Preference Representation: Preference Orders, Languages and Graphical Models

Paolo Viappiani <paolo.viappiani@lip6.fr>

Preferences in AI Research

- **Preference modelling**: defining preference relations and their characteristics
- **Preference representation**: providing a language allowing the expression of preferences in a compact way
- **Preference reasoning**: making inference about preferences (perhaps of somebody else, or reasoning about a group of agents)
- **Preference elicitation**: assessing preferences with specific questions (often given in a precise form of a protocol)
- **Preference learning**: acquiring preference information from data (sometime learning and elicitation can be smilier)

Outline

Preference representation

Preference representation is the task of expressing the preferences of an underlying preference relation in a practical way.

- Impractical to use explicit orders (beside in very small outcome spaces)
- Cognitive difficulty in comparing complete instances with many attributes
- Languages for representing preferences allow to specify "patterns" in preferences, and support automated reasoning and manipulation of preferences.

In this lecture, we discuss methods for representing preferences:

- Oder theory and preference relations
- Short review on utility
- Languages for preference representations
  - Graphical languages: CP-nets, GAI-nets, UCP-nets
  - Logical languages

Review of basic properties of binary relations

A relation on a set A is a subset of $A \times A$; it is customary to write $a\ R\ b$ instead of $(a, b) \in R$

A relation R is said to be

- *total (or complete):* for all $a, b \quad a\ R\ b \vee b\ R\ a$.
- *transitive:* $a\ R\ b \wedge b\ R\ c \rightarrow a\ R\ c$
- *negative transitive:* $a \not R\ b \wedge b \not R\ c \rightarrow a \not R\ c$
- R is *reflexive:* $\forall a \in A\ a\ R\ a$
- R is *irreflexive:* $\forall a \in A\ a \not R\ a$
- *symmetric:* $a\ R\ b \rightarrow b\ R\ a$
- *antisymmetric:* $a\ R\ b \wedge b\ R\ a \rightarrow a = b$.
- *asymmetric:* $\nexists a, b : a\ R\ b \wedge b\ R\ a$ (antisymmetric and irreflexive)
- *Ferrers:* $a\ R\ b \wedge c\ R\ d \rightarrow a\ R\ d \vee c\ R\ b$

Preference Relations

The basic ingredient is a binary relation $\succeq$ meaning "at least as good as" (typically $\succeq$ is reflexive)

- An item a is strictly preferred to b:
  $a \succ b$ iff $a \succeq b \wedge b \not\succeq a$
- An item a is indifferent to b :
  $a \approx b$ iff $a \succeq b \wedge b \succeq a$
- An item a is incomparable with b, if neither a is preferred to b nor b is preferred to a:
  $a \sim b$ iff $a \not\succeq b \wedge b \not\succeq a$

$(\succ, \approx, \sim)$ is the *preference structure* induced by $\succeq$.

Note: by construction $\succ$ is asymmetric; $\approx$ and $\sim$ are symmetric relations.

Typical Assumptions

Typically $\succeq$ is reflexive (an item is as good as itself), then $\approx$ is reflexive and $\sim$ is irreflexive

$\succeq$ is usually transitive, then in this case $\succ$ as well is transitive

If $\succeq$ is complete then $\sim$ is empty

What an Order can Be?

If $\succeq$ is...

- A total pre-order (reflexivity, transitive and complete relation), then $\succ$ and $\approx$ are transitive; $\sim$ is empty.
- Total order (reflexivity, antisymmetric, transitive and complete relation): $\approx$ is the set of pairs (a, a) (only indifferent to itself)
- A partial pre-order (reflexive and transitive relation): both $\succ$ and $\approx$ are transitive and $\sim$ is not empty
- A partial order (reflexive, antisymmetric, transitive): $\approx$ is the set of pairs (a, a) (only indifferent to itself)

Transitive Complete Preferences

If $\sim$ is empty and both $\succ$ and $\approx$ are transitive:

1. If $a \succ b$ and $b \approx c$ then $a \succ c$.
2. If $a \approx b$ and $b \succ c$ then $a \succ c$.

Proof of (1)

By contradiction, assume $a \not\succ c$: then either $a \approx c$ or $c \succ a$.

- If $a \approx c$ then by transitivity of $\approx$ we deduce $a \approx b$ but this is inconsistent with the hypothesis $a \succ b$.
- Similarly, if instead $c \succ a$, by transitivity of $\succ$ we derive $c \succ b$, but this is inconsistent with the hypothesis.

Therefore we deduce that $a \succ c$.

Properties of Negative Transitivity

Definition : if $a \not\succ b$ and $b \not\succ c$ then $a \not\succ c$.

Thanks to NT, it holds:

1. If $a \succ b$ and $b \approx c$ then $a \succ c$.
2. If $a \approx b$ and $b \succ c$ then $a \succ c$.

Relation between $\succeq$ and $\succ$

$\succeq$ is a total preorder iff $\succ$ is negative transitive, $\approx$ is reflexive and $\sim$ is empty.

Proof: $\Longrightarrow$: the difficult point is negative transitivity. Assume $a \not\succ b$ and $b \not\succ c$, we want to prove $c \not\succ a$. Then, since $\succeq$ is total, it means $b \succeq a$ and $c \succeq b$. By transitivity of $\succeq$, we have $c \succeq a$. Therefore $c \not\succ a$.

$\Longleftarrow$: (skipped)

Other observations

- If $\succ$ is negative transitive, then $\succ$ is also transitive.
- If $\succ$ is negative transitive and the $\sim$ is empty, then $\approx$ is transitive.

What are the best items?

What Should We Recommend?

Given an order return the "best" item (choice)

$\max(\succ)$ the maximal items according to $\succeq$ (items that are not strictly preferred by another item)

$$\max(\succ) = \{o | \nexists o' : o' \succ o\}$$

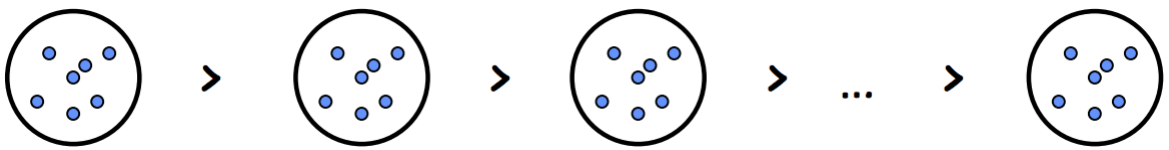
Similarly the "worst" items are the minimal elements (those not more preferred to any other).

Total Preorders

When $\succeq$ is total preorder, the induced indifference relation $\approx$ is an equivalence relation (reflexive, symmetric, transitive).

- We can define preferences in terms of an ordered partition $(E_1, \ldots, E_n)$ wrt $\succeq$
- no $E_1, \ldots, E_n$ is empty
- the union of $E_1, \ldots, E_n$ is the set X of all items
- $\forall o, o'; o \in E_i, o' \in E_j, i < j$ iff $o \succ o'$.
- o, o' both $\in E_i$ iff $o \approx o'$

E_1 is the set of "best" items, E_2 is the set of second-best items, etc.



Characterization of Total Preorders

A total preorder is complete, reflexive and transitive.

Characterisation of Preference Structure

Preference structures can be "characterised": showing the necessary and sufficient conditions for which actual preferences upon a set A happen to be one of these structures

Total Preorder

Numerical representation of the type:

$$x \succ y \leftrightarrow f(x) > f(y)$$

Not unique: all monotonic non decreasing transformations of the numerical representation are admissible.

Indifference

If the $\succ$ relation is antisymmetric, then $\succ$ is asymmetric and $\approx$ is the identity relation

- Total order: every item is indifferent to itself
- Weak order: $\approx$ is transitive
- What if indifference is not transitive ?
  cfr *Paradox of the heap* (or Sorites paradox; Eubulides of Miletus)

Let assume there exists a function f assigning numerical values to items (as in ordinal utility) and a threshold τ; define $\succeq$ such that:

$$x \succeq y \iff f(x) - f(y) \geq -\tau$$

and therefore $x \succ y$ if $f(x) - f(y) > \tau$ (and equally preferred if difference between $-\tau$ and τ).
This is known as a *semi-order* and can model hesitation in choice.

Interval Orders

Interval orders generalize semiorders: an item is preferred to another if its upper utility of the former is higher than the lower utility of the latter.

Axiomatization

In terms of the preference relation, a preference structure is a interval order iff $\succ$ is Ferrers, that is

- if $a \succ b$ and $c \succ d$ then either $a \succ d$ or $c \succ b$.

