

Representing and Reasoning with Preferences: CP Nets

CMPT 411/721


Preferences

- Much of reasoning in AI is concerned with attaining some *goal* or satisfying some hard constraint.
- In commonsense reasoning one frequently *prefers* one outcome over another. E.g.:
 - Dining out, one may prefer fish to pasta, and if having fish, prefer white wine to red wine.
 - In flying from Vancouver to Vienna, prefer a cheap flight with good connections and minimal number of stops.
- In AI and CS, preferences are important in areas including recommender systems, e-commerce, planning and scheduling, multiagent systems, configuration, and other problems involving decision support or decision making.
- Preferences have also been studied in economics, psychology, operations research and other areas.

Preferences

Preferences can be characterised by various criteria:

- Qualitative vs. quantitative
 - Qualitative: Give an ordering over alternatives
 - Quantitative: Typically expressed in terms of a *utility* function
- “Soft constraints” vs. “soft goals”
 - I.e. preferences over constraints vs. preferences over the outcome of decisions.
 - E.g. constraints in scheduling vs. constraints in planning
- Underlying preference order
 - E.g. total preorder vs. partial preorder vs. total order

 Clearly approaches may combine aspects of the above (e.g. have qualitative and quantitative aspects).

Preference Structures

- Have a set S of alternatives or outcomes
 - Think of elements of S as possible states of the world.
- Specifying preferences on S = compare/rank alternatives.
- E.g. Could have $S = \{pq, p\neg q, \neg pq, \neg p\neg q\}$, and
 - *Qualitative preference:*
prefer pq over $p\neg q$ over $\neg pq$ over $\neg p\neg q$.
 - *Quantitative preference:*
 $Value(pq) = 8;$
 $Value(p\neg q) = 6;$
 $Value(\neg pq) = 2;$
 $Value(\neg p\neg q) = -1;$

Preference Structures (ctd)

Qualitative preferences

Preference relation on S : reflexive and transitive relation \preceq

- $x \preceq y$: y is at least as good as x
- $x \prec y$: y is preferred to x (strict preference)
= $x \preceq y$ and not $y \preceq x$
- $x \sim y$: x and y are equally preferred (indifference)
= $x \preceq y$ and $y \preceq x$

Preference Structures (ctd)

Quantitative preferences

Utility function $u : S \mapsto \mathcal{R}$

Other models:

Interval orders, fuzzy preferences etc.

👉 We'll stick with qualitative preferences.

Domain Structure

- Assume a set of *variables* (or features or attributes)

$$\mathbf{V} = \{X_1, \dots, X_n\}$$

over which the decision maker has preferences.

- Each variable X_i is associated with a *domain*

$$Dom(X_i) = \{x_1^i, \dots, x_{n_i}^i\}$$

of values it can take.

- An *assignment* \mathbf{x} of values to a set $\mathbf{X} \subseteq \mathbf{V}$ of variables (also called an *instantiation* of \mathbf{X}) is a function that maps each variable in \mathbf{X} to an element of its domain.
- If $\mathbf{X} = \mathbf{V}$, \mathbf{x} is a *complete assignment*; otherwise \mathbf{x} is a *partial assignment*.
 - For complete assignments, values of \mathbf{x} would correspond to possible states of the world.

Domain Structure: Example

Example

Preferences on airplane tickets

options = (destination, price, dates, number-changes)

- Combinatorial explosion: Get a prohibitive number of alternative.
- E.g.: 50 destinations, 10 price ranges, 10 departure dates and 10 return dates, 0/1/2 changes \mapsto 150,000 alternatives

Problem:

- 👉 Need for concise representations for preferences

Another Problem

- Preferences are often not independent. E.g.:
 - Prefer white wine to red if fish is served, but beer if it's hot.
 - If in Madrid, prefer to visit cultural sites, while if in Sydney prefer to visit beaches.
- So need to express *preferential dependencies*
- But representing preferences exhaustively under all possible combinations is unfeasible in practice.



Need languages for compact preference representation

CP Nets[Boutilier *et al.*, 2004]

- Ideally, a preference representation should
 - capture natural statements of preferences,
 - be compact, and
 - support efficient inference.
- *CP-nets* (for *Conditional Preference networks*) are
 - graphical models for
 - representing and reasoning about conditional qualitative preferences.

CP Nets: Introduction

- Structurally, a CP net resembles a Bayes net.
 - Also has analogous independence assumptions.
- A CP-net is composed of
 - A *graph* representing the preferential dependencies between variables
 - For each variable, a *conditional preference table*.
 - A conditional preference table expresses the local preference on the values of its domain, given the possible combination of values of its parents.

