Consistency and Constrained Optimisation for Conditional Preferences[1]

Nic Wilson[2]

Abstract. TCP-nets are an extension of CP-nets which allow the expression of conditional relative importance of pairs of variables. In this paper it is shown that a simple logic of conditional preferences can be used to express TCP-net orders, as well as being able to represent much stronger statements of importance than TCP-nets allow. The paper derives various sufficient conditions for a subset of the logical language to be consistent, and develops methods for finding a total order on outcomes which is consistent with the set of conditional preferences. This leads also to an approach to the problem of constrained optimisation.

1 INTRODUCTION

CP-nets [3, 1] is a formalism for compactly expressing conditional preferences on outcomes in multivariate problems. TCP-nets [5], augments CP-nets with a representation of importance relations between variables. They allow that under certain conditions, one variable X is more important than another variable Y, but under other conditions, Y is more important than X. Allowing such acyclicity on the variable ordering is desirable as it can occur in many natural situations, but it considerably complicates matters, as it is much harder to find and test sufficient conditions for consistency.

It has been shown (Wilson, 04) [8] that a simple logic of conditional preferences extends the representational power of CP-nets, whilst retaining some of their good properties, in particular sufficient conditions for consistency, based on a completely acyclic variable ordering. It is shown in section 3 that TCP-nets orders can also be expressed in a simple way in this logical language. Furthermore, the logical language allows much more powerful (though still very natural) statements of importance between variables.

In section 4, we consider the problem of generating sufficient conditions for a subset of the language to be consistent, which are still general enough to allow acyclicities in the variable ordering. A set of conditional preference statements Γ is consistent if and only if there exists a total order on outcomes which is compatible with (i.e., extends) the partial order $>_\Gamma$ on outcomes associated with Γ. We approach consistency by focusing on a special kind of total order, that associated with a data structure called a *search tree*. Search tree orders have nice properties: it is very easy to generate the first k elements in the order, and checking which of two outcomes is preferred can be done very easily, as the order is a kind of lexicographic order. We define conditions which imply that there exists a search tree order that extends $>_\Gamma$ and hence that Γ is consistent. We also give methods for constructing such a search tree, and a compact representation of search trees that can allow the consistency checking to be performed much more efficiently. Therefore the approach gives also a method of generating outcomes in a total order compatible with $>_\Gamma$.

Under certain conditions, a partial ordering $\gg_\Gamma$ which extends $>_\Gamma$ can be compactly defined (section 5). This leads to an incomplete, but more generally feasible approach to the problem of constrained optimisation, by generating all the $\gg_\Gamma$-maximal solutions of a constraint satisfaction problem, which are therefore all also $>_\Gamma$-maximal solutions.

2 A LOGIC OF CONDITIONAL PREFERENCES

This section defines and gives some basic properties of the simple logic of conditional preferences introduced in [8], and strongly related to a system defined in (Lang, 02) [7].

Let V be a set of variables. For each $X \in V$ let $\underline{X}$ be the set of possible values of X. For subset of variables $U \subseteq V$ let $\underline{U} = \prod_{X\in U} \underline{X}$ be the set of possible assignments to set of variables U. The assignment to the empty set of variables is written $\top$. A *complete tuple* or *outcome* is an element of $\underline{V}$, i.e., an assignment to all the variables. For partial tuples $a \in \underline{A}$ and $u \in \underline{U}$, we may write $a \models u$ if $A \supseteq U$ and $a(U) = u$, i.e., a projected to U gives u. More generally we say that a *is compatible with* u if there exists outcome $\alpha \in \underline{V}$ with $\alpha \models a$ and $\alpha \models u$, i.e., $\alpha(A) = a$ and $\alpha(U) = u$.

