{N_i} u1$, so either $v \succ_{N_i} v'$ or $v' \succ_{N_i} v$.

Case (ii) $v(V_{i-1}) \neq v'(V_{i-1})$. Let $u = v(V_{i-1})$ and $u' = v'(V_{i-1})$. By the inductive hypothesis we have either $u \succ_{N_{i-1}} u'$ or vice versa. Assume without loss of generality that $u \succ_{N_{i-1}} u'$. Let u'' be minimal, with respect to $\succ_{N_{i-1}}$, such that $u'' \succ_{N_{i-1}} u'$, and define v'' to be u'' extended with the assignment $v(X_i)$ to X_i. We shall show that (a) $v'' \succ_{N_i} v'$ and (b) either $v = v''$ or $v \succ_{N_i} v''$. Putting these together will imply by transitivity that $v \succ_{N_i} v'$, as required.

To show (a): since u'' covers u', by Lemma 3 (Section 3.3), there exists a worsening flip from u'' to u'. We can also apply this worsening flip to get $u''1 \succ_{N_i} u'1$ and $u''0 \succ_{N_i} u'0$. If $1 \succ_{u''}^{X_i} 0$ then u'' must contain an even number of zeros, and so u' contains an odd number of zeros, since they differ on precisely one variable, and so $0 \succ_{u'}^{X_i} 1$. This implies that $u''1 \succ_{N_i} u''0 \succ_{N_i} u'0 \succ_{N_i} u'1$, which implies that $v'' \succ_{N_i} v'$, as required. A similar argument applies if $0 \succ_{u''}^{X_i} 1$.

To show (b): If $v \neq v''$ then $u \neq u''$ so $u \succ_{N_{i-1}} u''$, since u'' covers u' and $u \succ_{N_{i-1}} u'$, using the fact that $\succ_{N_{i-1}}$ is a strict total order. So, there exists a worsening sequence of flips from u to u''. The same sequence of flips can be used from v to v'', showing that $v \succ_{N_i} v''$.

Proof of Proposition 11 The problem is in coNP since if Γ is not locally consistent we can non-deterministically choose α and X such that $\succ_\alpha^X$ (which can be computed in polynomial time) is not irreflexive, i.e., there exists $x \in \underline{X}$ with $x \succ_\alpha^X x$.

To show coNP-hardness we can use a reduction from 3-SAT. Consider an instance of 3-SAT with m clauses involving propositional variables V'. For $k = 1, \ldots, m$, let c^k be the k^{th} clause, which we write as $l_1^k \vee l_2^k \vee l_3^k$. We generate a cp-theory Γ as follows: let V be $V' \cup \{Z\}$ where Z has domain $\{z_0, \ldots, z_m\}$. Let Γ_k consist of the three statements $l_j^k : z_{k-1} > z_k \, [\emptyset]$, for $j = 1, 2, 3$, and let Γ be $\Gamma_1 \cup \cdots \cup \Gamma_m \cup \{\top : z_k > z_0 \, [\emptyset]\}$. Then Γ is not locally consistent if and only if there exists an assignment u to V' with $\succ_u^Z$ not irreflexive, which is if and only if for each k there exists j with u satisfying l_j^k, which is if and only if u satisfies the 3-SAT instance.

Proof of Proposition 13 Sufficiency of (1) and (2): We need to show that $>_\sigma \supseteq \varphi^*$ for all $\varphi \in \Gamma$. Consider any $(\alpha, \beta) \in \varphi^*$. α and β are different outcomes, so there exists some node r of σ that divides them. $\alpha \models u_\varphi$ so, by (1), on the path to α, X_φ appears before all of W_φ. This implies that $Y_r = X_\varphi$, since α and β only differ on $\{X_\varphi\} \cup W_\varphi$, and certainly differ on X_φ. Since $\alpha \models u_\varphi, a_r$, tuples u_φ and a_r are compatible so, by (2), $x_\varphi \succ_r x'_\varphi$, i.e., $\alpha(X_\varphi) \succ_r \beta(X_\varphi)$ (since $(\alpha, \beta) \in \varphi^*$), which shows that $\alpha >_\sigma \beta$, i.e., $(\alpha, \beta) \in >_\sigma$. Since (α, β) is an arbitrary element of φ^*, and φ is an arbitrary element of Γ, we have that $>_\sigma \supseteq \varphi^*$ for all $\varphi \in \Gamma$.

Conversely, we need to prove the necessity of (1) and (2). Necessity of (1): Suppose that $\varphi \in \Gamma$ and $\alpha \models u_\varphi$, but that there exists some $Y \in W_\varphi$ which appears before X_φ on the path to α. Let r be the node with $Y_r = Y$. Since $\underline{Y}$ has at least two values, we can choose values y, y' of Y with $y \succ_r y'$. Define outcomes α' and β as follows: $\alpha'(X_\varphi) = x_\varphi$, $\beta(X_\varphi) = x'_\varphi$, $\alpha'(Y) = y'$, $\beta(Y) = y$, and for all other variables $Z \neq X_\varphi, Y$ define $\alpha'(Z) = \beta(Z) = \alpha(Z)$. Then r divides α' and β, so $\alpha' \not\succ_\sigma \beta$, since $\alpha'(Y_r) \not\succ_r \beta(Y_r)$. However, $(\alpha', \beta) \in \varphi^*$, since $\alpha \models u_\varphi$ and hence $\alpha', \beta \models u_\varphi$ (using the fact that U_φ is disjoint from $\{X_\varphi\} \cup W_\varphi$, and so α' and β agree with α on U_φ). Therefore, $>_\sigma \not\supseteq \varphi^*$, and so it is not the case that $>_\sigma$ satisfies Γ, proving the necessity of (1).

