



## Graphical Languages for Preference Representation and Applications

---

Sylvain Bouveret  
Onera Toulouse

Jérôme Lang  
Université Paris Dauphine

Twenty second International Joint Conference on Artificial Intelligence

Barcelona, July 16–22, 2011



## Why preferences in Artificial Intelligence?

- *planning/robotics*: an autonomous agent acts as a proxy for a user
- *individual/collective decision aid*: the system helps the user to make a decision.  
*Examples*: recommender systems, product configuration (and more generally e-commerce).

Specifying a goal: often insufficient

*I want a cheap, direct and fast train from Paris to Barcelona*

We need to allow for

- flexibility
- partial satisfaction
- trade-off solutions if the goal cannot be reached



## Preference structures

$\mathcal{E}$  set of **alternatives** or **outcomes**

Specifying preferences on  $\mathcal{E}$ : **comparing** / **ranking** alternatives.

### Ordinal preferences

Preference relation on  $\mathcal{E}$ : **reflexive and transitive relation**  $\succeq$

$x \succeq y$        $x$  is at least as good as  $y$

$x \succ y$      $\Leftrightarrow$      $x \succeq y$  and not  $y \succeq x$   
 $x$  is preferred to  $y$  (**strict preference**)

$x \sim y$      $\Leftrightarrow$      $x \succeq y$  and  $y \succeq x$   
 $x$  and  $y$  are equally preferred (**indifference**)

### Cardinal preferences

Utility function  $u : \mathcal{E} \rightarrow \mathbb{R}$  [more generally  $u : \mathcal{E} \rightarrow V$  ordered scale]

More sophisticated models: interval orders, fuzzy preferences etc.



## Preferences: models and queries



Figure borrowed from [Brafman and Domshlak, 2007]



## Preferences: models and queries

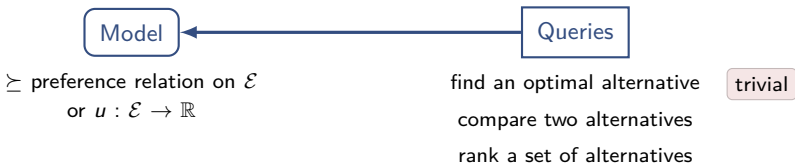


Figure borrowed from [Brafman and Domshlak, 2007]



## Preferences: models and queries

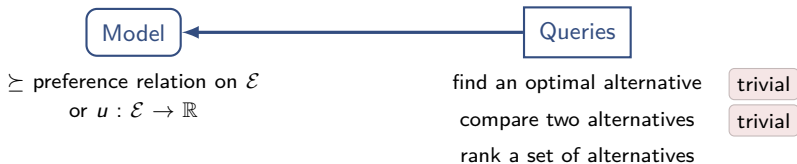


Figure borrowed from [Brafman and Domshlak, 2007]



## Preferences: models and queries

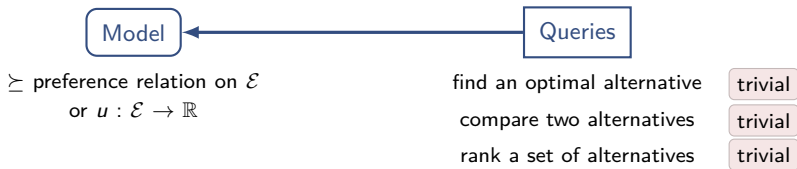


Figure borrowed from [Brafman and Domshlak, 2007]



## Conclusions

- + good news: everything is easy
- bad news: not much interesting to say



## Conclusions

- + good news: everything is easy
- bad news: not much interesting to say

Questions?



## Structure of the set of alternatives

A key question in individual and collective decision making: what is the *structure* of the set  $\mathcal{E}$  of alternatives?

Sometimes  $\mathcal{E}$  has a simple structure (and a small size):

- choose a transportation mode to come to the conference centre:  
 $\mathcal{E} = \{\text{metro, walk, bike}\}$
- choose the organizer of the next IJCAI:  
 $\mathcal{E} = \{\text{Angela Merkel, Dominique Strauss-Kahn, Jose Luis Zapatero}\}$

But sometimes,  $\mathcal{E}$  has a *combinatorial* structure.