The language $\mathcal{L}_V$ (abbreviated to $\mathcal{L}$) consists of statements of the form $u : x > x' \, [W]$ where u is an assignment to set of variables $U \subseteq V$ (i.e., $u \in \underline{U}$), x, x' are different values of variable X, and $\{X\}$, U and W are pairwise disjoint. Let $T = V - (\{X\} \cup U \cup W)$. Such a conditional preference statement is intended to represent that given u and any assignment to T, x is preferred to x' irrespective of the values of W. CP-nets can be represented by a set of statements of the form $u : x > x' \, [W]$ with $W = \emptyset$ (see [8]). In the next section it is shown that TCP-nets can be represented in terms of such statements with $W = \emptyset$ or $|W| = 1$. 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 $T_\varphi = T$. Let φ^* be the set of pairs of outcomes $\{(tuxw, tux'w') : t \in \underline{T}, w, w' \in \underline{W}\}$. 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^*.

Subsets Γ of the language $\mathcal{L}$ are called *conditional preference theories (CP-theories)*. For conditional preference theory Γ, define $\Gamma^* = \bigcup_{\varphi\in\Gamma} \varphi^*$, which represents a set of preferences. If $(\alpha, \beta) \in \Gamma^*$ we say that β *is a worsening swap from* α. We assume here that preferences are transitive, so it is then natural to define the associated order $>_\Gamma$, induced on $\underline{V}$ by Γ, to be the transitive closure of Γ^*. So α is preferred to β, i.e., $\alpha >_\Gamma \beta$, if and only if there is a sequence

[1] This work has received support from Science Foundation Ireland under Grant 00/PI.1/C075

[2] Cork Constraint Computation Centre, Department of Computer Science, University College Cork, Cork, Ireland, n.wilson@4c.ucc.ie

of worsening swaps from α to β (see [8]). CP-theory Γ is said to be consistent if there exists a strict total order $>$ satisfying it, i.e., such that $\alpha > \beta$ for all outcomes α and β with $\alpha >_\Gamma \beta$. Γ is consistent if and only if $>_\Gamma$ is irreflexive.[3]

We will associate with $\Gamma \subseteq \mathcal{L}$ a binary relation $H(\Gamma)$ on the set of variables V. Let $H(\varphi) = \{(Y, X_\varphi) : Y \in U_\varphi\}$, and define $H(\Gamma) = \bigcup_{\varphi \in \Gamma} H(\varphi)$. For a fixed Γ we will sometimes write the parents of X in $H(\Gamma)$ as U_X, so $Y \in U_X$ if and only if $(Y, X) \in H(\Gamma)$.

Example 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 a plane ($\mathbf{p}$), or drive and take a car ferry ($\overline{\mathbf{p}}$). I also have to decide whether to take my good camera $\mathbf{g}$, or my cheaper one $\overline{\mathbf{g}}$. 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 $\underline{X_1} = \{\mathbf{n}, \overline{\mathbf{n}}\}$, $\underline{X_2} = \{\mathbf{o}, \overline{\mathbf{o}}\}$, $\underline{X_3} = \{\mathbf{p}, \overline{\mathbf{p}}\}$ and $\underline{X_4} = \{\mathbf{g}, \overline{\mathbf{g}}\}$.

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 is represented by the following preference statement: $\top : \mathbf{n} > \overline{\mathbf{n}}\ [\{X_2, X_3, X_4\}]$. This implies that outcome α is preferred to β whenever $\alpha(X_1) = \mathbf{n}$ and $\beta(X_1) = \overline{\mathbf{n}}$, 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 TCP-net. Whenever I travel I'd prefer to go to Oxford than to Manchester. If I go next week I definitely want to fly, as I can't face the long drive, which is represented by $\mathbf{n} : \mathbf{p} > \overline{\mathbf{p}}\ [\{X_2, X_4\}]$. Also $\mathbf{n} : \mathbf{o} > \overline{\mathbf{o}}\ [\{X_4\}]$. 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). On the other hand, later in the year my preference for Oxford is irrespective of how I travel: $\overline{\mathbf{n}} : \mathbf{o} > \overline{\mathbf{o}}\ [\{X_3, X_4\}]$. If I go later, I'd prefer to drive than fly, (whether I go to Oxford or Manchester) $\overline{\mathbf{n}} : \overline{\mathbf{p}} > \mathbf{p}\ [\{X_4\}]$, 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: $\mathbf{p} : \overline{\mathbf{g}} > \mathbf{g}\ [\emptyset]$ and $\overline{\mathbf{p}} : \mathbf{g} > \overline{\mathbf{g}}\ [\emptyset]$.