Necessity of (2): Suppose that σ satisfies Γ, and that for node r, and $\varphi \in \Gamma$, we have $X_\varphi = Y_r$ and u_φ is compatible with a_r. Choose any outcome α which extends u_φ and a_r, and satisfies $\alpha(X_\varphi) = x_\varphi$ (this is possible since $X_\varphi \notin U_\varphi \cup A_r$). Define outcome β to satisfy $\beta(X_\varphi) = x'_\varphi$, and to agree with α on all other outcomes. $(\alpha, \beta) \in \varphi^* \subseteq \Gamma^*$, so $\alpha >_\Gamma \beta$, and hence, $\alpha >_\sigma \beta$. Node r divides α and β, so we have $x_\varphi = \alpha(Y) \succ_r \beta(Y) = x'_\varphi$, by definition of $>_\sigma$, and hence, $x_\varphi \succ_r x'_\varphi$, as required.

Proof of Proposition 14 Let σ be any cs-tree satisfying (1′) and (2′). Using Proposition 13, it is sufficient to show that σ satisfies (1) and (2) of Proposition 13. (1) follows immediately from (1′). Regarding (2), let $r = \langle A, a, Y, \succ \rangle$ be any node, and φ be any element of Γ such that $X_\varphi = Y$ and u_φ are compatible with a. Since u_φ is compatible with a, there exists some outcome, say, α, which extends both of them, and so r is on the path to α. By (1′), U_φ appears before X_φ, so $U_\varphi \subseteq A$. Since u_φ is compatible with a, we have that a extends u_φ. By definition (see Definition 4), $x_\varphi \succ_{u_\varphi}^{X_\varphi} x'_\varphi$, so $x_\varphi \succ_a^Y x'_\varphi$, and hence by (2′), $x_\varphi \succ x'_\varphi$, proving (2).

Proof of Proposition 16 Suppose that α is not optimal, so that there exists some outcome β with $\beta >_\Gamma \alpha$. Since $>_\Gamma$ is the transitive closure of Γ^* there exists outcome γ with $(\gamma, \alpha) \in \Gamma^*$, so for some $\varphi \in \Gamma$, $(\gamma, \alpha) \in \varphi^*$. We have $\alpha(X_\varphi) = x'_\varphi$ and $\alpha(U_\varphi) = u_\varphi$, so α is not a solution of C_Γ since it does not satisfy c_φ.

Conversely, suppose that outcome α is not a solution of C_Γ. Then there exists some $\varphi \in \Gamma$ such that α does not satisfy c_φ, and so $\alpha(X_\varphi) = x'_\varphi$ and $\alpha(U_\varphi) = u_\varphi$. Define outcome γ to be equal to α on all variables except X_φ, and define $\gamma(X_\varphi) = x_\varphi$. We have $(\gamma, \alpha) \in \varphi^*$ so $\gamma >_\Gamma \alpha$, showing that α is not optimal.

Proof of Lemma 8 First, suppose that $\alpha \succ_{p(\Gamma)} \beta$, and consider any $\sigma \in \mathcal{S}$. Let $r = \langle A, a, Y, \succ \rangle$ be the node that divides α and β, so that $\alpha(A) = \beta(A) = a$. Condition (a) implies that $U_Y \subseteq A$. $\alpha(Y) \neq \beta(Y)$, so $Y \in \Delta(\alpha, \beta)$.

Suppose that $Y \notin \Theta(\alpha, \beta)$. Then there exists $Z \in \Delta(\alpha, \beta)$ which is an ancestor of Y with respect to $G(\Gamma)$. Condition (a) above implies that Z would

appear before Y on the path to α, so that $Z \in A$; α and β agree on A and hence Z (since r divides α and β) which is contradiction since $Z \in \Delta(\alpha, \beta)$.

We have proved that $Y \in \Theta(\alpha, \beta)$. Since $\alpha \succ_{p(\Gamma)} \beta$ we have $\alpha(Y) \succ_{\alpha}^{Y} \beta(Y)$. Let $u = \alpha(U_Y)$, which equals $a(U_Y)$, because $U_Y \subseteq A$ and α extends a. Thus, using Lemma 5, $\alpha(Y) \succ_{u}^{Y} \beta(Y)$, which implies that $\alpha(Y) \succ_r \beta(Y)$ by condition (b), and hence $\alpha >_\sigma \beta$, as required.

Conversely, suppose that it is not the case that $\alpha \succ_{p(\Gamma)} \beta$; we will show that there exists some $\sigma \in \mathcal{S}$ with $\alpha \not>_\sigma \beta$. This follows immediately if $\alpha = \beta$, so let us assume that $\alpha \neq \beta$.

Since $\alpha \not\succ_{p(\Gamma)} \beta$, there exists $Y \in \Theta(\alpha, \beta)$ such that $\alpha(Y) \not\succ_{\alpha}^{Y} \beta(Y)$. We can create a cs-tree σ satisfying the following properties:-

(i) σ uses a fixed variable ordering compatible with $G(\Gamma)$ in all paths from the root to outcomes, where Y is only preceded in this ordering by its ancestors in $G(\Gamma)$;

(ii) let r' be the node that divides α and β; by (i), Y is only preceded in the variable ordering by variables not in $\Theta(\alpha, \beta)$, and so not in $\Delta(\alpha, \beta)$, hence α and β agree on variables before Y; this implies that $Y_{r'} = Y$; we choose $\succ_{r'}$ to be some strict total order extending $\succ_{\alpha}^{Y}$ and such that $\alpha(Y) \not\succ_{r'} \beta(Y)$;

(iii) for all other nodes r in σ, set $\succ_r$ to be some strict total order extending $\succ_{u}^{Y_r}$, where $u = a_r(U_{Y_r})$, and U_{Y_r} is the parents of Y_r in $H(\Gamma)$.

Then σ satisfies conditions (a) and (b), so is in $\mathcal{S}$. We also have $\alpha \not>_\sigma \beta$.