## Example 1: a configuration problem

### *Buying a tour package on Internet*

- I'd like a return ticket Ithaca – Troy, or else Ithaca – Barcelona.
- if I go to Troy I prefer a palace; in Barcelona, a youth hostel.
- if the trip back is by boat, I'd prefer it with several stopovers.
- if the trip includes a flight, I prefer *Cyclope Airways*.
- if the trip back is by boat I prefer to avoid Charybd and Scylla, except if this saves me at least 50  $\phi$ .
- my budget is 600  $\phi$ . ( $\phi$  = Ithaca currency, not convertible).

*How can we express such preferences in an efficient way?*

*How can we find the best  $k$  solutions?*



## Example 2: multiple referenda

2 binary variables:

$S$  (build a swimming pool or not) and  $T$  (build a tennis court or not)

Voters' preferences

- **voters 1 and 2**  $\rightarrow s\bar{t} \succ \bar{s}t \succ \bar{s}\bar{t} \succ st$
- **voters 3 and 4**  $\rightarrow \bar{s}t \succ s\bar{t} \succ \bar{s}\bar{t} \succ st$
- **voter 5**  $\rightarrow st \succ s\bar{t} \succ \bar{s}t \succ \bar{s}\bar{t}$

*How will voters 1-4 express their preferences on  $\{s, \bar{s}\}$  and on  $\{t, \bar{t}\}$ ?*



## Example 2: multiple referenda

2 binary variables:

$S$  (build a swimming pool or not) and  $T$  (build a tennis court or not)

Voters' preferences

- voters 1 and 2  $\rightarrow s\bar{t} \succ \bar{s}t \succ \bar{s}\bar{t} \succ st$
- voters 3 and 4  $\rightarrow \bar{s}t \succ s\bar{t} \succ \bar{s}\bar{t} \succ st$
- voter 5  $\rightarrow st \succ s\bar{t} \succ \bar{s}t \succ \bar{s}\bar{t}$

*How will voters 1-4 express their preferences on  $\{s, \bar{s}\}$  and on  $\{t, \bar{t}\}$ ?*

Suppose they do it in an “optimistic” way :

- voters 1, 2 and 5:  $s$ ; voters 3 and 4:  $\bar{s} \Rightarrow$  decision =  $s$ ;
- voters 3, 4 and 5:  $t$ ; voters 1 and 2:  $\bar{t} \Rightarrow$  decision =  $t$ .



## Example 2: multiple referenda

2 binary variables:

$S$  (build a swimming pool or not) and  $T$  (build a tennis court or not)

Voters' preferences

- voters 1 and 2  $\rightarrow s\bar{t} \succ \bar{s}t \succ \bar{s}\bar{t} \succ st$
- voters 3 and 4  $\rightarrow \bar{s}t \succ s\bar{t} \succ \bar{s}\bar{t} \succ st$
- voter 5  $\rightarrow st \succ s\bar{t} \succ \bar{s}t \succ \bar{s}\bar{t}$

*How will voters 1-4 express their preferences on  $\{s, \bar{s}\}$  and on  $\{t, \bar{t}\}$ ?*

Suppose they do it in an “optimistic” way :

- voters 1, 2 and 5:  $s$ ; voters 3 and 4:  $\bar{s} \Rightarrow$  decision =  $s$ ;
- voters 3, 4 and 5:  $t$ ; voters 1 and 2:  $\bar{t} \Rightarrow$  decision =  $t$ .

$st$  is chosen, although it is the worst alternative for all voters but one.