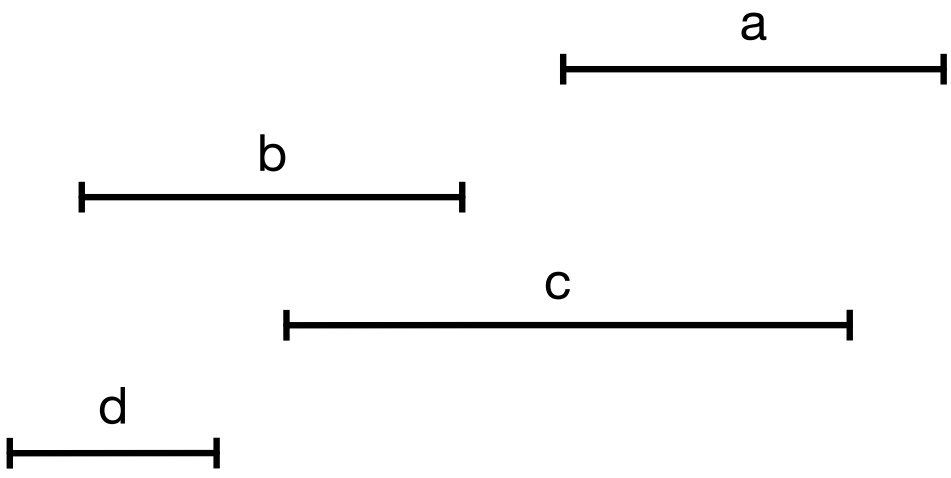
Note that in interval orders $\succ$ is transitive but not negative transitive, $\approx$ is not transitive and $\sim$ is empty.

Characterization

A preference structure is a interval order iff each item can be associated with a pair of (ordinal) values f_1 and f_2, with $f_1(x) \leq f_2(x)$ such that

- $a \succ b$ iff $f_1(a) > f_2(b)$
  (iff the intervals are disjoint and that of a is on the "right")
- $a \approx b$ iff $f_1(a) \leq f_2(b) \wedge f_2(a) \geq f_1(b)$
  (iff the intervals $[f_1(a), f_2(a)]$ and $[f_1(b), f_2(b)]$ overlap).

Interval Orders: Example



$a \succ b$, $a \approx c$, $a \succ d$
$b \approx c$, $b \approx d$
$c \succ d$

Sources of Incomparability

There are two main interpretations of incomparability:

- Lack of knowledge (extrinsic incomparability; epistemic incomparability)
- Incommensurability, inability to compare (intrinsic incomparability)

Multi-attribute domains

Consider multi-attribute domains, where there are several attributes $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$
An item $x = (x_1, x_2, \ldots, x_n)$ is defined as a vector where each $x_i \in Dom(X_i)$.

- Number of outcomes (configuration) is exponential in the number of attributes/variables
- Difficult (or impossible) to express a complete preference order in combinatorial domains
- Luckily, preferences usually exhibit some patterns

Representation Languages

- A *Compact* representation is useful so that preferences can be formulated with statements that encompass several alternatives
- A preference statement: *"I prefer red cars over to blue cars"*
- But....*What does this exactly mean? All red cards are preferred to blue cars? Some red cards are preferred to blue cars? There is at lest one red car preferred to a blue car? I prefer the average red car to the average blue car?*
- The need of a principled "semantics" for preference statements

The representation of preferences should be natural, easy to elicit, compact (in typical cases), and support effective inference.

Preference Independence

Preference Independence (PI) is a key notion in preference reasoning. It is the analogous of probabilistic independence for preference reasoning.

A subset of variables $X \subset \mathcal{X}$ is preferential independent of its complement $Y = \mathcal{X} - X$ iff, for all assignments $\mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}'$, holds:

$$(\mathbf{x}, \mathbf{y}) \succeq (\mathbf{x}', \mathbf{y}) \text{ iff } (\mathbf{x}, \mathbf{y}') \succeq (\mathbf{x}', \mathbf{y}')$$

We write $PI(X, Y)$ for preference independence.

Note: PI is not necessarily symmetric. We may have PI(X,Y) but not PI(Y,X)!

Mutual Preference Independence (MPI)

If all subsets of variables $X \subset \mathcal{X}$ are preferential independent of its complement $Y = \mathcal{X} - X$ then we have Mutual Preference Independence.

$\forall X \subset \mathcal{X}, PI(X, \mathcal{X} - X)$

We write *MPI* for mutual preference independence.

Why not just Singletons?

We can have that every attribute alone is independent by all others but not coalitions of attributes.

- We can have preferential indepence a preferred to a', b preferred to b', c preferred to c'
- It follows that the combination (a, b, C) is preferred to the combination (a', b', C) for $C = c, c'$
- But we can have (a, b', c) preferred to (a', b, c) with $C = c$ and (a', b, c') preferred to (a, b', c') (the tradeoff between A and B depends on C)

PI and Pareto Dominance

If all attributes are preference independent of their complement, one can use the notion of "Pareto-dominance"

x Pareto-dominates y *iff*
for all i, $x_i \succeq y_i$ and $\exists j$ such that $x_j \succ y_j$

- Price: $600 \succ_{Price} 601 \succ_{Price} 602 \succ_{Price} \ldots$
- Size: $100sm \succ_{Size} 99sm \succ_{Size} 98sm \succ_{Size} \ldots$
- Location: Downtown $\succ_{Loc}$ Uptown $\succ_{Loc}$ suburbs

Then $(700\$, 80, \textit{Uptown})$ dominates $(800\$, 70, \textit{Uptown})$ and the latter can be eliminated.

Useful as a first pruning of the set of choices.

Ordinal v.s. Cardinal Utility

Ordinal Utility

Monotone transformation
The only meaning is the order that they represent. No notion of "strength" of preferences

Cardinal utility

affine transformation (interval scale)
Given U, define $V = aU + b$; then U and V represent the same behavior.

- Therefore there is no meaning in saying, for example *"item 1 has two times the utility of item 2"*
- However we can compare intervals (as for temperatures).

Common Types of Scales

Name	Permissible transformation	Examples
Absolute	$\varphi(x) = x$	counting, numbers
Ratio scale	$\varphi(x) = \alpha x;\ \alpha > 0$	mass (kg to pounds: $\varphi(k) = 0.45k$)
		length (miles to km: $\varphi(m) = 1.60m$)
Interval scale	$\varphi(x) = \alpha x + \beta$	time (calendar)
		temperature (Celsius, Fahrenheit)
Ordinal scale	$\varphi(x)$ s.t.	preferences
	$x > y \implies \varphi(x) > \varphi(y)$	Mohs' scale to measure hardness
	$x = y \implies \varphi(x) = \varphi(y)$	
Nominal	φ one-to-one	subjects, brands of product

Table from book "Modeling Decision" of Torra and Narukawa, Springer, 2007.

Utility Representation

- Utility function $u : X_1 \times \ldots X_n \rightarrow [0, 1]$
- Ideal item $x^\top$ such that $u(x^\top) = 1$ and $u(x_\perp) = 0$ (scaling)
- Representing, eliciting u difficult in explicit form
- Flat utility representation often unrealistic

	Manufacturer	*Type*	*Fuel*	*Color*	...	**Utility**
item 1 (Yaris)	Toyota	compact	hybrid	black	...	0.82
item 2 (Zoe)	Renault	ultra compact	electric	red	...	1
item 3 (Kadjar)	Renault	suv	diesel	red	...	0.96
...	...	...	...	...	...	...

Not much better than representing preference orders directly!

Additive Utility Functions

MPI is a necessary (but not sufficient, in general) condition for an Additive Utility represenation

- Sum of local utility functions u_i over attributes (or local value functions v_i multiplied by scaling weights)
- Exponential reduction in the number of needed parameters

$$u(x) = \sum_{i=1}^{n} u_i(x_i) = \sum_{i=1}^{n} \alpha_i v_i(x_i) \quad (1)$$

Color	v_1
red	1.0
blue	0.7
...	...
green	0.0

Manufacturer	v_2
Toyota	1
Renault	0.6
...	...
FIAT	0

Importance for attribute "color": $\alpha_1 = 0.2$, for "manufacturer": $\alpha_2 = 0.3$.

Additive Utility Functions and PI

It is easy to see that AU satifies MPI

Dinner example: choose main course (Meat or Fish), wine (Red or white) and dessert (yes/no).

$u((Meat, Red, no)) > u((Fish, Red, no)) \iff$
$u_c(Meat) + u_w(Red) + u_d(no) > u_c(Fish) + u_w(Red) + u_d(no) \iff$
$u_c(Meat) > u_c(Fish)$

But then it also holds:
$u_c(Meat) + u_w(White) + u_d(yes) > u_c(Fish) + u_w(White) + u_d(yes)$
that means that $u((Meat, White, yes)) > u((Fish, White, yes))$

AU makes very strong assumptions: the additional utility for having Meat instead of Fish is always the same no matter the wine and the dessert. (we cannot model that we prefer white wine with fish, for example!)

Graphical Models for Preferences

GMs for preference representation adopt essential ideas of Bayes nets into preference modelling

- Structure:
  - Directed graph with variables as nodes
  - Edges represent direct influence
- Preference independence is used to reduce the required information

Main graphical models:

- Qualitative: CP-nets, TCP-nets
- Quantitative: GAI-nets, UCP-nets

Applications: product configuration, e-commerce search, personalized agents.