Proof of Proposition 17 For any φ, $(\overline{\varphi})^* \supseteq \varphi^*$, so $(\overline{\Gamma})^* \supseteq \Gamma^*$, and hence, taking the transitive closure of both sides, $>_{\overline{\Gamma}} \supseteq >_\Gamma$. Also, $H(\overline{\Gamma}) = H(\Gamma)$ so $\overline{\Gamma}$ is locally consistent if and only if Γ is locally consistent, as they both give rise to the same local orderings $\succ_{\alpha}^{X}$ on $\underline{X}$. The transitive closure $G^\circ(\Gamma)$ of $G(\Gamma)$ is equal to the transitive closure $G^\circ(\overline{\Gamma})$ of $G(\overline{\Gamma})$ so, for $U \subseteq V$, $\max_{G^\circ(\Gamma)}(U) = \max_{G^\circ(\overline{\Gamma})}(U)$. These observations imply that $\succ_{p(\Gamma)}$ equals $\succ_{p(\overline{\Gamma})}$. By Theorem 4 (applied to $\overline{\Gamma}$), $\succ_{p(\overline{\Gamma})} \supseteq >_{\overline{\Gamma}}$. It just remains to show that $\succ_{p(\Gamma)} \subseteq >_{\overline{\Gamma}}$.

Suppose $\alpha \succ_{p(\Gamma)} \beta$, so $\alpha \neq \beta$. Abbreviate $G(\Gamma)$ to G, abbreviate $H(\Gamma)$ to H, and $\Theta(\alpha, \beta)$ to Θ. Write Θ as $\{X_1, \ldots, X_k\}$. Since $\Delta(\alpha, \beta) \neq \emptyset$ and G is acyclic, $\Theta \neq \emptyset$.

Define $\beta_0 = \alpha$, and for $i = 1, \ldots, k$ define β_i inductively as follows:

$\beta_i(X_i) = \beta(X_i)$, and $\beta_i(Z) = \beta(Z)$ for all Z that are descendants of X_i in G, (i.e., such that (X_i, Z) is in the transitive closure of G). For all other $Y \in V$, let $\beta_i(Y) = \beta_{i-1}(Y)$. We have $\beta_k = \beta$, since every element of $\Delta(\alpha, \beta)$ is equal to, or is a descendant of, some X_i in Θ.

For each $X_i \in \Theta$, $\mathrm{Pa}_H(X_i) \cap \Delta(\alpha, \beta) = \emptyset$, else X_i would be a descendant of a variable in $\Delta(\alpha, \beta)$, contradicting the definition of Θ. So, $\alpha(\mathrm{Pa}_H(X_i)) = \beta(\mathrm{Pa}_H(X_i))$. We therefore have, for all $j = 0, \ldots, k$, $\alpha(\mathrm{Pa}_H(X_i)) = \beta_j(\mathrm{Pa}_H(X_i))$. This implies $\succ_{\alpha}^{X_i}$ equals $\succ_{\beta_j}^{X_i}$ for all i and j. By definition of $\succ_{p(\Gamma)}$ we have for

each $X_i \in \Theta$, $\alpha(X_i) \succ_{\alpha}^{X_i} \beta(X_i)$. So, for all j, $\alpha(X_i) \succ_{\beta_j}^{X_i} \beta(X_i)$, and hence, $\beta_{i-1}(X_i) \succ_{\beta_{i-1}}^{X_i} \beta_i(X_i)$, since $\beta_{i-1}(X_i) = \alpha(X_i)$, because X_i is not a descendant of any of $X_1, \ldots, X_{i-1}$. This implies $\beta_{i-1} >_{\overline{\Gamma}} \beta_i$, since we can apply a sequence of worsening swaps for $\overline{\Gamma}$ which start with β_{i-1}, change with the last swap the value of X_i to $\beta_i(X_i)$, and with (e.g.,) the last swap also changing the values of all descendants of X_i to their values in β_i. Since $>_{\overline{\Gamma}}$ is transitive, we have $\beta_0 >_{\overline{\Gamma}} \beta_k$, i.e., $\alpha >_{\overline{\Gamma}} \beta$. Thus, $\succ_{p(\Gamma)} \subseteq >_{\overline{\Gamma}}$, as required.

Proof of Proposition 18 First let us assume that conditions (1) and (2) hold. We need to show that σ strongly satisfies Γ, i.e., for any body node r of σ, with associated tuple $\langle A, a, Y, \succ \rangle$, that (I) Y is strongly a-undominated, and (II) the ordering $\succ$ on $\underline{Y}$ extends $\succ_a^Y$. (II) follows immediately from (2).

To prove (I), we need to show that for all $\varphi \in \Gamma$ such that u_φ is compatible with a, (i) if $X_\varphi = Y$ then $U_\varphi \subseteq A$; (ii) if $W_\varphi \ni Y$ then $U_\varphi \cup \{X_\varphi\} \subseteq A$. Since u_φ is compatible with a, there exists an outcome α that extends both. r is on the path to α, so, by (1), if $X_\varphi = Y$ then U_φ appears before Y, i.e., $U_\varphi \subseteq A$, proving (i). Similarly, if $W_\varphi \ni Y$, then U_φ and X_φ appear before Y on the path to α, so $U_\varphi \cup \{X_\varphi\} \subseteq A$, proving (ii).

Conversely, suppose that σ strongly satisfies Γ. We need to show that conditions (1) and (2) hold. For any body node r of σ with associated tuple $\langle A, a, Y, \succ \rangle$, r strongly satisfies Γ, so $\succ$ extends $\succ_a^Y$, proving (2).