*How should such a voting process be handled?*



## Example 3: combinatorial auctions

$\mathcal{O} = \{o_1, \dots, o_p\}$  set of objects

for each agent  $i$ ,  $V_i : 2^{\mathcal{O}} \rightarrow \mathbb{N}$

$V_i(X)$  = maximum price that  $i$  is ready to pay for the set of objects  $X$ .

if  $V_i$  additive for all  $i$ : then sell each object to its highest bidder

but  $V_i$  is generally *non-additive* :

- {left shoe}: 10 €; {right shoe}: 10 €; {left shoe, right shoe}: 50 €
- {horchata}: 2 €; {beer}: 3 €; {horchata, beer}: 4 €

optimal allocation  $\pi^*$ : maximizes the seller's revenue  $\sum_{i=1}^n V_i(\pi(i))$   
where  $\pi(i)$  is the set of objects allocated to agent  $i$

*How can bidders express their functions  $V_i$ ?*

*How can the seller determine  $\pi^*$ ?*



## Preferential dependencies

Examples 1 to 3 : the set of alternatives  $\mathcal{E}$  has a *combinatorial structure*

$$\mathcal{E} = D_1 \times \dots \times D_n$$

$D_i$  finite domain of values for variable  $X_i$

Existence of *preferential dependencies* between variables:

- if I go to Troy I prefer a palace; in Barcelona, a youth hostel
- if the swimming pool is built then I'd prefer the tennis court not to be built
- if I receive the left shoe then I'm ready to pay more for the right shoe

⇒ makes it hard to decompose a problem into independent subproblems.



## Combinatorial spaces...

### The combinatorial trap...

Two binary variables...

$x_1 \succ x_2 \succ x_1 x_2 \succ \emptyset \rightarrow 4$  subsets to compare



## Combinatorial spaces...

### The combinatorial trap...

Four binary variables...

$x_1x_2 \vee x_2x_3x_4 \vee x_1 \vee \emptyset \vee x_2 \vee x_1x_2x_3x_4 \vee x_1x_3 \vee x_2x_4 \vee x_3x_4 \vee x_1x_4 \vee$   
 $x_1x_3x_4 \vee x_2x_3 \vee x_4 \vee x_3 \vee x_1x_2x_4 \vee x_1x_2x_3 \rightarrow 15$  subsets



## Combinatorial spaces...

### The combinatorial trap...

Twenty binary variables...

$x_8 x_5 \wedge x_5 x_3 x_9 \wedge x_8 \wedge \emptyset \wedge x_5 \wedge x_8 x_5 x_3 x_9 \wedge x_8 x_3 \wedge x_5 x_9 \wedge x_3 x_9 \wedge x_8 x_9 \wedge$   
 $x_8 x_3 x_9 \wedge x_5 x_3 \wedge x_9 \wedge x_3 \wedge x_8 x_5 x_9 \wedge x_8 x_5 x_3 x_1 x_2 x_5 x_8 x_9 \wedge x_1 x_5 x_6 \wedge$   
 $x_7 \wedge x_2 x_3 x_4 x_5 x_6 x_7 x_8 \wedge x_1 x_2 x_3 x_4 x_5 \wedge x_1 x_3 \wedge x_2 \wedge x_1 x_3 x_7 x_9 \wedge x_1 x_5 \wedge$   
 $x_1 x_7 x_8 x_9 \wedge x_2 \wedge x_4 \wedge x_6 \wedge x_1 x_7 \wedge x_1 x_2 x_3 \wedge x_1 x_2 \wedge x_2 x_5 x_4 \wedge x_1 \wedge$   
 $x_2 \wedge x_1 x_2 x_5 x_4 \wedge x_1 x_5 \wedge x_2 x_4 \wedge x_5 x_4 \wedge x_1 x_4 \wedge x_1 x_5 x_4 \wedge x_2 x_5 \wedge x_4 \wedge$   
 $x_5 \wedge x_1 x_2 x_4 \wedge x_1 x_2 x_5 \wedge x_1 x_5 \wedge x_5 x_3 x_9 \wedge x_1 \wedge \emptyset \wedge x_5 \wedge x_1 x_5 x_3 x_9 \wedge$   
 $x_1 x_3 \wedge x_5 x_9 \wedge x_3 x_9 \wedge x_1 x_9 \wedge x_1 x_3 x_9 \wedge x_5 x_3 \wedge x_9 \wedge x_3 \wedge x_1 x_5 x_9 \wedge$   
 $x_1 x_5 x_3 x_9 x_6 x_5 x_1 x_9 \wedge x_9 x_5 x_6 \wedge x_7 \wedge x_6 x_3 x_4 x_5 x_6 x_7 x_1 \wedge x_9 x_6 x_3 x_4 x_5 \wedge$   
 $x_9 x_3 \wedge x_6 \wedge x_9 x_3 x_7 x_9 \wedge x_9 x_5 \wedge x_9 x_7 x_1 x_9 \wedge x_6 \wedge x_4 \wedge x_6 \wedge x_9 x_7 \wedge$   
 $x_9 x_6 x_3 \wedge x_9 x_6 \wedge x_6 x_5 x_4 \wedge x_9 \wedge x_6 \wedge x_9 x_6 x_5 x_4 \wedge x_9 x_5 \wedge x_6 x_4 \wedge x_5 x_4 \wedge$   
 $x_9 x_4 \wedge$