Conditional Preference Independence

Conditional Preference Independence (CPI)

Let (X, Y, Z) be a partition of $\mathcal{X}$ into three disjoint non-empty sets. X is *conditionally preferentially independent of Y given $Z = \mathbf{z}$* if and only if, for all $\mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}'$ holds

$$(\mathbf{x}, \mathbf{y}, \mathbf{z}) \succeq (\mathbf{x}', \mathbf{y}, \mathbf{z}) \text{ iff } (\mathbf{x}, \mathbf{y}', \mathbf{z}) \succeq (\mathbf{x}', \mathbf{y}', \mathbf{z})$$

If this is true for all $\mathbf{z} \in Dom(Z)$, then X is *conditionally preferentially independent of Y given Z*.

We write $CPI(X, Y|Z)$ for conditional preference independence of X and Y given Z.

Ceteris Paribus semantics

- **Ceteris Paribus** (latin): all else being equal
  The preference holds only when comparing two outcomes that differ in the variables mentioned, and are the same on all variables
- Example: I prefer wine to beer with my meal
  Given two identical meals, one with wine and one with beer, I prefer the former

<<When discussing with my wife what table to buy for our living room, I said: "A round table is better than a square one." By this I did not mean that irrespectively of their other properties, any round table is better than any square-shaped table. Rather, I meant that any round table is better (for our living room) than any square table that does not differ significantly in its other characteristics, such as height, sort of wood, finishing, price, etc. This is preference ceteris paribus or "everything else being equal". Most of the preferences that we express or act upon seem to be of this type >>

Hansson. What is ceteris paribus preference. *Journal of Philosophical Logic*, 1996.

Ceteris Paribus Statements

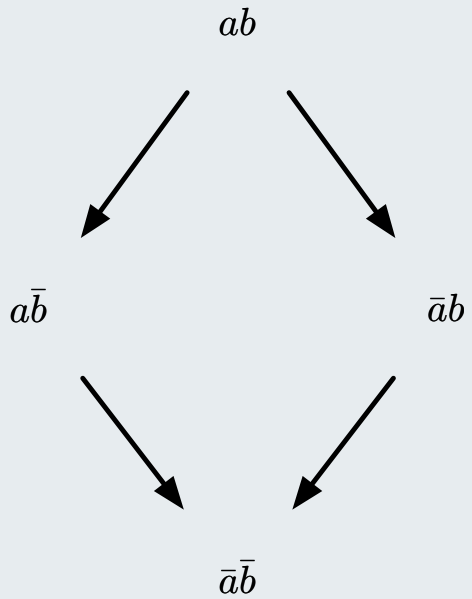
Let A be an attribute and B = X - A the set of all other attributes.

CP Statements

a preferred to a' *ceteris paribus* **iff**
$(a, \mathbf{b}) \succeq (a', \mathbf{b}) \quad \forall \mathbf{b} \in Dom(B)$

CP statements may be joined together to induce a partial order:

$a \succ \bar{a}$
$b \succ \bar{b}$



- From $a \succ \bar{a}$: we deduce $(a, b) \succ (\bar{a}, b)$ and $(a, \bar{b}) \succ (\bar{a}, \bar{b})$
- From $b \succ \bar{b}$: we deduce $(a, b) \succ (a, \bar{b})$ and $(\bar{a}, b) \succ (\bar{a}, \bar{b})$

- If we assume transitivity of preference, we can deduce (a, b) preferred to $(\bar{a}, \bar{b})$
  Note: in the graph we may not show redundant arcs (*Hasse diagram*)
- Nothing is known about the preference between $(a, \bar{b})$ and $(\bar{a}, b)$

Conditional CP Statements

"I prefer red wine to white wine with my meal, ceteris paribus, given that meat is served"

Given two identical meals in which meat is served, I prefer red wine to white wine.
It tells us nothing about two identical meals in which meat is NOT served.

Conditional CP Statements

- Let A and C be two attributes, and $B = \mathcal{X} - A - C$ all the other attributes
- a preferred to a' given c *ceteris paribus* **iff** $(a, \mathbf{b}, c) \succeq (a', \mathbf{b}, c)$ for all $\mathbf{b} \in Dom(B)$

CP Statements and Independence

Ceteris Paribus preference statements induce independence relations:

- If my preference for wine depends on (and only on) the main course, then wine choice is conditionally preferentially independent of all other variables given the main course value.
- If my ceteris paribus preferences is unconditional, then main course is preference independent of all other variables.

Example

A dinner consists of a *main course*, a *wine* and a *dessert*

- I strictly prefer meat to a fish as main course.
- I (strictly) prefer to open a bottle of red whine if the main course is meat, and white wine if the main course is fish.
- If the main course is meat I strictly prefer not having dessert, otherwise I do (strictly) prefer to have dessert.

Ceteris-paribus interpretation of these statements:

- $(\text{Meat}, w, d) \succ (\text{Fish}, w, d)$ for any $w \in \{\text{Red}, \text{White}\}, d \in \{\text{Yes}, \text{No}\}$
- $(\text{Meat}, \text{Red}, d) \succ (\text{Meat}, \text{White}, d)$ for any $d \in \{\text{Yes}, \text{No}\}$

  $(\text{Fish}, \text{White}, d) \succ (\text{Fish}, \text{Red}, d)$ for any $d \in \{\text{Yes}, \text{No}\}$
- $(\text{Meat}, w, \text{Yes}) \succ (\text{Meat}, w, \text{No})$ for any $w \in \{\text{Red}, \text{White}\}$

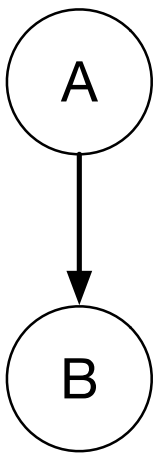
  $(\text{Fish}, w, \text{Yes}) \succ (\text{Fish}, w, \text{No})$ for any $w \in \{\text{Red}, \text{White}\}$

CP Networks

A CP network over variables $\mathcal{V}$ is a directed graph $G = (\mathcal{V}, \mathcal{E})$ where

- Vertices V are variables representing attributes of the decision problems
- Edges (X,Y) indicate that the preference over Y might depend on the value assigne to X.

  The set of "parents" of X: $Pa(X) = \{Y \in \mathcal{V} | (Y, X) \in \mathcal{E}\}$
- Vertices are annotated with *conditional preferences tables (CPT)*: each $CPT(X)$, for a variable $X \in \mathcal{V}$ expresses a total order $\succ_u^X$ with each instantiation u of X's parents $Pa(X) = U$.



$a \succ \bar{a}$

$a : b \succ \bar{b}$

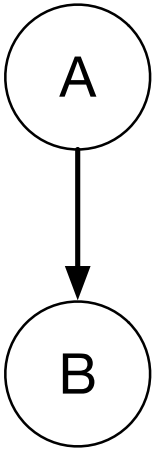
$\bar{a} : \bar{b} \succ b$

$Pa(A) = \{\}$

$Pa(B) = \{A\}$

Semantics:

- a preference order $\succ$ satisfies a rule of type "$z : x_1 \succ x_2 \succ \ldots$" iff $\succ$ is such that $(x_1, z, y) \succ (x_2, z, y)$ for all values z of varibales Z not in the parents of X
- a preference order $\succ$ satisfies a CPT table of a variable X iff $\succ$ satisfies each rule of the CPT table (i.e. for all values z of the parents' variables of X)
- a CP network is satisfiable if it exists a total pre-order over outcomes that satisfy all preferences given by CPT tables for each of the node



$a \succ \bar{a}$

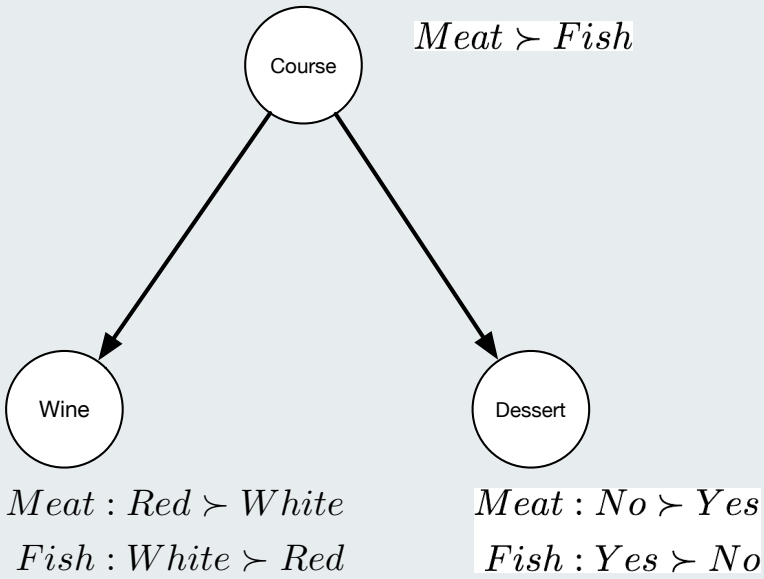
$a : b \succ \bar{b}$

$\bar{a} : \bar{b} \succ b$

Satisfiable by preference order: $(a, b) \succ (a, \bar{b}) \succ (\bar{a}, \bar{b}) \succ (\bar{a}, b)$