Let Γ be this set of preference statements. Let $\alpha = \mathbf{n}\overline{\mathbf{o}}\overline{\mathbf{p}}\overline{\mathbf{g}}$ and $\beta = \overline{\mathbf{n}}\mathbf{o}\overline{\mathbf{p}}\mathbf{g}$, and let φ be the first preference statement, ($\top : \mathbf{n} > \overline{\mathbf{n}}\ [\{X_2, X_3, X_4\}]$). Then $(\alpha, \beta) \in \varphi^*$ since $\alpha(X_1) = \mathbf{n}$ and $\beta(X_1) = \overline{\mathbf{n}}$. 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 $>_\Gamma$, so there does not exist outcome γ with $\alpha >_\Gamma \gamma >_\Gamma \beta$.[4] Also, in this case, $>_\Gamma$ is a total order on the set of outcomes.

3 REPRESENTING TCP-NETS WITHIN THE LANGUAGE

A TCP-net on set of variables V consists of a 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 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 total order $\succ_u^X$ on $\underline{X}$, i.e., it totally 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.

A total 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 $\succ_u^X$; total 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, total order $>$ is said to satisfy an i-arc $X \to Y$ if $rxy > rx'y'$ for all x, x' such that $x \succ_{r(U_X)}^X x'$ and y, y' such that $y' \succ_{r(U_Y)}^Y y$ and all assignments r to $V - \{X, Y\}$.

Given a TCP-net N, total order $>$ satisfies ci-statement $X \to_s Y$ if $rsxy > rsx'y'$ 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' such that $y' \succ_v^Y y$ where u is rs restricted to U_X and v is rs restricted to U_Y.

Total 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 total orders $>$ satisfying N.

Representing TCP-nets as 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 $\Gamma_{\text{cp}} \subseteq \mathcal{L}$ 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-arc $X \to_s Y$ define $\Gamma_{X \to_s Y}$ 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 Γ_{ci} be the union of $\Gamma_{X \to_s Y}$ over all ci-arcs $X \to_s Y$ of N.

Finally, define the CP-theory Γ_N to be $\Gamma_{\text{cp}} \cup \Gamma_{\text{i}} \cup \Gamma_{\text{ci}}$. These definitions easily lead to the following result, once one notices that the condition *all y, y' such that $y' \succ_v^Y y$* (in the definition of $>$ satisfying ci-statement $X \to_s Y$) can be replaced by just *all $y, y' \in \underline{Y}$* (and similarly for the corresponding condition for i-arcs). The point is that since $\succ_v^Y$ is a total order, if we don't have $y' \succ_v^Y y$ then we have either (i) $y' = y$ or (ii) $y \succ_v^Y y'$. If (i) then $rsxy > rsx'y'$ follows for $>$ satisfying N because $x \succ_u^X x'$; and if (ii) then $rsxy > rsx'y > rsx'y'$ follows for $>$ satisfying N because $x \succ_u^X x'$ and $y \succ_v^Y y'$.

Proposition 1 *TCP-net N is satisfiable if and only if Γ_N is consistent. If N is satisfiable, then $>_N = >_{\Gamma_N}$.*

This means that the logic of conditional preferences described in section 2 is more general than TCP-nets. 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

[3] Relation $>$ on set A is said to be *irreflexive* if and only if for all $a \in A$, it is not the case that $a > a$. It is acyclic if and only if its transitive closure is irreflexive, so that there are no cycles $a > a' > a'' > \cdots > a$. Binary relation $>$ on set $\underline{V}$ is defined to be a subset of $\underline{V} \times \underline{V}$, and the notations "$(\alpha, \beta) \in >$" and "$\alpha > \beta$" are used interchangeably.