→ 1048575 subsets → the expression takes more than 12 days.



## The dilemma

- The expression of **preferential dependencies** is often necessary.
- **but** ... Representing and eliciting  $\succeq$  or  $u$  *in extenso* is unfeasible in practice.



## The dilemma

- The expression of **preferential dependencies** is often necessary.
- **but** ... Representing and eliciting  $\succeq$  or  $u$  *in extenso* is unfeasible in practice.

⇒ *languages for compact preference representation*



## Outline

- 1 Languages for compact preference representation**
- 2 Logical and bidding languages**
  - Propositional logic
  - Bidding languages
- 3 Graphical languages for ordinal preferences**
  - An introduction to graphical languages
  - Preferential independence
  - CP-nets
  - Outcome optimisation with CP-nets
  - CP-nets: extensions and variants
- 4 Graphical languages for cardinal preferences**
  - Additivity generalized
  - Some other languages
- 5 Collective decision making**
  - Voting
  - MultiAgent Resource Allocation
- 6 Conclusion**



## Preferences, languages

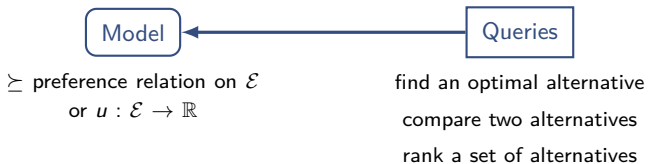


Figure borrowed from [Brafman and Domshlak, 2007]



## Preferences, languages

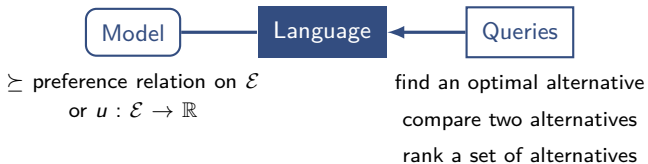


Figure borrowed from [Brafman and Domshlak, 2007]



## Preferences, languages

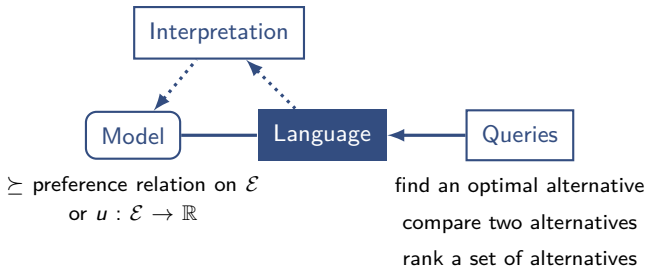


Figure borrowed from [Brafman and Domshlak, 2007]



## Representation languages

**Representation language** :  $\langle L, I_L \rangle$ , where

- $L$  language
- $I_L : \Phi \in L \mapsto$  preference relation  $\succeq_\Phi$  or utility function  $u_\Phi$  induced by  $\Phi$