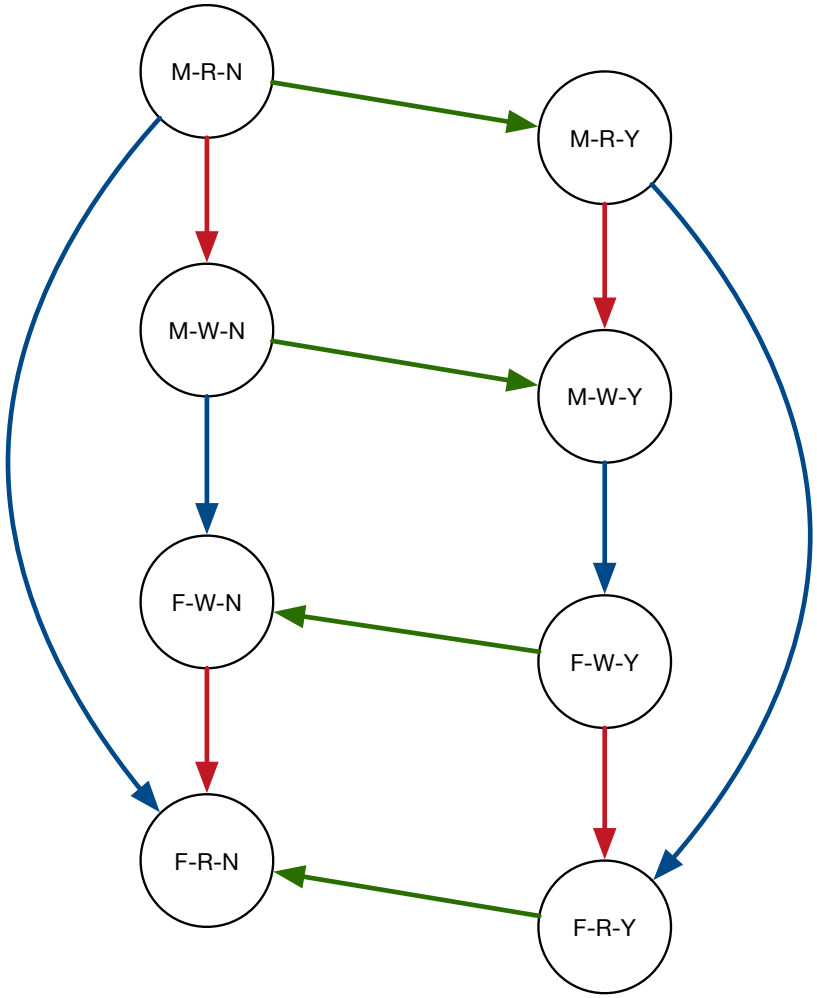
Dinner Example: a dinner consists of a *main course*, a *wine* and a *dessert*

- I strictly prefer meat to a fish as main course.
- I (strictly) prefer to open a bottle of red whine if the main course is meat, and white wine if the main course is fish.
- If the main course is meat I strictly prefer not having dessert, otherwise I do (strictly) prefer to have dessert.



PI(Course, { Wine, Dessert })
CPI(Wine, Dessert | Course)
CPI(Dessert, Wine | Course)

Preference graph for the dinner example.



CP Statements:

1: $(\text{Meat}, \cdot, \cdot) \succ (\text{Fish}, \cdot, \cdot)$

2: $(\text{Meat}, \text{Red}, \cdot) \succ (\text{Meat}, \text{White}, \cdot)$,
$(\text{Fish}, \text{White}, \cdot) \succ (\text{Fish}, \text{Red}, \cdot)$

3: $(\text{Meat}, \cdot, \text{No}) \succ (\text{Meat}, \cdot, \text{Yes})$,
$(\text{Fish}, \cdot, \text{Yes}) \succ (\text{Fish}, \cdot, \text{No})$

In general there are several total pre-orders consistent with a CP-net.

$(M, R, N) \succ (M, R, Y) \succ (M, W, N) \succ (M, W, Y) \succ (F, W, Y) \succ (F, W, N) \succ (F, R, Y) \succ (F, R, N)$

$(M, R, N) \succ (M, W, N) \succ (M, R, Y) \succ (M, W, Y) \succ (F, W, Y) \succ (F, R, Y) \succ (F, W, N) \succ (F, R, N)$

...

The "semantic" of a CP-network is represented by a partial (strict) order $\succ_{Ncal}$ that specifies which preferences hold "for sure".

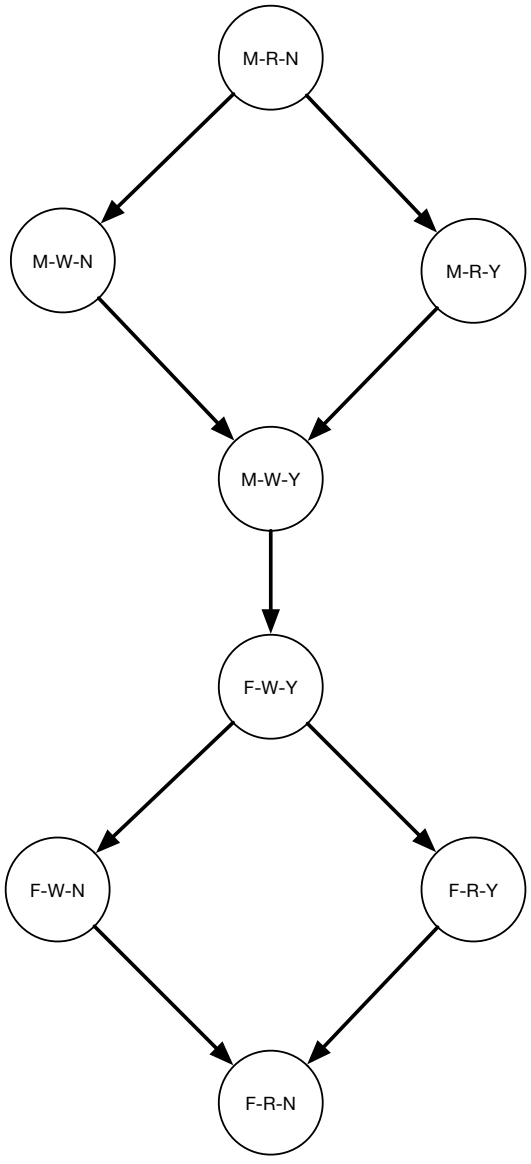
Formally one says that a CP network $\mathcal{N}$ entails (written $\models$) a preference if this holds for all preference orders satisfying the network.

In our example

$$\mathcal{N} \models (M, R, N) \succ (M, W, Y)$$
$$\mathcal{N} \models (M, R, Y) \succ (F, W, N)$$
$$\mathcal{N} \models (M, R, Y) \succ (F, W, N)$$

...

$\succ_{\mathcal{N}}$ is such that $o \succ_{\mathcal{N}} o'$ iff $\mathcal{N} \models o \succ o'$



Hasse diagram of the partial order $\succ_{\mathcal{N}}$ associated to the CP-net of the example.

It can be obtained from the preference graph by removing redundant links due to transitivity.

CP-nets with Indifference

Example

Two variables A and B with A parent of B

- CPT for A: $a \sim \bar{a}$
- CPT for B:
  - if $A = a$ then $b \succ \bar{b}$
  - if $A = \bar{a}$ then $\bar{b} \succ b$

Then we derive

$$(a, b) \succ (a, \bar{b}) \succeq (\bar{a}, \bar{b}) \succ (\bar{a}, b) \succeq (a, b)$$

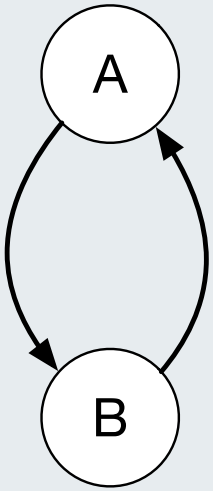
hence the CP network is not satisfiable.