To prove (1), consider any $\varphi \in \Gamma$ and any outcome α such that $\alpha \models u_\varphi$. Consider first the node r on the path to α with $Y_r = X_\varphi$. Then, Y_r is strongly a_r-undominated, so $U_\varphi \subseteq A_r$, and hence, U_φ appears before X_φ on the path to α. Now consider any node r on the path to α with $Y_r \in W_\varphi$. Since Y_r is strongly a_r-undominated, we have $U_\varphi \cup \{X_\varphi\} \subseteq A_r$ showing that X_φ appears before each element of W_φ on the path to α, completing the proof of (1).

Proof of Lemma 10 $\Rightarrow$: Suppose that Y is F_a-dominated by some element Z of $V - A$. Then there exists $\varphi \in \Gamma$ such that u_φ is compatible with a and either (i) $Y = X_\varphi$ and $Z \in U_\varphi$, or (ii) $Y \in W_\varphi$ and $Z \in U_\varphi \cup \{X_\varphi\}$. In either case, Y is not strongly a-undominated.

$\Leftarrow$: Suppose that Y is not F_a-dominated by any element of $V - A$. Consider any $\varphi \in \Gamma$ such that u_φ is compatible with a. (i) If $X_\varphi = Y$ then $U_\varphi \cap (V - A) = \emptyset$, and so $U_\varphi \subseteq A$. (ii) If $Y \in W_\varphi$ then $(\{X_\varphi\} \cup U_\varphi) \cap (V - A) = \emptyset$ and so $\{X_\varphi\} \cup U_\varphi \subseteq A$. This proves that Y is strongly a-undominated.

Proof of Proposition 20 (i): $F_b \subseteq F_a$ follows immediately from the fact that, for any $\varphi \in \Gamma$, if b is compatible with u_φ then a is compatible with u_φ.

(ii) Suppose $Y \in V - B$ is strongly a-undominated. By Lemma 10, Y is not F_a-dominated by any element of $V - A$, so, using the fact that $F_b \subseteq F_a$, Y is not F_b-dominated by any element of $V - A$, so in particular is not F_b-dominated by any element of $V - B$. By Lemma 10, Y is strongly b-undominated.

(iii) Suppose that F_a restricted to $V - A$ is acyclic, and consider any b extending a. F_b is a subrelation of F_a, so F_b restricted to $V - A$ is acyclic, and

hence F_b restricted to $V-B \subseteq V-A$ is acyclic, so there exists some $Y \in V-B$ which is F_b-undominated in $V-B$. By Lemma 10, Y is strongly b-undominated.

Proof of Proposition 22 By Definition 14, we have to show that for all $A \subseteq V$ and $a \in \underline{A}$, there exists a strongly a-undominated variable (see Definition 13). Let α be any outcome extending a. Then $\rhd_\alpha$ is irreflexive, since Γ is cuc-acyclic. This implies that $\rhd_a$ is irreflexive (since $Y \rhd_a Z$ implies $Y \rhd_\alpha Z$). So, $\rhd_a$ is acyclic. This implies that there exists some $Y \in V-A$ which is $\rhd_a$-undominated in $V-A$. If $\varphi \in \Gamma$ is such that a is compatible with u_φ, and $Y \in \{X_\varphi\} \cup W_\varphi$ then for all $Z \in U_\varphi$, $Z \rhd_a Y$ (by definition of $\rhd_a$) so $U_\varphi \subseteq A$ (else Y wouldn't be $\rhd_a$-undominated in $V-A$), which implies also that a extends u_φ. If $W_\varphi \ni Y$, then $X_\varphi \rhd_a Y$ and so $X_\varphi \in A$, proving that Y is a-undominated.

We will use the following basic technical lemma to prove Lemma 14.

Lemma 22 *Let A be a subset of V, and let $a \in \underline{A}$ be an assignment to the variables A, let $\alpha \in \underline{V}$ be an outcome such that $\alpha \models a$. Y is undominated in $V-A$ with respect to $\rhd_a$ if and only if Y is undominated in $V-A$ with respect to $\rhd_\alpha$. Furthermore, if Y is undominated in $V-A$ with respect to $\rhd_a$, then the relations $\succ_a^Y$ and $\succ_\alpha^Y$ (see Definition 4) are equal.*

Proof: If Y is undominated in $V-A$ with respect to $\rhd_\alpha$ then clearly Y is undominated in $V-A$ with respect to $\rhd_a$, since the relation $\rhd_\alpha$ extends $\rhd_a$. Conversely, suppose that Y is undominated in $V-A$ with respect to $\rhd_a$ and, to prove a contradiction suppose that Y is not undominated in $V-A$ with respect to $\rhd_\alpha$, so that $Z \rhd_\alpha Y$ for some $Z \in V-A$. Then there exists $\varphi \in \Gamma$ with either (i) $U_\varphi \ni Z$ and $Y \in \{X_\varphi\} \cup W_\varphi$ or (ii) $X_\varphi = Z$ and $W_\varphi \ni Y$ and $\alpha \models u_\varphi$. We have $U_\varphi \subseteq A$ since Y is undominated in $V-A$ with respect to $\rhd_a$. Since $Z \notin A$, we have $U_\varphi \not\ni Z$ so case (i) cannot hold. Consider case (ii). We have $\alpha \models u_\varphi$ and so $a \models u_\varphi$, since $U_\varphi \subseteq A$ and $\alpha \models a$. This implies by Definition 15 that $Z \rhd_a Y$ which is a contradiction of Z being in $V-A$, as Y is $\rhd_a$-undominated in $V-A$.

For the last part, $\succ_\alpha^Y$ clearly contains $\succ_a^Y$. We also have that all the parents of Y are in A: $U_Y \subseteq A$, since any element of U_Y dominates Y with respect to $\rhd_a$ (or $\rhd_\alpha$). This implies that $\succ_a^Y$ and $\succ_\alpha^Y$ are equal, since if $Y = X_\varphi$ for some $\varphi \in \Gamma$ with $\alpha \models u_\varphi$, then $a \models u_\varphi$.