**A prototypical language**: plain propositional logic

- $L$ : set of all propositional formulas built on a finite set of propositional symbols  $PS$ .
- $I_L : \varphi \mapsto u_\varphi$  defined by

$$\text{for all } \omega \in 2^{PS}, u_\varphi(\omega) = \left\{ \begin{array}{ll} +1 & \text{if } \omega \models \varphi \\ 0 & \text{if } \omega \models \neg\varphi \end{array} \right\}$$

or equivalently

- $I_L : \varphi \mapsto \succeq_\varphi$  defined by

$$\text{for all } \omega, \omega' \in 2^{PS}, \omega \succeq_\varphi \omega' \text{ iff } \omega \models \varphi \text{ or } \omega' \models \neg\varphi$$



## Representation languages

**Representation language** :  $\langle L, I_L \rangle$ , where

- $L$  language
- $I_L : \Phi \in L \mapsto$  preference relation  $\succeq_\Phi$  or utility function  $u_\Phi$  induced by  $\Phi$

**Another prototypical language**: the “explicit” representation.

- $L$ : set of all weak orders on  $D = D_1 \times \dots \times D_n$
- $I_L$ : identity

(and similarly for utility functions)



## Representation languages

*On which criteria can we evaluate the different languages?*

- **Expressive power:** what is the set of all preference structures expressible in the language?

*Example:* expressivity of plain propositional logic = set of all *dichotomous* preference relations



## Representation languages

*On which criteria can we evaluate the different languages?*

- **Expressive power:** what is the set of all preference structures expressible in the language?

*Example:* expressivity of plain propositional logic = set of all *dichotomous* preference relations

- **Succinctness:** (informally) language  $L_1$  is at least as succinct as language  $L_2$  is any preference structure expressible in  $L_2$  can be expressed in  $L_1$  *without any exponential growth of size.*



## Representation languages

*On which criteria can we evaluate the different languages?*

- **Expressive power:** what is the set of all preference structures expressible in the language?

*Example:* expressivity of plain propositional logic = set of all *dichotomous* preference relations

- **Succinctness:** (informally) language  $L_1$  is at least as succinct as language  $L_2$  is any preference structure expressible in  $L_2$  can be expressed in  $L_1$  *without any exponential growth of size.*
- **Computational complexity:** how hard is it to compare two alternatives or to find an optimal alternative when the preference structure is represented in  $L$ ?



## Representation languages

*On which criteria can we evaluate the different languages?*

- **Expressive power:** what is the set of all preference structures expressible in the language?

*Example:* expressivity of plain propositional logic = set of all *dichotomous* preference relations

- **Succinctness:** (informally) language  $L_1$  is at least as succinct as language  $L_2$  is any preference structure expressible in  $L_2$  can be expressed in  $L_1$  *without any exponential growth of size.*
- **Computational complexity:** how hard is it to compare two alternatives or to find an optimal alternative when the preference structure is represented in  $L$ ?
- **Easiness of elicitation**  
*Preference elicitation* = interaction with a user, so as to acquire her preferences, encoded in a language  $L$ .  
Is it easy to construct protocols for eliciting the agent's preferences in  $L$ ?



## More on expressive power

Representation language:  $\langle L, I_L \rangle$

*Expressive power of a language* = set of all preference structures that can be expressed in the language =  $I_L(L)$ .

$\langle L, I_L \rangle$  at least as expressive as  $\langle L', I_{L'} \rangle$  iff  $I_L(L) \supseteq I_{L'}(L')$ .

Examples :

- expressive power of plain propositional logic: all dichotomous utility functions (or all dichotomous preference relations)
- expressive power of explicit representation: all preference relations / utility functions

The explicit representation is more expressive than plain propositional logic.



## More on succinctness

*Relative* notion:

$L_1$  is at least as succinct as  $L_2$  if there exists  $F : L_2 \rightarrow L_1$  and a polynomial function  $\rho$  such that for all  $\Phi \in L_2$ :

- $I_{L_2}(\Phi) = I_{L_1}(F(\Phi))$ :  $\Phi$  and  $F(\Phi)$  induce the same preferences
- $|F(\Phi)| \leq \rho(|\Phi|)$ : the translation is succinct

Example:

- $L_1 =$  plain propositional logic
- $L_2 =$  explicit representation restricted to dichotomous preferences:  
 $L_2 = \{X \mid X \subseteq 2^{PS}\}$
- $L_3 =$  explicit representation language (non restricted to dichotomous preferences)

$L_1$  is strictly more succinct than  $L_2$ ; but  $L_1$  and  $L_3$  are incomparable.