Possible ways to handle indifference:

- restrict the values of CPTs: if we are indifferent between assigning $X = x$ and $X = x'$ then the CPTs of the children of X should have the same values whenever in the context of x or $x' \Longrightarrow$ satisfiable
- clustering of variables
- cyclic networks

Cyclic CP-nets

Example 1

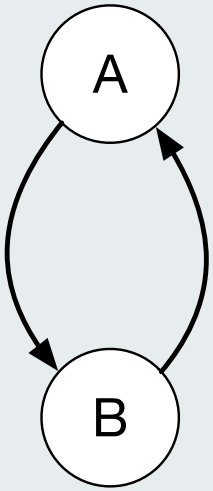


A and B both with binary values

- CPT for A:
  - if $B = b : a \succ \bar{a}$
  - if $B = \bar{b} : \bar{a} \succ a$
- CPT for B:
  - if $A = a : b \succ \bar{b}$
  - if $A = \bar{a} : \bar{b} \succ b$

Cyclic CP-nets

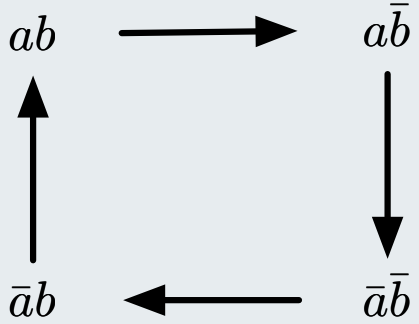
Example 2



A and B both with binary values

- CPT for A:
  - if $B = b$: $\bar{a} \succ a$
  - if $B = \bar{b}$: $a \succ \bar{a}$
- CPT for B:
  - if $A = a$: $b \succ \bar{b}$
  - if $A = \bar{a}$: $\bar{b} \succ b$

Preference oder: (CP not satisfiable)



What do we do with CP-nets?

Typical problems:

- Optimization: find optimal outcome (unconstrained), find undominated outcome given a set of possible or feasible outcomed (constrainted optimization)
- Comparison of two outcomes given a CP-net (dominance test)
- Set ordering

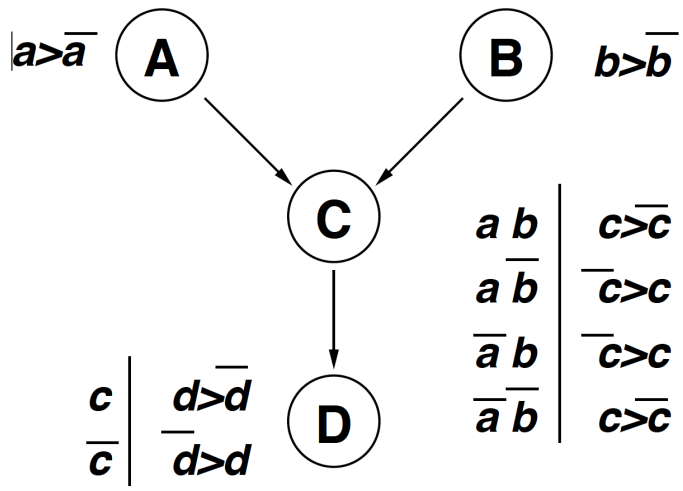
Optimization in CP-nets

Forward-sweep Algorithm

1. Sort vertices in topological order (parents comes before sons)
2. Assign the next attribute to its most preferred value given the parents
3. **Repeat** step 2 until all attributes are assigned

Example:

- Start from A and B that have no parents
  assign $A = a$ and $B = b$
- Then C (whose parents are A and B)
  Look most preferred value given $(A, B) = (a, b)$:
  assign $C = c$
- Then assign d to D



Optimal outcome is (a, b, c, d).

Constrained Optimization

Some configuration may be *unfeasible*; we therefore aim at finding the "best" feasible configuration.

- Let feas : $x \rightarrow Bool$ be the feasibility function
- x such that $\text{feas}(x) \wedge \nexists y : y \succ x \wedge \text{feas}(y)$.

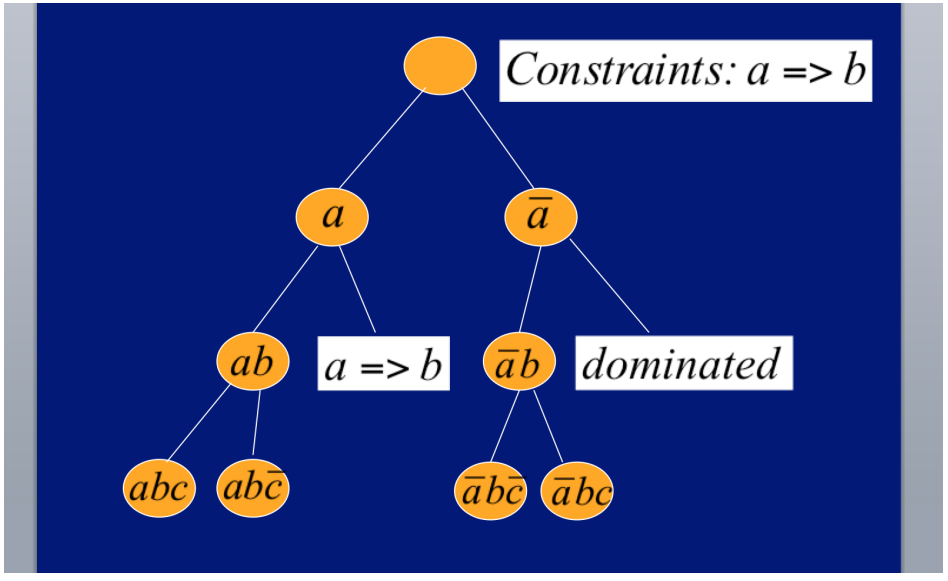
Algorithm "Ordered Generate & Test"

1. Generate outcomes in non-increasing order
2. Test for feasability (the first feasible outcome is optimal!)

If more than one is needed, maintain a set of optimal solutions and compare new feasible solutions against the current optimal set using dominance testing.

Non-improving Sequence of Outcomes

- Select a topological ordering over variables (consistent with the CP-net structure)
- Build an search tree by instantiating variables in this order
- Variable values are ordered based on the CPT
- Leaf nodes, ordered left to right provide a non-increasing sequence of outcomes
- Prune with constraints



Dominance test

Problem

Given a CP-net $\mathcal{N}$ and two outcomes o^1 and o^2 determine whether $\mathcal{N} \models o^1 \succ o^2$

- If there is no flipping sequence between o and o', they are incomprarable
- A sequence of improving flips from one outcome to another provides a proof that one outcome is preferred, or dispreferred, to another
- Worsening flips: changing the value of an attribute in a way that we obtain an outcome that is less preferred according to one of the statements
- An outcome o^1 is preferred to o^2 iff there is a sequence of worsening flips from o^1 to o^2

Flipping Sequence Search Algorithm

TreeDT for binary-valued tree-structure CP nets

Test $o \succ o'$

1. Initialize variables with values from o'
2. **Loop**
   - Remove from V all the leafs whose value is the same as o
   - If $V = \varnothing$ then **return** YES
   - Find a variable X s.t. its value can be improved, *while none of its descendent can*, given the current assignment
   - If no variable can be found **return** NO
   - Update the value of X

Note that TreeDT is backtrack-free.

Complexity is $O(n^2)$ where n is the number of variables in the CP-net.

Dominance test

- In the general case, dominance can implemented as a planning problem
- Search techniques with pruning methods can be used

Complexity results

- Dominance test for CP-nets with tree structure is quadratic in the number of variables
- Dominance test for CP-nets with polytree structure are also easy (polynomial)
- In the other cases, it is hard (NP-complete for specific kind of graphs, PSPACE-complete in general)

Ordering Queries

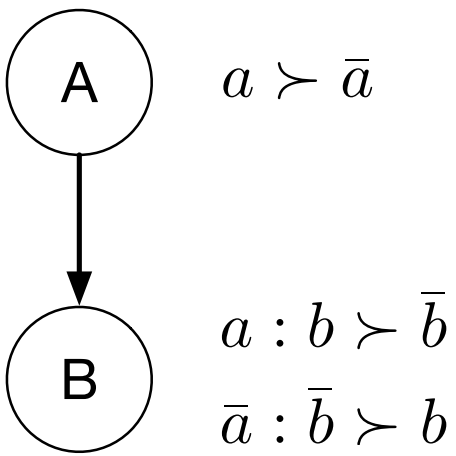
- Given outcomes o and o' the system must report whether $\mathcal{N} \not\models o \succ o'$ or $\mathcal{N} \not\models o' \succ o$
- Surprinsigely this can computed very fast in acyclic networks.
- Ordering queries with respect to acyclic (not nec. binary) CP-nets can be answered in linear time wrt the number of variables.

Ordering Several Outcomes

Given an acyclic CP-nets defined on variables $\mathcal{X}$ and a set of outcomes (configurations) $o_1, \ldots, o_m$, ordering these outcomes consistenlty can be done using ordering queries only.

Induced Importance Relations

Configurations that "violates" the preference of node *A* are placed lower than configurations that violate the preferences of *B*



Resulting partial order induced by the network:

$$(a, b) \succ (a, \bar{b}) \succ (\bar{a}, \bar{b}) \succ (\bar{a}, b)$$

$(a, \bar{b})$, that violates the child's preference, is preferred to $(\bar{a}, \bar{b})$, violating the parent's preference

Explicit Importance Statements

In the partial order $\succ_{\mathcal{N}}$ represented by a CP-net, there might be several incomparable outcomes.

When considering trade-offs, one may wish to express that one variable's value is more important than another's: idea of extending CP-nets with importance statements.