[4] Since α and β differ on three variables, there exists no TCP-net N on V with $>_N = >_\Gamma$. This follows from the definition of a flipping sequence and lemma 5 in [5] (and also from proposition 1) since, being consecutive, they would have to be a single flip apart, and so differ at most on two variables.

this makes: how much is lost by approximating a statement $\varphi = (u : x > x' [W])$ by a set Γ 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 Γ^*, which represent the direct consequences of the conditional preference statements. For example, with all binary variables, $|\Gamma^*|/|\varphi^*| = (k+1)2^{-k}$, 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.

4 ENSURING CONSISTENCY

In [8], it was shown that CP-theory Γ is consistent as long as a local consistency property holds (see below) and the ordering of variables in Γ is completely acyclic, i.e., there exists some total order $\rhd$ on variables V such that for all $\varphi \in \Gamma$, if $Y \in U_\varphi$ then $Y \rhd X_\varphi$ and if $Z \in W_\varphi$ then $X_\varphi \rhd Z$. Because the importance ordering of variables can depend, in some natural situations, on the values of other variables, we want to weaken these conditions: deriving much less stringent acyclicity conditions that are still sufficient for consistency.

A CP-theory is consistent if and only if there exists some total order $>$ satisfying it. We generate sufficient conditions for consistency by focusing on a special type of total order, those generated by a *search tree.*[5] If we can find a search tree satisfying Γ then Γ is consistent. Moreover, these search trees have nice computational properties: we can use the search tree order to efficiently generate outcomes in an order compatible with $>_\Gamma$, and we can easily determine which of two outcomes is better according to a search tree order.

Local consistency. In certain cases, it's clear that Γ is not consistent, by just looking at local conditions: if there's a sequence of worsening swaps starting and ending with the same outcome α, which just change the value of a single variable X. Fix $\Gamma \subseteq \mathcal{L}$, and consider variable $X \in V$ and assignment $a \in \underline{A}$ for some $A \subseteq V$. Say that pair (x, x') of values of X *is validated by* a if there exists some statement $(u : x > x' [W]) \in \Gamma$ with $a \models u$ (i.e., u is a projection of a). Define relation $\succ_a^X$ 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 for all outcomes $\alpha \in \underline{V}$ and $X \in V$, $\succ_\alpha^X$ is irreflexive. Local consistency is a necessary condition for consistency. It is easily seen that Γ is locally consistent if and only if for all $X \in V$ and $u \in \underline{U_X}$, $\succ_u^X$ is irreflexive (recall U_X is the parents of X with respect to $H(\Gamma)$).

As discussed in [8] checking local consistency will often be easy; in particular, when the sets U_X are all small (as in intended applications of (T)CP-nets) one can efficiently construct each relation $\succ_u^X$ explicitly.

4.1 Search trees and consistency

A search tree is a rooted directed tree with its $|\underline{V}|$ leaves corresponding to outcomes. Associated with each node e in the body of the tree is a variable Y_e, which is instantiated with a different value in each of the node's $|\underline{Y_e}|$ children, and also an ordering $\succ_e$ of the values of Y_e. So a directed arc 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. We also associate with e a set of variables A_e which is the set of all variables $Y_{e'}$ associated to nodes e' above e in the tree (i.e., on the path from e to the root), and an assignment a_e corresponding to the assignments made to these variables in the arcs between e and the root. The root node e^* has $A_{e^*} = \emptyset$ and $a_{e^*} = \top$, the assignment to the empty set of variables. More formally, define a *body node* e to be a tuple $\langle A_e, a_e, Y_e, \succ_e \rangle$, where $A_e \subseteq V$ is a set of variables, $a_e \in \underline{A_e}$ is an assignment to those variables, $Y_e \in V - A_e$ is another variable, and $\succ_e$ is a total order on the set $\underline{Y_e}$ of values of Y_e. Define a *leaf node* e to be a pair $\langle A_e, \alpha_e \rangle$, with $A_e = V$ and outcome $\alpha_e \in \underline{V}$.