Proof of Lemma 12 We first prove that if Y is $\rhd_a$-undominated in $V-A$ then Y is strongly a-undominated in $V-A$. Suppose that Y is $\rhd_a$-undominated and consider any $\varphi \in \Gamma$ such that u_φ is compatible with a. (i) If $X_\varphi = Y$ then $U_\varphi \cap (V-A) = \emptyset$, so $U_\varphi \subseteq A$. Also, (ii) if $W_\varphi \ni Y$ then $(U_\varphi \cup \{X_\varphi\}) \cap (V-A) = \emptyset$, so $U_\varphi \cup \{X_\varphi\} \subseteq A$. This proves that Y is strongly a-undominated in $V-A$.

If a cs-tree σ cu-satisfies Γ then each body node of it cu-satisfies Γ. This implies, using the result above, that each body node strongly satisfies Γ (see Definitions 17 and 13), implying that σ strongly satisfies Γ, and hence, by Proposition 19, σ satisfies Γ.

Proof of Lemma 14 Suppose $\alpha \gg_\Gamma \beta$ and let σ be a cs-tree cu-satisfying Γ. Let $r = \langle A, a, Y, \succ \rangle$ be the node that divides α and β. In order to prove the required condition $\alpha >_\sigma \beta$ we need to show that $\alpha(Y) \succ \beta(Y)$. Now, α and β differ on Y so $Y \in \Delta(\alpha, \beta)$. Y is $\rhd_a$-undominated in $V - A$, so by Lemma 22 is undominated in $V - A$ with respect to $\rhd_\alpha$, and hence we have $Y \in \Theta'(\alpha, \beta)$. Since $\alpha \gg_\Gamma \beta$ we have $\alpha(Y) \succ_\alpha^Y \beta(Y)$, which implies that $\alpha(Y) \succ_a^Y \beta(Y)$ by Lemma 22. This implies, by the definition of a body node cu-satisfying Γ, that $\alpha(Y) \succ \beta(Y)$ as required.

Proof of Lemma 15 If $\alpha = \beta$ then the result follows from Lemma 13, since $>_\sigma$ is irreflexive for any cs-tree σ.

Now, suppose that $\alpha \neq \beta$. This implies $\Delta(\alpha, \beta) \neq \emptyset$, and $\Theta'(\alpha, \beta) \neq \emptyset$. Since it is not the case that $\alpha \gg_\Gamma \beta$ there must exist some $Y \in \Theta'(\alpha, \beta)$ such that it is not the case that $\alpha(Y) \succ_\alpha^Y \beta(Y)$. Since, $Y \in \Theta'(\alpha, \beta)$, we also have $\alpha(Y) \neq \beta(Y)$. List the variables in an order compatible with $\rhd_\alpha$ such that Y appears as early as possible, i.e., so that there's no order compatible with $\rhd_\alpha$ in which Y appears earlier. We will construct a cs-tree σ in which the variables on the path from the root to α appear in that order. Thus, at any node r on the path to α, Y_r will be $\rhd_\alpha$-undominated in $V - A_r$, since any dominating variables come earlier, as the ordering is compatible with $\rhd_\alpha$.

For the node r' on the path to α with $Y_{r'} = Y$, we define relation $\succ_{r'}$ to be any total order on $\underline{Y}$ extending $\succ_\alpha^Y$ and such that $\alpha(Y) \not\succ_{r'} \beta(Y)$; this is possible, by Lemma 1(i), because $\succ_\alpha^Y$ is a partial order such that $\alpha(Y) \not\succ_\alpha^Y \beta(Y)$. For the other nodes r in the path to α we define the relation $\succ_r$ to be any total order extending $\succ_{a_r}^{Y_r}$.

We generate the rest of the cs-tree iteratively so that each body node r cu-satisfies Γ (by choosing any variable Y_r which is $\rhd_{a_r}$-undominated in $V - A_r$ and choosing any total order $\succ$ on $\underline{Y_r}$ extending $\succ_{a_r}^{Y_r}$). This generates a cs-tree σ which cu-satisfies Γ.

The condition that Y appears as early as possible in the variable ordering ensures that $A_{r'}$ consists only of variables Z with $Z \rhd_\alpha Y$. (If not, let Z be the latest variable appearing in the list before Y *not* satisfying $Z \rhd_\alpha Y$. Then moving Z to just after Y gives an ordering of the variables which is still compatible with $\rhd_\alpha$, but where Y comes earlier, which contradicts the definition of the variable ordering.) $Y \in \Theta'(\alpha, \beta)$ then implies $A_{r'} \cap \Delta(\alpha, \beta) = \emptyset$, i.e., α and β agree on all variables in $A_{r'}$. Then the node r' divides α and β. We also have $\alpha(Y) \not\succ_{r'} \beta(Y)$ which implies it is not the case that $\alpha >_\sigma \beta$.

Proof of Proposition 24 If Γ is not strongly conditionally conditionally acyclic then either it is not locally consistent, or there exists a proper subset A of V, and an assignment $a \in \underline{A}$ such that there exists no strongly a-undominated variable. Thus, to determine that Γ is not strongly conditionally acyclic we can either non-deterministically choose α and X such that $\succ_\alpha^X$ (which can be computed in polynomial time) is not irreflexive (proving that Γ is not locally consistent), or non-deterministically choose A and $a \in \underline{A}$ such that there exists no

strongly a-undominated variable (and for each variable $Y \in V - A$ we can check in polynomial time that Y is not a-undominated). This proves membership in coNP.

The proof of membership for cuc-acyclicity is similar, using the fact that for given cp-theory Γ and outcome α, computing the relation $J_\alpha(\Gamma)$—and hence determining that it is not acyclic—can be done in polynomial time.