If it is more important to me that the value of X be high than the value of Y be high, then X is *more important* than Y.

If, given $z \in Dom(Z)$, it is more important to me that the value of X be high than the value of Y be high, then X is *conditionally more important* than Y

TCP-nets

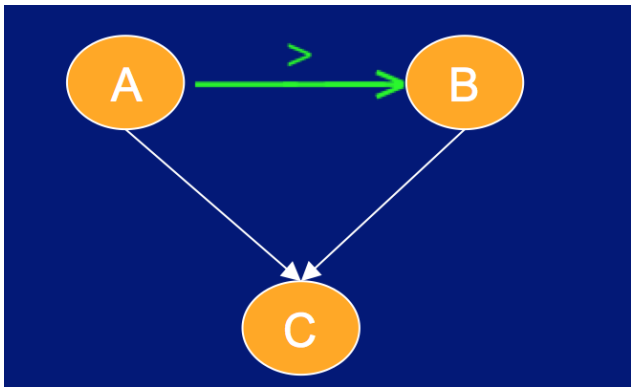
TCP-nets (conditional preference networks with tradeoffs) allow the user expresses preferences over *compromises*

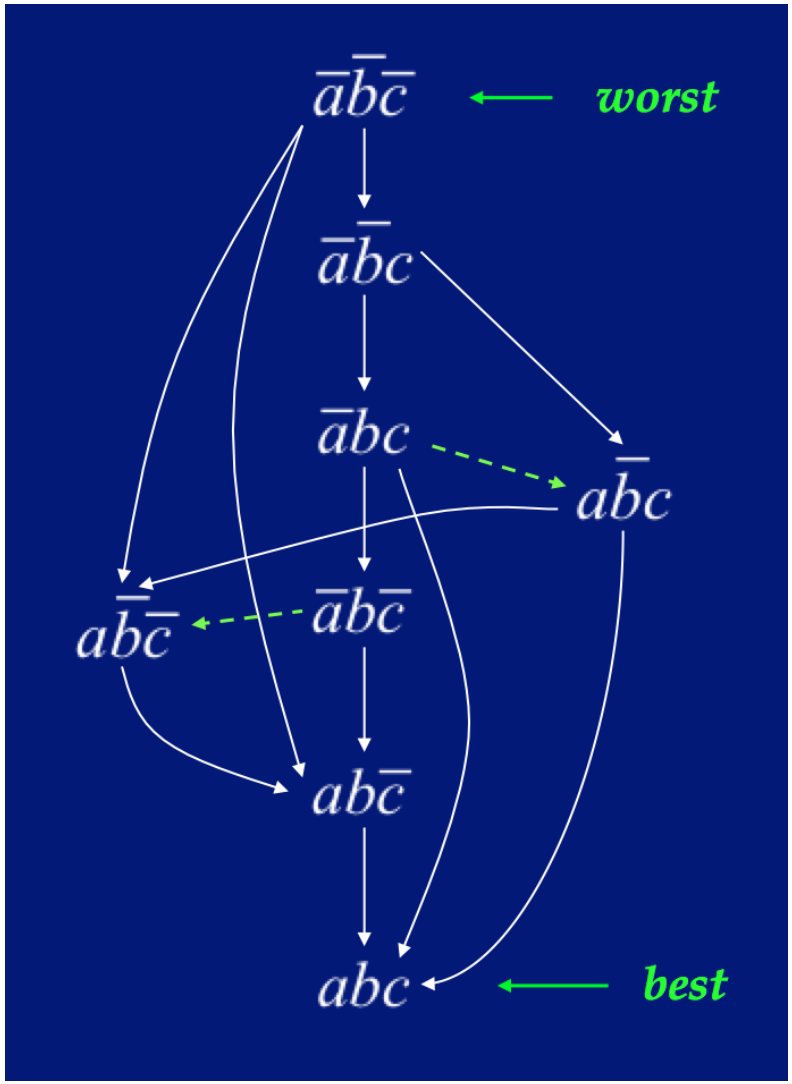
A TCP-net supports the following statements:

- ceteris paribus statements (as in CP-nets)
- unconditional importance statements (A more important than B)
- conditional importance stratements (if A=a then B is more important than C)

In addition to CPT: conditional importance table (CIT)

CPT for A: $a \succ \bar{a}$
CPT for B: $b \succ \bar{b}$
CPT for C: if $(a, b) \vee (\bar{a}, \bar{b})$: $c \succ \bar{c}$
CPT for C: if $(a, \bar{b}) \vee (\bar{a}, b)$: $\bar{c} \succ c$
A more important than B





Return to the Quantitative World

- We strive for the effectiveness of the quantitative approach (based on utility) while profiting from decomposition
- From additive utility to generalized additive utility
- Graphical model: GAI networks

Generalized Additive Utility

- Sum of local utility functions u_I over *sets of attributes* (or local value functions v_i multiplied by scaling weights)
- Higher descriptive power than strictly additive utilities, while still having a manageable number of parameters

$$u(x) = \sum_{i=1}^{m} u_{J_i}(x_{J_i}) = \sum_{i=1}^{n} \alpha_{J_i} v_{J_i}(x_{J_i})$$

where J_i is a set of indices, x_{J_i} the projection of x on J_i and m the number of factors.

Fuel	**Manufacturer**	$v_{color,brand}$
hybrid	Toyota	1
gas	Toyota	0.5
electric	Fiat	0.4
diesel	Renault	0.8
...	...	...

Color	v_{color}
red	0.6
blue	0.8
black	1

Importance for factor "fuel,brand": $\alpha_{J_1} = 0.2$, for "color": $\alpha_{J_2} = 0.3$.
Obviously GA utility generalizes AU.

GAI: Factors can Overlap

The union of the factors J_i must give the set of attributes.

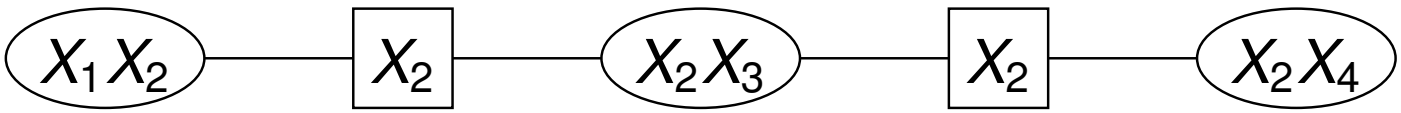
- dependency between factors
- decomposition not unique: utility can "shift"

$$\begin{aligned} u(x_1, x_2, x_3, x_4) &= u_1(x_1) + u_2(x_2, x_3) + u_3(x_3, x_4) \\ &= u_1(x_1) + [u_2(x_2, x_3) + f(x_3)] + [u_3(x_3, x_4) - f(x_3)] \\ &= u_1(x_1) + u'_2(x_2, x_3) + u'_3(x_3, x_4) \end{aligned}$$

$\rightarrow$ elicitation of GAI networks need special attention

GAI networks

Graphical model that exploits the structure of the utility decomposition.



- Eclipses: factor (interpreted as a clique)
- Rectangle: separator (common attributes)

If factors do not overlap: rectangles are empty, easy

Property

For any pair of factors J_1, J_2 with a non empty intersection of attributes S, S is contained in all factors and all separators in the chain between J_1 and J_2.

Similarity with jonction trees in Bayesian networks.

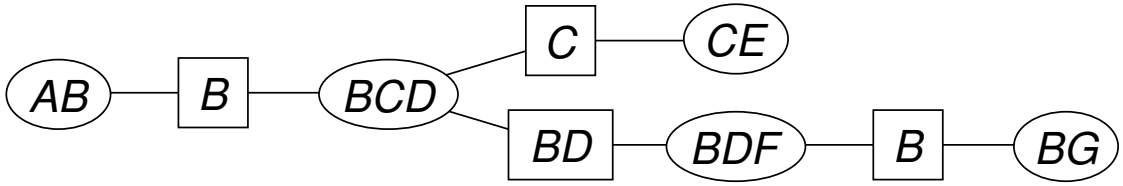
GAI networks

- Comparison (dominance) test: easy, just compute
- Optimization: message-passing algorithm (next slides)

Note: in the CP net it was easy (when without cyles) to find optimal assignement, but it was more difficult to determine whether a configuration was more preferred to another one.