Each search tree σ has an associated total order $>_\sigma$ on outcomes which can be defined as follows. Let $\alpha, \beta \in \underline{V}$ be outcomes, and let e be the deepest node (i.e., furthest from the root) which is both on the path between the root and α, and on the path between the root and β. Then $\alpha >_\sigma \beta$ if and only if $\alpha(Y_e) \succ_e \beta(Y_e)$. In other words, we compare two outcomes by considering the lowest node e which is above both of them, and use the ordering $\succ_e$ to compare them.

If we draw the search tree σ with the directed arcs pointing downwards from the root, and the arcs from a node e in the order $\succ_e$, 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, if we instantiate at each node e the best value according to $\succ_e$ first, and on backtracking, the values in the order $\succ_e$, (using a depth-first search) 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 search tree order.

Search trees satisfying CP-theory Γ. We are interested in search 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 W_φ, as X_φ is a more important variable. Furthermore, we want the conditioning variables U_φ to all appear before X_φ, as we may need to know the values of relevant parents of X_φ before we can decide which values of X_φ are preferred. 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 a-undominated 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 node $\langle A, a, Y, > \rangle$ is said to satisfy Γ if (i) Y is a-undominated and (ii) the ordering $\succ$ on $\underline{Y}$ is a completion of $\succ_a^Y$ (i.e., if $y \succ_a^Y y'$ then $y \succ y'$). A search tree is said to satisfy Γ if each body node in the search tree satisfies Γ.

Proposition 2 *If a search tree σ satisfies CP-theory Γ, then its associated order $>_\sigma$ satisfies Γ.*

We define different sufficient conditions for $\Gamma \subseteq \mathcal{L}$ to be consistent. We say that Γ is

— *weakly conditionally acyclic* if there exists a search tree satisfying Γ;
— *conditionally acyclic* if it is locally consistent and for all $A \subseteq V$ and $a \in \underline{A}$, there exists an a-undominated variable;

Suppose Γ is a conditionally acyclic CP-theory, and a is any assignment to any proper subset A of V. The definitions imply that there exists Y and total order $\succ$ on $\underline{Y}$ such that $\langle A, a, Y, \succ \rangle$ is a body node satisfying Γ. This means that it is very easy to construct a search tree satisfying a conditionally acyclic Γ: we start by picking a root node satisfying Γ, and we proceed inductively, choosing children satisfying Γ for each node already chosen.

We will also consider a stronger condition on the CP-theory, which is useful for constrained optimisation (see section 5). The essential difference from conditional acyclicity is that we insist that the orderings always respect the generalised parent relation, where Y is said to be a generalised parent of Z if there exists $\varphi \in \Gamma$ with $Y \in U_\varphi$ and $Z \in \{X_\varphi\} \cup W_\varphi$. Let $\alpha \in \underline{V}$ be an outcome. For $\varphi \in \Gamma$ define $G(\varphi)$ to be $\{(Y, Z) : Y \in U_\varphi, Z \in \{X_\varphi\} \cup W_\varphi\}$. If φ is such

[5] Search trees are also used, explicitly and implicitly, in previous work on CP-nets, e.g., [2, 1].

that $\alpha \models u_\varphi$ define $G_\alpha(\varphi)$ to be $G(\varphi) \cup \{(X_\varphi, Z) : Z \in W_\varphi\}$. For other φ, let $G_\alpha(\varphi) = G(\varphi)$. For $\Gamma \subseteq \mathcal{L}$, define binary relation $G_\alpha(\Gamma)$ (abbreviated to G_α) on V to be $\bigcup_{\varphi\in\Gamma} G_\alpha(\varphi)$. Define order $\rhd_\alpha$ on V to be the transitive closure of G_α. We say that CP-theory Γ is *strongly conditionally acyclic* if it is locally consistent and for each outcome $\alpha \in \underline{V}$, $G_\alpha(\Gamma)$ is acyclic, i.e., $\rhd_\alpha$ is irreflexive. The following lemma implies, by Proposition 2, that any of the three forms of conditional acyclicity is sufficient for Γ to be consistent.