To show coNP-hardness we use a reduction from 3-SAT. Consider an instance of 3-SAT with m clauses involving propositional variables V'. For $k = 1, \ldots, m$, let c^k be the k^{th} clause, which we write as $l_1^k \vee l_2^k \vee l_3^k$. We generate a cp-theory Γ as follows: let V be $V' \cup \{Z_0, Z_1, \ldots, Z_m\}$ where, for $k = 0, \ldots, m$, Z_k has domain $\{z_k, z'_k\}$. Let Γ_k consist of the three statements $l_j^k : z_{k-1} > z'_{k-1}\,[\{Z_k\}]$, for $j = 1, 2, 3$, and let Γ be $\Gamma_1 \cup \cdots \cup \Gamma_m \cup \{\top : z_m > z'_m\,[Z_0]\}$. Now, the 3-SAT instance has a satisfying assignment u if and only if there exists no u-undominated variable in $V - V'$, which implies that Γ is not strongly conditional acyclic.

Conversely, if Γ is not strongly conditional acyclic, then, for some proper subset A of V and tuple $a \in \underline{A}$, there exists no a-undominated variable in $V - A$ (since Γ is clearly locally consistent). This can only happen if A contains V' and for each k there exists j with a satisfying l_j^k, which is if and only if a satisfies the 3-SAT instance.

Hence the 3-SAT instance is satisfiable if and only if Γ is not strongly conditionally acyclic. The same construction applies also for cuc-acyclicity.

Proof of Proposition 25 Let σ be a cs-tree; we will define a VON τ compatible with σ. For each body node r of σ with associated tuple $\langle A, a, Y, \succ \rangle$ we define a node $\overline{r}$ of τ with associated tuple $\langle A, a, Y \rangle$. We create a sink node $\overline{r_*}$ with a_{r_*} equal to some arbitrary outcome. For each edge from node r to node r' in σ we create and edge from $\overline{r}$ to node $\overline{r'}$ in τ, where $\overline{r'}$ is defined to be the sink node if r' is a leaf node of σ. Clearly, τ is compatible with σ. (Naturally, there will often be much more compact variable ordering networks compatible with σ than τ.)

Conversely, suppose that τ is a variable ordering network. We construct a cs-tree σ compatible with τ inductively, starting from the root. To generate a body node r we need to define the associated tuple $\langle A, a, Y, \succ \rangle$. Assume, inductively, that A and a have already been defined (for the root node they are, by definition, equal to $\emptyset$ and $\top$, respectively). a generates a path in τ from the root node to some node $\overline{r}$. We let Y equal $Y_{\overline{r}}$, and we choose $\succ$ to be some arbitrary strict total order on $\underline{Y}$. We create $|\underline{Y}|$ child nodes of r, so r has $|\underline{Y}|$ edges coming from it. Each such edge e has associated variable $Y_e = Y$ and a different associated value y_e. If e goes from node r to r' then $A_{r'} = A_r \cup \{Y\}$, and $a_{r'}$ is the tuple formed by extending a with the assignment $Y_e = y_e$. This inductively defines the whole of σ. Clearly, σ is compatible with τ.

Proof of Lemma 16 By definition, $\langle t, Z, Y \rangle$ is in $\mathcal{T}_{ab}$ if and only if there exists assignment s to some set of variables S with s compatible with ab, $S - (A \cup B) =$

T, $s(T) = t$ and $\langle s, Z, Y\rangle \in \mathcal{T}$.

Given such an assignment s, define $U = S - A$ and $u = s(U)$. Then (i) $U - B = T$ and $u(T) = s(T) = t$ and u is compatible with b, since s is compatible with b; also we have (ii) $\langle u, Z, Y\rangle$ is in $\mathcal{T}_a$, since s is compatible with a.

Conversely, suppose there exists assignment u to some set of variables $U \subseteq V - A$ such that (i) $U - B = T$, $u(T) = t$, and u compatible with b and (ii) $\langle u, Z, Y\rangle$ is in $\mathcal{T}_a$. Hence, there exists assignment s (to some set of variables S) which is compatible with a, is such that $S - A = U$ and $s(U) = u$, and $\langle s, Z, Y\rangle \in \mathcal{T}$. We have $S - (A \cup B) = (S - A) - B = U - B = T$. Also, $T \subseteq U$, so $s(T) = u(T)$, which equals t. Since A and B are disjoint, $S \cap B = (S - A) \cap B = U \cap B$. Tuple s is compatible with b since $s(S \cap B) = s(U \cap B) = u(U \cap B) = b(U \cap B) = b(S \cap B)$, as u is compatible with b. Therefore, s is compatible with ab since it is also compatible with a. This proves that $\langle t, Z, Y\rangle$ is in $\mathcal{T}_{ab}$.

Proof of Lemma 18 This will be proved by induction on the cardinality of $|A_r|$. Suppose that it is true for all nodes r' with $|A_{r'}| < k$. Consider node r with $|A_r| = k$, and consider any path from the root to r with associated assignment a. Let r' be the parent of r along that path, and let a' be $a(A_{r'})$, and let $Y = y$ be the assignment along the edge between r' and r. By induction we have $\mathcal{T}_{a'} = \mathcal{T}_{a_{r'}}$. Using Lemma 17 and extending both a' and $a_{r'}$ by the assignment $Y = y$ we obtain $\mathcal{T}_a = \mathcal{T}_{a''}$ where a'' is $a_{r'}$ extended with the assignment $Y = y$. We also have by condition (ii) of Definition 23 that $\mathcal{T}_{a''} = \mathcal{T}_{a_r}$. Hence, $\mathcal{T}_a = \mathcal{T}_{a_r}$ as required.

Proof of Proposition 26 *Proof:* Consider some element $\langle u, Y, Z\rangle$ in $\mathcal{T}$, where u is an assignment to some set of variables U. Consider any outcome α extending u. We need to show that Y appears before Z in $\mathcal{O}_\tau(\alpha)$. We proceed using Proof by Contradiction.