## More on computational complexity

What is the computational complexity of the following problems *when the preferences on  $D$  are represented in the language  $L$* :

Given an input  $\Phi$  in the language  $L$ , ...

- DOMINANCE: and  $\vec{x}, \vec{y} \in D$ , do we have  $\vec{x} \succeq_{\Phi} \vec{y}$ ?
- OPTIMISATION: find the preferred option (or one of the preferred options)  $\vec{x}$ ;
- CONSTRAINED OPTIMISATION: and a subset  $C$ , possibly defined succinctly as well succinctly, find the preferred option (or one of the preferred options)  $\vec{x} \in C$ .

Examples:

- plain propositional logic: DOMINANCE in P, OPTIMISATION (constrained or not) NP-hard
- explicit representation: all problems in P (provided that  $C$  is also represented explicitly)



## More on elicitation

*Preference elicitation* = interaction with a user, so as to acquire her preferences, encoded in a language  $L$  (or more generally, so as to acquire enough information about her preferences for suggesting her an optimal decision).

Construction of elicitation protocols for some families of languages:

- exploiting preferential independencies so as to reduce the amount of information to elicit and the cognitive effort spent in communication;
- trade-off expressivity vs. elicitation complexity.



## Back to preferential dependencies

Expressing preferential dependencies: classes of languages

First dichotomy:

- **Logical and bidding languages:**
  - use logic to express preferential dependencies
  - a special kind of languages designed for auctions
- **Graphical languages:**
  - **graphical component** describing preferential dependencies between variables
  - collection of **local preferences** on single variables or small subsets of variables, compatible with the dependence structure.

Second dichotomy:

- languages for expressing **ordinal preferences**
- languages for expressing **cardinal preferences**

In this talk:

- a quick glimpse on logical and bidding languages
- a special focus on graphical languages: CP-nets, GAI-nets and extensions



## Acknowledgements

- AAI 2007 tutorial on **Representing, Eliciting, and Reasoning with Preferences** given by *Ronen I. Brafman and Carmel Domshlak*  
<http://iew3.technion.ac.il/~dcarmel/tutorial/pref-tutorial.pdf>
- **Tutorial on MultiAgent Resource Allocation** given by *Ulle Endriss and Nicolas Maudet* at the European Agent Systems Summer School 2006-2007  
<http://staff.science.uva.nl/~ulle/teaching/easss-2007/>
- **Tutorial on Fair Division** given by *Ulle Endriss* at the COST-ADT Doctoral School on COMSOC in 2010  
<http://staff.science.uva.nl/~ulle/teaching/cost-adt-2010/>
- **Lecture on Combinatorial Auctions** given by *Ulle Endriss* at the ILLC Amsterdam  
<http://staff.science.uva.nl/~ulle/teaching/comsoc/2009/slides/comsoc-auctions.pdf>



## Outline

- 1 **Languages for compact preference representation**
- 2 **Logical and bidding languages**
  - Propositional logic
  - Bidding languages
- 3 **Graphical languages for ordinal preferences**
  - An introduction to graphical languages
  - Preferential independence
  - CP-nets
  - Outcome optimisation with CP-nets
  - CP-nets: extensions and variants
- 4 **Graphical languages for cardinal preferences**
  - Additivity generalized
  - Some other languages
- 5 **Collective decision making**
  - Voting
  - MultiAgent Resource Allocation
- 6 **Conclusion**



## Binary variables

**Particular case:** binary variables  $\rightarrow D_i = \{\top, \perp\}$  for all  $i$ .

Can be used to represent subsets of elements.