Lemma 1 *If CP-theory Γ is strongly conditionally acyclic then it is conditionally acyclic. If Γ is conditionally acyclic then it is weakly conditionally acyclic.*

Generating a total order satisfying CP-theory Γ. The procedure above for generating a search tree from conditionally acyclic Γ, could usually, depending on the choices made, generate many different search 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 search 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 e we choose Y_e to be X_i with minimal i (among the choices). Similarly we define $\succ_e$ 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.

For some applications it is sufficient 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 explicitly have to construct search 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 search tree σ. 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 which are above both leaf node $\langle V, \alpha\rangle$ and leaf node $\langle V, \beta\rangle$.

However, there remains the difficulty of checking consistency. The methods (in particular, lemmas 2, 3 and 4) of [5] can be easily adapted for checking that Γ is strongly conditionally acyclic, and hence consistent. An alternative approach is given in the next section.

4.2 Variable ordering networks

If Γ happens to be conditionally acyclic then search tree $\sigma(\Gamma)$ will satisfy Γ. However, confirming the consistency of Γ by constructing a search 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 search trees satisfying Γ. The idea is to turn the search 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 search tree is choosing an a_e-undominated variable Y_e for each body node e, where a_e is the tuple of assignments to earlier variables; if there is such a variable for each body node, then we have shown that Γ is consistent, assuming local consistency. (Given local consistency, choosing the local ordering $\succ_e$ is never a problem.) Consider two nodes d and e with associated tuples a and a', which 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 implies that any consistent subtree below node d corresponds to a consistent subtree below e and vice versa. Hence we can make the same choices of a-undominated variables, in nodes below d, as for nodes below e, or, more neatly, we can *merge nodes d and e*. We will define a data structure that enables us to assert such equivalences.

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 $Z \in W_\varphi$. This set encodes the information necessary to determine which variable orderings are allowed. In particular if $\langle u, Y, Z\rangle \in [\Gamma]$ and u is satisfied then Y must come before Z in the variable ordering. Given partial tuple $a \in \underline{A}$, we construct set $[\Gamma]_a$ by considering only triples in $[\Gamma]$ with u compatible with a, and we restrict such triples to $V - A$. Define $[\Gamma]_a$ to be the set of all triples $\langle u', Y, Z\rangle$ with $Y, Z \in V - A$ and $u' \in \underline{U'}$, such that there exists triple $\langle u, Y, Z\rangle$ in $[\Gamma]$ with $u \in \underline{U}$ compatible with a and $u(U - A) = u'$.

It can be shown that Y is a-undominated if and only if there does not exist any triple of the form $\langle u', Z, Y\rangle$ in $[\Gamma]_a$. Furthermore, suppose $B \subseteq V - A$ and $b \in \underline{B}$. It can also be shown that $[\Gamma]_{ab}$ can be computed from $[\Gamma]_a$ and b. This implies that if $[\Gamma]_a = [\Gamma]_{a'}$ then $[\Gamma]_{ab} = [\Gamma]_{a'b}$. So Y is ab-undominated if and only if Y is $a'b$-undominated. Hence $[\Gamma]_a$ encodes the information needed to determine the c-undominated variables for any c extending a, and so can be used to determine equivalence of two nodes: that they can be merged.

The variable ordering network $\tau(\Gamma)$ for a conditionally acyclic CP-theory Γ consists of body nodes e which are triples $\langle A_e, a_e, Y_e\rangle$ with, as for search trees, $a_e \in \underline{A_e}$, and $Y_e \notin A_e$; there is a single leaf node e_* which equals $\langle A_{e_*}, \alpha\rangle$, where α is some outcome and $A_{e_*} = V$.