Suppose that Z appears before Y in $\mathcal{O}_\tau(\alpha)$. Consider node r in τ with $Y_r = Z$ in the path associated with α. Since $Z = Y_r \notin A_r$ we also have $Y \notin A_r$, since Z appears before Y in the path associated with α, by definition of $\mathcal{O}_\tau(\alpha)$. Let $a = \alpha(A_r)$. By Lemma 18 we have $\mathcal{T}_a = \mathcal{T}_{a_r}$. Now, u is compatible with a, since α extends both u and a. Let $u' = u(U - A_r)$. Since $Y, Z \notin A_r$, we have $\langle u', Y, Z\rangle \in \mathcal{T}_a$, and so $\langle u', Y, Z\rangle \in \mathcal{T}_{a_r}$, i.e., $\langle u', Y, Y_r\rangle \in \mathcal{T}_{a_r}$, and hence there exists $\langle u'', Y, Y_r\rangle \in \mathcal{T}$ with u'' compatible with a_r (and u'' extending u'). By condition (i) of Definition 23, $Y \in A_r$ which contradicts $Y \notin A_r$, completing the Proof by Contradiction.

Proof of Proposition 27 Suppose that a and a' agree on $Q_\mathcal{T}(A)$. Consider any $\langle u', Y, Z\rangle$ in $\mathcal{T}_a$. Then, by Definition 21, there exists some triple $\langle u, Y, Z\rangle$ in $\mathcal{T}$ such that $Y, Z \in V - A$, tuple $u \in \underline{U}$ is compatible with a, and $u(U - A) = u'$. Now, $U \cap A \subseteq Q_\mathcal{T}(A)$, so a and a' agree on $U \cap A$, which implies that u is also compatible with a'. Hence, $\langle u', Y, Z\rangle$ in $\mathcal{T}_{a'}$, by Definition 21. We've shown that

$\mathcal{T}_a \subseteq \mathcal{T}_{a'}$. Swapping the roles of a and a' in the above argument shows that $\mathcal{T}_{a'} \subseteq \mathcal{T}_a$, so $\mathcal{T}_a = \mathcal{T}_{a'}$, as required.

Proof of Lemma 19 Y is not strongly a-undominated if and only if there exists $Z \in V - A$ and $\varphi \in \Gamma$ with u_φ compatible with a such that either (a) $Z \in U_\varphi$ and $Y \in X_\varphi \cup W_\varphi$ or (b) $Z = X_\varphi$ and $Y \in W_\varphi$. This is if and only if there exists $Z \in V - A$ and $\varphi \in \Gamma$ with u_φ compatible with a such that $\langle u_\varphi, Z, Y\rangle \in [\Gamma]$. This is if and only if there exists $Z \in V - A$ and u compatible with a such that $\langle u, Z, Y\rangle \in [\Gamma]$, which is if and only if there exists an element of the form $\langle u', Z, Y\rangle$ in $[\Gamma]_a$.

Proof of Lemma 20 Let r be a body node in σ with associated tuple $\langle A, a, Y, \succ\rangle$. Suppose that there exists some triple $\langle u, Z, Y\rangle$ in $[\Gamma]$ with $Z \in V - A$ and u compatible with a. Then, for any α extending both u and a, Z must appear before Y on the path to α in τ (since τ respects $[\Gamma]$, see Definition 22), and hence on the path to α in σ (since σ is compatible with τ), which contradicts the fact that $Z \in V - A$.

Therefore, there exists no triple $\langle u, Z, Y\rangle$ in $[\Gamma]$ with $Z \in V - A$ and u compatible with a. By Lemma 19, Y is strongly a-undominated. Therefore, by Definition 13 (Section 6.1), r strongly satisfies Γ, and hence σ strongly satisfies Γ, since r is an arbitrary body node of σ.

Proof of Lemma 21 Note that local consistency of Γ ensures that $\mathcal{Q}$ is non-empty, using the proof of Theorem 7. First suppose that $\alpha \gg^{\Gamma}_{\tau} \beta$, and let $r' = \langle A, a, Y\rangle$ be the node in τ that divides α and β. Consider any $\sigma \in \mathcal{Q}$, and let r be the node in σ which divides α and β. Because σ is compatible with τ, and so the variable ordering is the same on the path to α in τ and σ, we must have $A_r = A$ and $Y_r = Y$. Since $\alpha \gg^{\Gamma}_{\tau} \beta$ we have $\alpha(Y) \succ^{Y}_{a} \beta(Y)$, and so $\alpha(Y_r) \succ_r \beta(Y_r)$, and hence $\alpha >_\sigma \beta$.

Conversely, suppose that it is not the case that $\alpha \gg^{\Gamma}_{\tau} \beta$, and it will be shown that there exists some $\sigma \in \mathcal{Q}$ with $\alpha \not>_\sigma \beta$. If $\alpha = \beta$ then the implication holds because of irreflexivity of all $>_\sigma$, so we can assume that $\alpha \neq \beta$. Let $r' = \langle A, a, Y\rangle$ be the node in τ that divides α and β. Then $\alpha(Y) \not\succ^{Y}_{a} \beta(Y)$. By local consistency, $\succ^{Y}_{a}$ is acyclic, so there exists a total order $\succ$ on $\underline{Y}$ that extends $\succ^{Y}_{a}$ and is such that $\alpha(Y) \not\succ \beta(Y)$. Consider any cs-tree $\sigma \in \mathcal{Q}$. Let r be the node in σ which divides α and β. Because σ is compatible with τ, we must have $A_r = A$ and $Y_r = Y$. Change σ by changing $\succ_r$ to $\succ$. σ is still in $\mathcal{Q}$, and we have $\alpha(Y) \not\succ_r \beta(Y)$ which implies that $\alpha \not>_\sigma \beta$, as required.