Optimization in GAI networks

Slide courtesy of Christophe Gonzales



Optimum

$\max_{A,B,C,D,E,F,G}$
$u_1(A,B) + u_2(C,E) + u_3(B,C,D) + u_4(B,D,F) + u_5(B,G)$

$\max_A \max_B \max_C \max_D \max_E \max_F \max_G$
$u_1(A,B) + u_2(C,E) + u_3(B,C,D) + u_4(B,D,F) + u_5(B,G)$

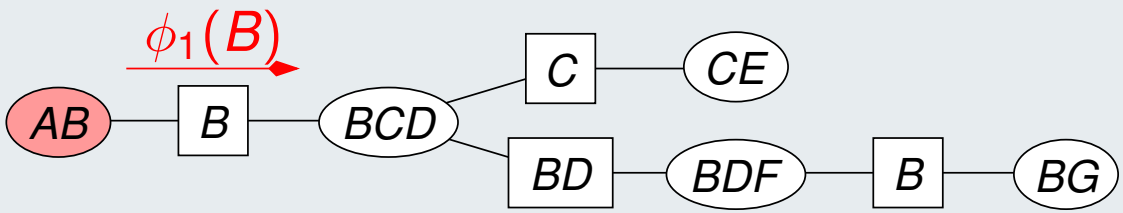
Aggregation Phase (1/5)

Slide courtesy of Christophe Gonzales

Elimination of A

$$\max_B \max_C \max_D \max_E \max_F \max_G \max_A u_1(A,B) + u_2(C,E) + u_3(B,C,D) + u_4(B,D,F) + u_5(B,G)$$

$$= \max_B \max_C \max_D \max_E \max_F \max_G \phi_1(B) + u_2(C,E) + u_3(B,C,D) + u_4(B,D,F) + u_5(B,G)$$



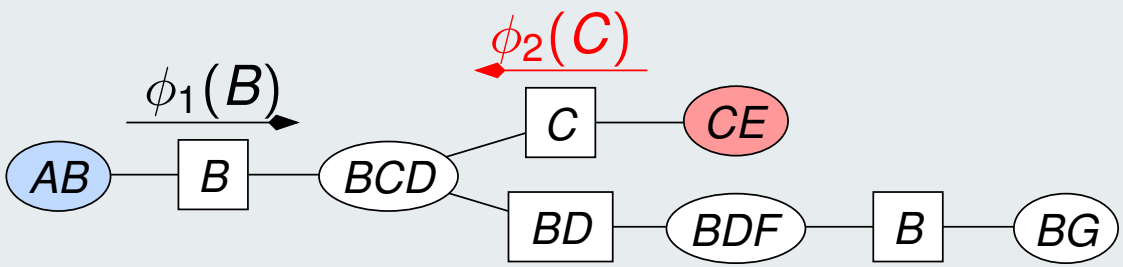
Aggregation Phase (2/5)

Slide courtesy of Christophe Gonzales

Elimination of E

$$\max_B \max_C \max_D \max_F \max_G \max_E u_2(C,E)$$
$$+\phi_1(B) + u_3(B,C,D) + u_4(B,D,F) + u_5(B,G)$$

$$= \max_B \max_C \max_D \max_F \max_G$$
$$\phi_1(B) + \phi_2(C) + u_3(B,C,D) + u_4(B,D,F) + u_5(B,G)$$



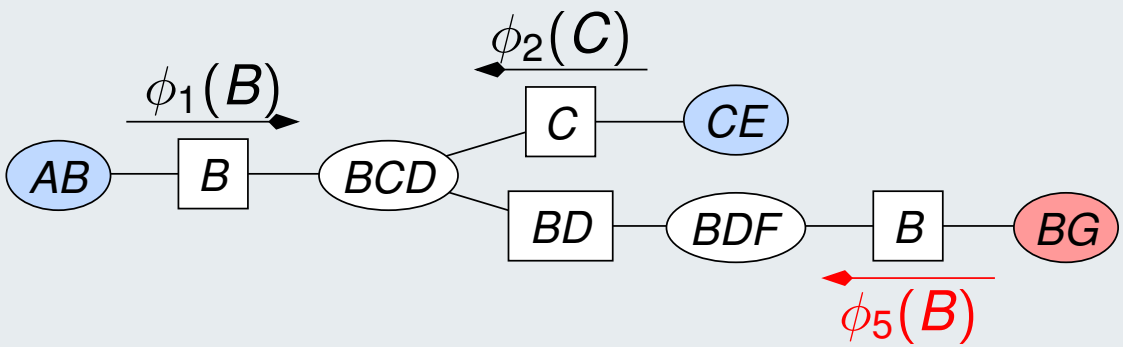
Aggregation Phase (3/5)

Slide courtesy of Christophe Gonzales

Elimination of G

$$\max_B \max_C \max_D \max_F \max_G u_5(B,G) + \phi_1(B) + \phi_2(C) + u_3(B,C,D) + u_4(B,D,F)$$

$$= \max_B \max_C \max_D \max_F \max_G \phi_1(B) + \phi_2(C) + u_3(B,C,D) + u_4(B,D,F) + \phi_5(B)$$



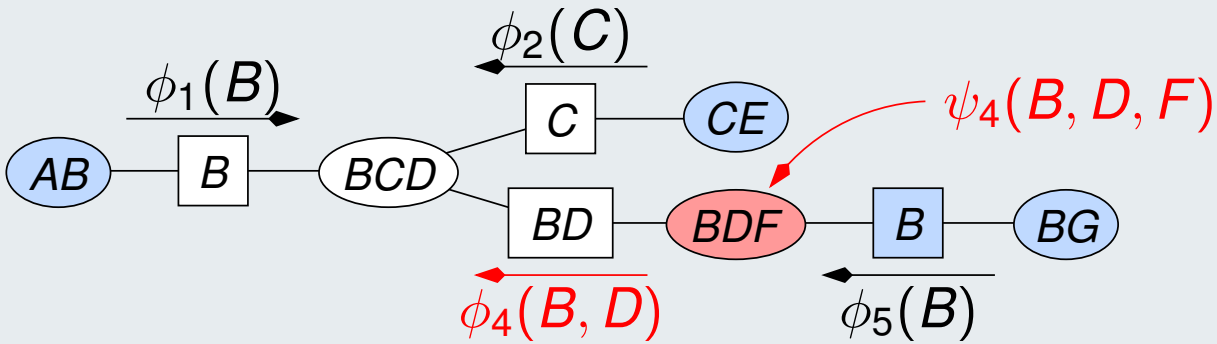
Aggregation Phase (4/5)

Slide courtesy of Christophe Gonzales

Elimination of F

$\max_B \max_C \max_D \max_F \psi_4(B, D, F)$
$+\phi_1(B) + \phi_2(C) + u_3(B, C, D)$
where $\psi_4(B, D, F) = u_4(B, D, F) + \phi_5(B)$

$= \max_B \max_C \max_D \phi_1(B) + \phi_2(C) + u_3(B, C, D) + \phi_4(B, D)$



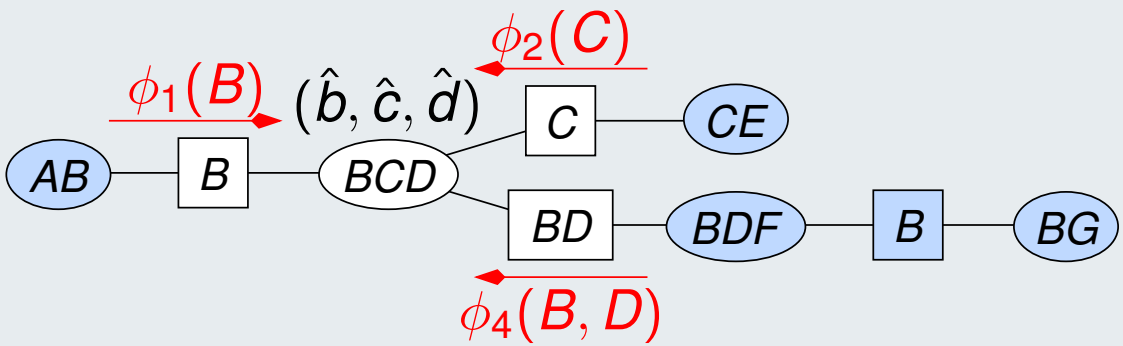
Aggregation Phase (5/5)

Slide courtesy of Christophe Gonzales

Elimination of B, C, D

$\max_{B,C,D} \psi_3(B, C, D)$
where $\psi_3(B, C, D) = \phi_1(B) + u_3(B, C, D) + \phi_4(B, D) + \phi_2(C)$

$\Longrightarrow$ preferred element: $(\hat{b}, \hat{c}, \hat{d})$

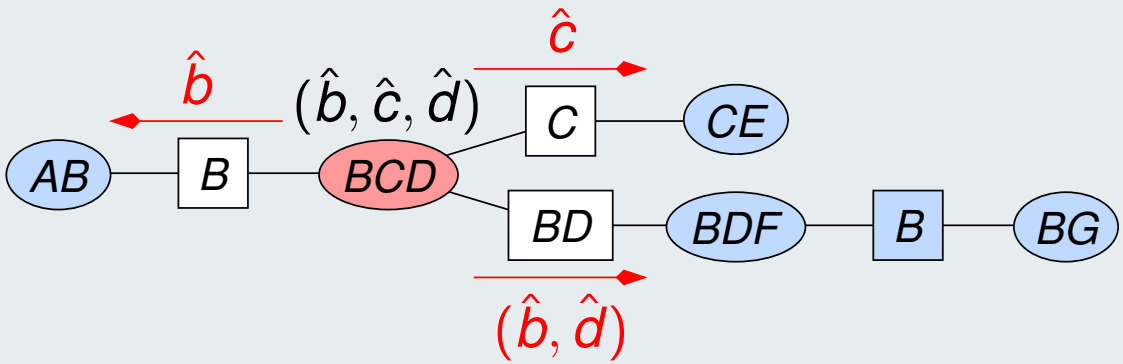


Distribution Phase (1/4)