To construct $\tau(\Gamma)$ we first construct the (single) root node; then we iteratively construct the children of a node already constructed. As for search trees, for each node e we generate $|Y_e|$ directed arcs from e, each associated with some value of variable Y_e. Let a be a_e extended with assignment y to Y_e, and let $A = A_e \cup \{Y_e\}$. We construct $[\Gamma]_a$, and then we look for any other node d already constructed with $A_d = A$ and $[\Gamma]_{a_d} = [\Gamma]_a$ (this is the condition for equivalence discussed above). If there is such a node d then we add a directed arc from e to d with associated value y of Y_e. If there is no such node then 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 $V - A$ (conditional acyclicity ensures that there is such a variable).

Assume now that CP-theory Γ is locally consistent but not necessarily conditionally acyclic. We can apply the same construction but it is not guaranteed to succeed; the construction of $\tau(\Gamma)$ succeeds if and only if the construction of $\sigma(\Gamma)$ succeeds, because $\tau(\Gamma)$ is a compact representation of $\sigma(\Gamma)$ where we merge some nodes, and we throw away the local orderings $\succ_e$. We can therefore use $\tau(\Gamma)$ as a representation of the search tree ordering $>_{\sigma(\Gamma)}$, by defining the local orderings (on values of a variable Y_e, given a particular partial tuple a) when we need them. So a test for consistency of Γ is being able to construct variable ordering network $\tau(\Gamma)$: if this succeeds then Γ is consistent. Moreover, if Γ happens to be conditionally acyclic, then this test for consistency is bound to succeed.

The variable ordering network $\tau(\Gamma)$ for the example has only seven nodes, with at most two nodes at each level, as opposed to the 31 nodes in the corresponding search 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 search tree, the variable ordering network may still be small, and thus consistency efficiently checked.

5 CONSTRAINED OPTIMISATION

The maximally preferred outcomes of a CP-theory Γ are precisely the solutions of a particular constraint satisfaction problem (CSP) C_Γ on V (cf. [4]): 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 $(U_\varphi = u) \Rightarrow (X_\varphi \neq x')$. Furthermore, as observed in [8], if $H(\Gamma)$ is acyclic and (e.g.,) if Γ is locally consistent it's very easy to find $>_\Gamma$-maximal 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 harder (see [7]) when we have a set of constraints C on V, and we wish to find maximally preferred solutions to C. Let $\Omega \subseteq \underline{V}$ be the set of solutions of C. We say that α is $>_\Gamma$-maximal given C if it is $>_\Gamma$-maximal in Ω, i.e., there does not exist $\beta \in \Omega$ with $\beta >_\Gamma \alpha$. If one is only interested in finding *some* outcomes which are maximally preferred, then one can try to solve the CSP with constraints $C_\Gamma \cup C$. Any solution of this CSP will be $>_\Gamma$-maximal given C since it is $>_\Gamma$-maximal in $\underline{V}$. Of course $C_\Gamma \cup C$ may well have no solutions.

Suppose one can find a search tree σ satisfying Γ (for example, $\sigma(\Gamma)$, or using the associated variable ordering network $\tau(\Gamma)$). This can be used to generate the solutions of C in the order $>_\sigma$, by using the natural backtracking algorithm associated with σ (as in the approach in [2]). The first solution, α, that it generates will be $>_\Gamma$-maximal since $\beta >_\Gamma \alpha$ implies $\beta >_\sigma \alpha$ by Proposition 2. At each point in the search we have a set Ω^* of $>_\Gamma$-maximal 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 $>_\Gamma$-maximal solution and we add it to Ω^*. This is a complete algorithm, with the final Ω^* being the set of all $>_\Gamma$-maximal solutions, and at each point the set Ω^* contains only $>_\Gamma$-maximal solutions. The problem with this algorithm (and the similar algorithms in [2, 5]) is that determining if $\beta >_\Gamma \alpha$ (or not) will often be infeasible [1, 6] unless the problem is small, since it involves searching for swapping sequences, which generalise flipping sequences.