References

Bienvenu, M., Lang, J., & Wilson, N. (2010). From preference logics to preference languages, and back. In *Proceedings of KR 2008*.

Boutilier, C., Bacchus, F., & Brafman, R. (2001). UCP-networks: A directed graphical representation of conditional utilities. In *Proceedings of UAI 2001*, pp. 56–64.

Boutilier, C., Brafman, R., Hoos, H., & Poole, D. (1999). Reasoning with conditional *ceteris paribus* preference statements. In *Proceedings of UAI 1999*, pp. 71–80.

Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H., & Poole, D. (2004a). CP-nets: A tool for reasoning with conditional *ceteris paribus* preference statements. *Journal of Artificial Intelligence Research*, *21*, 135–191.

Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H., & Poole, D. (2004b). Preference-based constrained optimization with CP-nets. *Computational Intelligence*, *20*(2), 137–157.

Brafman, R., & Domshlak, C. (2002). Introducing variable importance trade-offs into CP-nets. In *Proceedings of UAI 2002*, pp. 69–76.

Brafman, R., & Domshlak, C. (2008). Graphically structured value-function compilation. *Artificial Intelligence*, *172*, 325–349.

Brafman, R., Domshlak, C., & Shimony, E. (2006). On graphical modeling of preference and importance. *Journal of Artificial Intelligence Research*, *25*, 389–424.

Brafman, R. I., & Dimopoulos, Y. (2004). Extended semantics and optimization algorithms for CP-networks. *Computational Intelligence*, *20*(2), 218–245.

Brafman, R. I., Domshlak, C., & Shimony, S. E. (2004). Qualitative decision making in adaptive presentation of structured information. *ACM Trans. Inf. Syst.*, *22*(4), 503–539.

Domshlak, C. (2002). *Modeling and reasoning about preferences with CP-nets*. Ph.D. thesis, Ben-Gurion University of the Negev.

Domshlak, C., & Brafman, R. I. (2002). CP-nets—reasoning and consistency testing. In *Proceedings of KR 2002*, pp. 121–132.

Domshlak, C., Prestwich, S., Rossi, F., Venable, K. B., & Walsh, T. (2006). Hard and soft constraints for reasoning about qualitative conditional preferences. *Journal of Heuristics*, *12*(4–5), 263–285.

Domshlak, C., Rossi, F., Venable, K., & Walsh, T. (2003). Reasoning about soft constraints and conditional preferences: complexity results and approximation techniques. In *Proceedings of IJCAI 2003*, pp. 215–220.

Doyle, J., & Wellman, M. P. (1994). Representing preferences as *ceteris paribus* comparatives. In *Working Notes of the AAAI Symposium on Decision-Theoretic Planning*.

Freuder, E., Heffernan, R., Wallace, R., & Wilson, N. (2010). Lexicographically-ordered constraint satisfaction problems. *Constraints*, *171*(1), 3–25.

Goldsmith, J., Lang, J., Truszczyński, M., & Wilson, N. (2005). The computational complexity of dominance and consistency in CP-nets. In *Proceedings of IJCAI 2005*, pp. 144 –149.

Goldsmith, J., Lang, J., Truszczyński, M., & Wilson, N. (2008). The computational complexity of dominance and consistency in CP-nets. *Journal of Artificial Intelligence Research*, *33*, 403–432.

Hansson, S. O. (1996). What is ceteris paribus preference?. *Journal of Philosophical Logic*, *425*, 307–332.

Hansson, S. O. (2001a). Preference logic. In Gabbay, D., & Guenthner, F. (Eds.), *Handbook of Philosophical Logic*, pp. 319–393. Kluwer.

Hansson, S. O. (2001b). *The structure of values and norms*. Cambridge University Press.

Kaci, S., & Prade, H. (2007). Relaxing ceteris paribus preferences with partially ordered priorities. In *Proceedings of ECSQARU 2007*, pp. 660–671.

Lang, J. (2002). From preference representation to combinatorial vote. In *Proceedings of KR 2002*, pp. 277–288.

Lang, J. (2004). Logical preference representation and combinatorial vote. *Ann. Mathematics and Artificial Intelligence*, *42*(1), 37–71.

Marczewski, E. (1930). Sur l'extension de l'ordre partiel. *Fundamenta Mathematicae*, *16*, 386–389.

McGeachie, M., & Doyle, J. (2002). Efficient utility functions for ceteris paribus preferences. In *Proceedings of AAAI 2002*, pp. 279–284.

McGeachie, M., & Doyle, J. (2004). Utility functions for ceteris paribus preferences. *Computational Intelligence*, *20*(2), 158–217.

van Benthem, J., Girard, P., & Roy, O. (2009). Everything else being equal: A modal logic for ceteris paribus preferences. *Journal of Philosophical Logic*, *38*(1), 83–125.

von Wright, G. H. (1963). *The logic of preference*. Edinburgh University Press.

von Wright, G. H. (1972). The logic of preference reconsidered. *Theory and Decision*, *3*, 140–169.

Wilson, N. (2004a). Consistency and constrained optimisation for conditional preferences. In *Proceedings of ECAI 2004*, pp. 888–892.

Wilson, N. (2004b). Extending CP-nets with stronger conditional preference statements. In *Proceedings of AAAI 2004*, pp. 735–741.

Wilson, N. (2006). An efficient upper approximation for conditional preference. In *Proceedings of ECAI 2006*, pp. 472–476.

Wilson, N. (2009). Efficient inference for expressive comparative preference languages. In *Proceedings of IJCAI 2009*, pp. 961–966.

Xia, L., Conitzer, V., & Lang, J. (2008). Voting on multiattribute domains with cyclic preferential dependencies. In *Proceedings of AAAI 2008*, pp. 202–207.