CP Nets: Intuitions

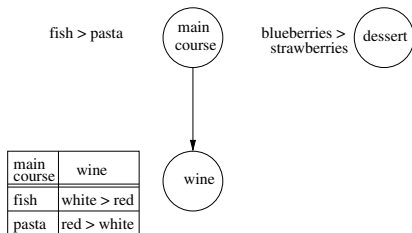
- Structurally a CP net is a directed graph, where nodes represent attributes of interest.
- For attribute A , the parents of A are those attributes that directly influence preference for values of A .
- Associated with each node is a table giving preferences of values of that node, given values for its parents.
- E.g., consider “I prefer red wine to white wine if pasta is served”
 - *Wine* will be a node with parent *Pasta*.
 - The table associated with *Wine* will assert that when *Pasta* is true that the meal with a red wine is preferred to one meal with a white wine.

Example

Consider the following:

- I (unconditionally) prefer a fish main meal to pasta.
- If fish is served I prefer white wine to red.
- If pasta is served I prefer red wine to white.
- For dessert I prefer blueberries to strawberries.

CP net:



CP Nets: Intuitions

The structure of a CP Net reflects a *ceteris paribus* assumption.

- I.e., under the ceteris paribus assumption,
“I prefer red wine to white wine if pasta is served”
asserts that,
 - for two meals with pasta that differ *only* in the kind of wine served,
 - a meal with red wine is preferred to a meal with white wine.
- This interpretation corresponds to a “least committing” interpretation of the information provided by the user.

Preferential Independence

- A set of variables \mathbf{X} is *preferentially independent* of $\mathbf{Y} = \mathbf{V} \setminus \mathbf{X}$ iff for all values x_1, x_2 for elements of \mathbf{X} and y_1, y_2 of \mathbf{Y} , we have

$$x_1 y_1 \succeq x_2 y_1 \text{ iff } x_1 y_2 \succeq x_2 y_2$$

- We say x_1 is preferred to x_2 *ceteris paribus*.

Conditional Preferential Independence

- Let \mathbf{X} , \mathbf{Y} , and \mathbf{Z} be nonempty and partition \mathbf{V} .

\mathbf{X} is *conditionally preferentially independent* of \mathbf{Y} given an assignment z to \mathbf{Z} iff for all values x_1, x_2 for elements of \mathbf{X} and y_1, y_2 of \mathbf{Y} , we have

$$x_1y_1z \succeq x_2y_1z \text{ iff } x_1y_2z \succeq x_2y_2z$$

- I.e., \mathbf{X} is preferentially independent of \mathbf{Y} when \mathbf{Z} is assigned z .

If \mathbf{X} is conditionally preferentially independent of \mathbf{Y} for all assignments z , then \mathbf{X} is *conditionally preferentially independent* of \mathbf{Y} given \mathbf{Z} .



Think of \mathbf{Z} as being a *context* in which \mathbf{X} is preferentially independent of \mathbf{Y} .

CP nets and Preferential Independence

- For each variable X_i , we identify a set of parent variables $Pa(X_i)$ that can affect preference over values of X_i .
- I.e. given an assignment to $Pa(X_i)$, one can determine a preference order for the values of X_i , all other things being equal.
- Formally, given $Pa(X_i)$, we have that X_i is conditionally preferentially independent of $\mathbf{V} \setminus (Pa(X_i) \cup \{X_i\})$.
 - This is the independence assumption underlying CP nets.
- We use this information to create a directed graph where nodes represent the problem variables, and every node X_i has $Pa(X_i)$ as its parents.
- Each node X_i is annotated with a conditional preference table (CPT) describing the preferences over the values of X_i given every combination of parent values.

CP Nets: Formal Definition

Definition

A *CP-net* over variables $\mathbf{V} = \{X_1, \dots, X_n\}$ is

- a directed graph G over X_1, \dots, X_n ,
- whose nodes are annotated with conditional preference tables $CPT(X_i)$ for each $X_i \in \mathbf{V}$.

Each conditional preference table $CPT(X_i)$ gives a total order \prec over values of X_i for each instantiation of X_i 's parents.

Example

Consider the following:

- I (unconditionally) prefer a fish main meal to pasta.
- If fish is served I prefer white wine to red.
- If pasta is served I prefer red wine to white.
- For dessert I prefer blueberries to strawberries.