A different approach (generalising the approach for the fully acyclic case in [8]) is to find a partial order $\gg$ which contains $>_\Gamma$, and to use dominance testing with respect to $\gg$ (cf. the discussion of 'ordering queries', and the proof of Theorem 6 in [1]). Consider fixed strongly conditionally 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 maximal elements in $\Delta(\alpha, \beta)$ with respect to $\rhd_\alpha$. The binary relation $\gg_\Gamma$ on outcomes is defined by: $\alpha \gg_\Gamma \beta$ if and only if for all $X \in \Theta(\alpha, \beta)$, $\alpha(X) \succ_\alpha^X \beta(X)$.

Proposition 3 *Suppose CP-theory $\Gamma \subseteq \mathcal{L}$ is strongly conditionally acyclic. Then $\gg_\Gamma \supseteq >_\Gamma$, i.e., if $\alpha >_\Gamma \beta$ then $\alpha \gg_\Gamma \beta$.*

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. We can then adapt the algorithm described above for generating the $>_\Gamma$-maximal solutions by replacing each test $\beta >_\Gamma \alpha$ by the generally much simpler test $\beta \gg_\Gamma \alpha$. The final Ω^* will be all the $\gg_\Gamma$-maximal solutions. But all these will also be $>_\Gamma$-maximal solutions, since if $\beta >_\Gamma \alpha$ then $\beta \gg_\Gamma \alpha$. Note that Ω^* only ever contains $\gg_\Gamma$-maximal, and hence $>_\Gamma$-maximal, solutions.

This amended algorithm is an anytime algorithm for finding some of the $>_\Gamma$-maximal solutions, which will generally be much faster than the complete algorithm.

6 DISCUSSION

The simple conditional preference logic $\mathcal{L}$ is rich enough to express TCP-nets, as well as being able to express that one variable dominates (i.e., is much more important than) a set of variables, given a partial assignment to other variables. The pairwise importance relations of TCP-nets do not usually closely approximate this kind of dominance. The logic can also express 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. Indeed the logical language representation is very flexible as regards elicitation: we can reason with an arbitrary subset Γ of the language, 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, all of our previous deductions from Γ will still hold, in particular whether one outcome is preferred to another (however, the set of optimal solutions to a CSP may be reduced).

The sufficient conditions derived for consistency of Γ allow the importance between variables to be conditional and even, (except in the case of strong conditional acyclicity) a variable X may be a parent of another variable Y under some circumstances, but Y may be a parent of X under other circumstances. Checking these sufficient conditions, as for TCP-nets, looks to be hard in general. However, we imagine that in many real situations, even in complex problems, there will tend to be limited numbers of these conditional changes of importance. Confirming consistency of Γ should often then be feasible, either using the techniques derived for TCP-nets for checking strong conditional acyclicity of Γ, or using a variable ordering network to compactly represent a search tree satisfying Γ.

REFERENCES

[1] C. Boutilier, R. I. Brafman, C. Domshlak, H. Hoos, and D. Poole, 'CP-nets: A tool for reasoning with conditional *ceteris paribus* preference statements', *Journal of Artificial Intelligence Research*, **21**, (2004).

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

[3] C. Boutilier, R. I. Brafman, H. H. Hoos, and D. Poole, 'Reasoning with conditional ceteris paribus preference statements', in *Proc. UAI99*, pp. 71–80, (1999).

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

[5] R. I. Brafman and C. Domshlak, 'Introducing variable importance tradeoffs into CP-nets', in *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence (UAI-02)*. Morgan Kaufmann Publishers.

[6] R. I. Brafman and C. Domshlak, 'CP-nets—reasoning and consistency testing', in *Proc. KR02*, (2002).

[7] J. Lang, 'From preference representation to combinatorial vote', in *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR2002)*, pp. 277–288, (2002).

[8] N. Wilson, 'Extending CP-nets with stronger conditional preference statements', in *Proc. AAAI 2004*, (2004).