Slide courtesy of Christophe Gonzales

Distribution from factor BCD

The factor *BCD* sends messages to his adjacent separators with the value of their atttributes at the optimum



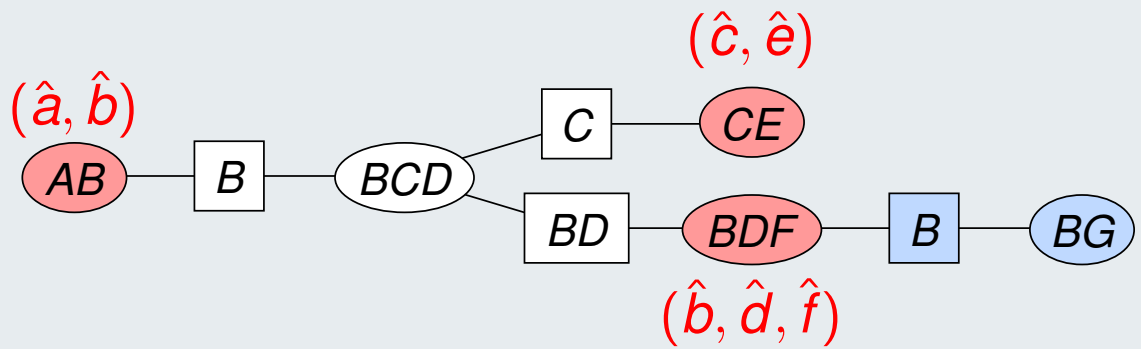
Distribution Phase (2/4)

Slide courtesy of Christophe Gonzales

Propagation from the separators

Factor $AB \implies \hat{a} = \text{argmax}_A \; u_1(A, \hat{b})$
Factor $CE \implies \hat{e} = \text{argmax}_E \; u_2(\hat{c}, E)$
Factor $BDF \implies \hat{f} = \text{argmax}_F \; \psi_5(\hat{b}, \hat{d}, F)$

Note: the argmax are calculated during the aggregation phase (ϕ)

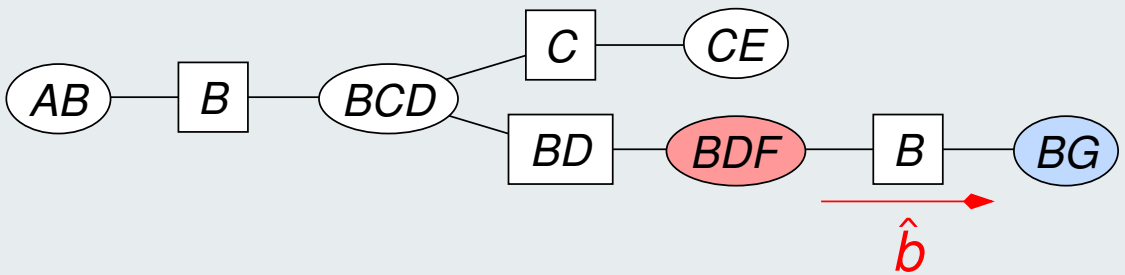


Distribution Phase (3/4)

Slide courtesy of Christophe Gonzales

Distribution par la clique BDF

The factor *BDF* sends a message to the separator *B* with the value of *B* at the optimum



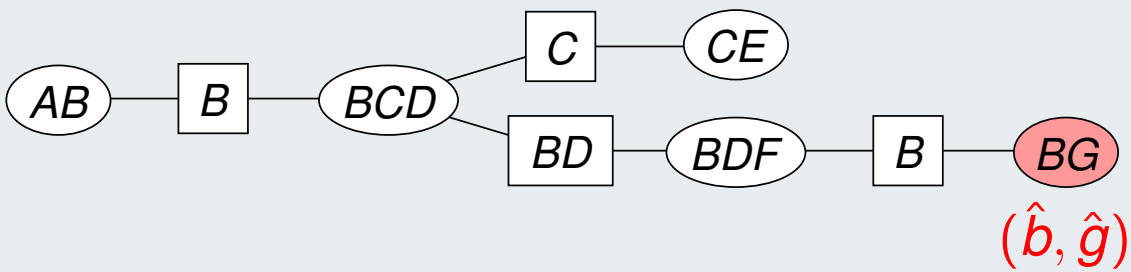
Distribution Phase (4/4)

Slide courtesy of Christophe Gonzales

Distribution par la clique BDF

Factor $BG \Longrightarrow \hat{g} = \text{argmax}_G \; u_5(\hat{b}, G)$

$\Longrightarrow$ an optimal (preferred) element $(\hat{a}, \hat{b}, \hat{c}, \hat{d}, \hat{e}, \hat{f}, \hat{g})$



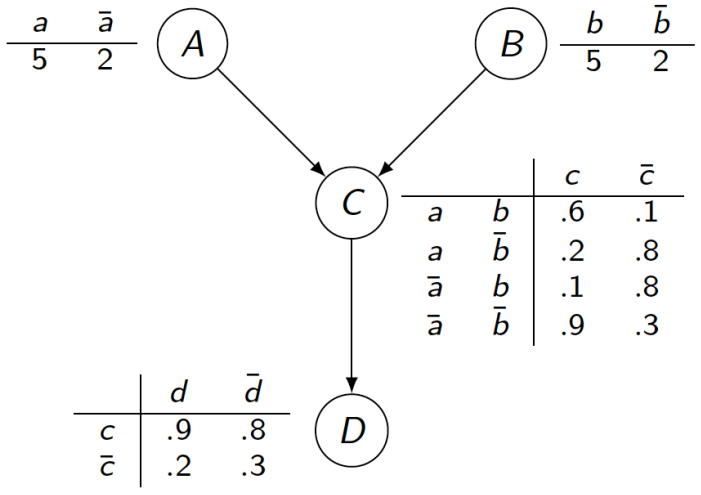
UCP networks

- Directed graphical representation of utility functions: UCP networks
- It combines generalized additive models and CP networks.
- Decomposition of utility function into a number of additive factors
- The CP-semantics ensures that computing optimization and dominance queries is very efficient.
  - Optimization: easy (forward-sweep procedure)
  - Dominance: easy (GAI computation)

Note:

- every UCP-net specifies a GAI decomposition of the underlying utility function u.
- BUT not every GAI decomposition can be represented in a UCP-net.

UCP networks



The graph must be a DAG

GAI decomposition:

$$u(a, b, c, d) = u_A(a) + u_B(b) + u_C(a, b, c) + u_D(c, d)$$

Attributes on "top" independent from the ones on "bottom"

Logical Languages for Preferences

Components

- Set of states or worlds Ω (i.e. the alternatives)
- Literals: example wine or meat
- Connectors: $\wedge$, $\vee$, $\neg$, $\rightarrow$
- Formulas: ϕ combine literals with connectors;
  ϕ : wine $\wedge$ meat

We write $\omega \models \phi$ if ϕ is satisfied in the state ω.

The model of a forumula $Mod(\phi)$ is the set of the states that satisfy ϕ:
$\omega \models \phi \iff w \in Mod(\phi)$

- What does it mean to say $\phi \succ \psi$?
- Von Wright's interpretation: worlds such $\phi \wedge \neg\psi$ are preferred to $\neg\phi \wedge \psi$.

Logical Languages for Preferences

Classical logic looks for the states that satisfy *all* formulas given in input

Our goal: given a set of (preference) statements, output a preference order on the ω

Logical languages for preferences

Idea 1: compute a score assigned to each formula by aggregating a score based on satisfaction of the formulas.

- Penalty logic
- Possibilistic logic
- Weighted Logics

Idea 2: assume a precise semantic (ceteris paribus, but may also be different one) and combine in a preference order

- Conditional Logic

References

F Bacchus et A Grove (1995). "Graphical models for preference and utility", Proceedings d'UAI-95

Craig Boutilier, Fahiem Bacchus, Ronen I. Brafman: *UCP-Networks: A Directed Graphical Representation of Conditional Utilities*. UAI 2001: 56-64

Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, David Poole: *CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements*. J. Artif. Intell. Res. (JAIR) 21: 135-191 (2004)

C. Gonzales, P. Perny (2004) *GAI Networks for Utility Elicitation*. Proceedings de KR-04.

D Braziunas, C Boutilier (2005). *Local utility elicitation in GAI models*, Proceedings d'UAI-05

Ronen I. Brafman, Carmel Domshlak, Solomon Eyal Shimony: *On Graphical Modeling of Preference and Importance*. J. Artif. Intell. Res. (JAIR) 25: 389-424 (2006)

Souhila Kaci. *Working with Preferences: Less Is More.* Cognitive Technologies, Springer 2011