Example: CP Net

fish > pasta



blueberries >
strawberries



main course	wine
fish	white > red
pasta	red > white



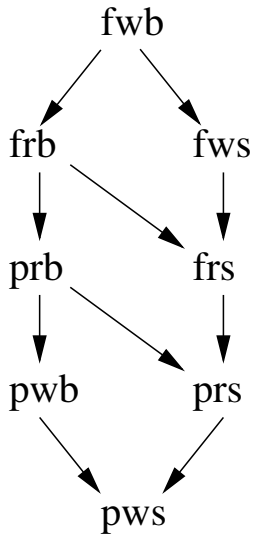
Preference Ordering

- The *ceteris paribus* interpretation implies that 2 variable assignments that differ for one variable (while the others remain the same) are always ordered.
- To compute the ordering over all variable assignments, use the notion of a *worsening flip*.
- A *worsening flip* is a change in the value of a single feature to a value which is less preferred according to a cp-statement for that feature.
 - E.g. $frb \mapsto prb$ is a worsening flip, since fish is preferred to pasta.
 - Similarly $fwb \mapsto frb$

Outcome Ordering

- A complete assignment to variables is called an *outcome*.
 - Again, think of an outcome as a possible state of the world.
- An outcome α is preferred to β , written $\alpha \succ \beta$, iff there is a chain of worsening flips from α to β .
- This defines a strict preorder over outcomes, called the *induced graph*.
- An outcome is optimal if there is no other outcome which is preferred to it.

Example: Induced Graph

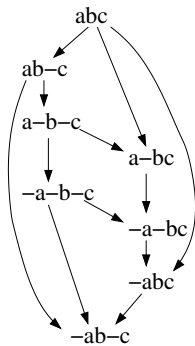
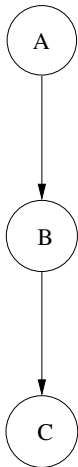


Another Example

$$a > -a$$

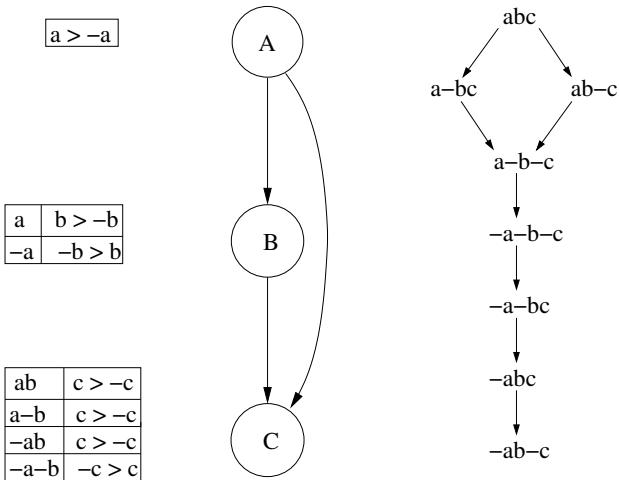
a	b > -b
-a	-b > b

b	c > -c
-b	-c > c



And Another Example

(Omitting transitive edges)



Properties

- A CP net is *satisfiable* iff there is a preference ranking \succ that satisfies it.

Theorem: Every acyclic CP-net is satisfiable.

- Finding an optimal outcome of a general CP net is NP-hard.
- Find an optimal outcome of an acyclic CP net is linear using the *forward sweep* algorithm:
 - ➡ Sweep through the CP-net, following the arrows in the dependency graph and assigning at each step the most preferred value in the current feature's preference table.
- So, an acyclic network determines a unique best outcome.

Properties

- Determining if one outcome is preferred to another (i.e. testing *dominance*) is PSPACE-complete for acyclic CP-nets.
 - Intuitively, to check whether an outcome is preferred to another one, one finds a chain of worsening flips;
 - these chains can be exponentially long.
- When a binary-valued CP-net forms a directed tree, the complexity of dominance testing is quadratic in the number of variables.
- Dominance testing for binary-valued CP-nets is NP-complete if the number of paths between any pair of nodes is polynomially bounded.

Extensions

The basic framework allows *cycles* in the preference graph (which we haven't considered here).

The basic framework has been extended in various ways

- The notion of *indifference* is addressed in the full approach.
- TCP nets allow for variables taking on differing *importance levels*.



C. Boutilier, R.I. Brafman, C. Domshlak, H. H. Hoos, and D. Poole.

CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements.

Journal of Artificial Intelligence Research, 21:135–191, 2004.