## Binary variables

**Particular case:** binary variables  $\rightarrow D_i = \{\top, \perp\}$  for all  $i$ .

Can be used to represent subsets of elements.

A set of elements  $\mathcal{O} = \{o_1, \dots, o_p\} \rightarrow$  binary variables  $\{O_1, \dots, O_n\}$ , where each variable  $O_i$  stands for the presence or absence of  $o_i$ .

$\rightarrow$  each instantiation / interpretation represents a **subset**  $\pi$  of  $\mathcal{O}$

Example of application: allocation of indivisible goods

**Example:**  $o_1 \bar{o}_2 \bar{o}_3 o_4 \bar{o}_5$  represents subset  $\{o_1, o_4\}$ .



## Binary variables

**Particular case:** binary variables  $\rightarrow D_i = \{\top, \perp\}$  for all  $i$ .

Can be used to represent subsets of elements.

A set of elements  $\mathcal{O} = \{o_1, \dots, o_p\} \rightarrow$  binary variables  $\{O_1, \dots, O_n\}$ , where each variable  $O_i$  stands for the presence or absence of  $o_i$ .

$\rightarrow$  each instantiation / interpretation represents a **subset**  $\pi$  of  $\mathcal{O}$

Example of application: allocation of indivisible goods

**Example:**  $o_1 \bar{o}_2 \bar{o}_3 o_4 \bar{o}_5$  represents subset  $\{o_1, o_4\}$ .

### Properties:

- **Monotonic preferences:**  $\pi \subseteq \pi' \Rightarrow \pi \preceq \pi'$
- **(sub-)(super-)modular preferences:**  $u(A \cup B) (\leq) (\geq) = u(A) + u(B) - u(A \cap B)$
- **(sub-)(super-)additive preferences:** (sub-)(super-)modular +  $u(\emptyset) = 0$   
additive preferences  $\rightsquigarrow$  e.g.  $u(o_1 o_2 o_4) = u(o_1) + u(o_2) + u(o_4)$ .
- **Separable preferences:** if  $(X \cup Y) \cap Z = \emptyset$  then  $X \succ Y$  iff  $X \cup Z \succ Y \cup Z$ .  
 $\rightsquigarrow$  e.g.  $o_1 \preceq o_2 \Rightarrow o_1 o_4 \preceq o_2 o_4$ .



## Of logic and goals

**Logic-based languages** suit well when we have to deal with **binary variables** (e.g. resource allocation problems).

- A propositional syntax  $L_{\mathcal{O}} \dots$ 
  - set of propositional symbols  $\mathcal{O}$ ,
  - usual connectives  $\neg, \wedge, \vee$



## Of logic and goals

**Logic-based languages** suit well when we have to deal with **binary variables** (e.g. resource allocation problems).

- A propositional syntax  $L_{\mathcal{O}} \dots$ 
  - set of propositional symbols  $\mathcal{O}$ ,
  - usual connectives  $\neg, \wedge, \vee$

### Example

- $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}$ .
- Set of requests for one agent:
  - $\text{DVD} \wedge \left( (\text{printer} \wedge \text{monitor}) \vee \text{laptop} \right)$ ,
  - $\text{camera} \wedge \text{printer}$ .



## Dichotomous preferences...

What to do with all these goals ?



## Dichotomous preferences...

What to do with all these goals ?

A first (simplistic) example

### Example

Variables  $\mathcal{O} = \{o_1, o_2, o_3\}$

- $o_2 \wedge \neg o_1$

$$\{o_2\}, \{o_2, o_3\} \succ \emptyset, \{o_1\}, \{o_3\}, \{o_1, o_2\}, \{o_1, o_3\}, \{o_1, o_2, o_3\}$$

- $(o_1 \wedge o_2 \wedge \neg o_3) \vee (\neg o_1 \wedge o_2 \wedge o_3)$

$$\{o_1, o_2\}, \{o_2, o_3\} \succ \emptyset, \{o_1\}, \{o_2\}, \{o_3\}, \{o_1, o_3\}, \{o_1, o_2, o_3\}$$



## A bit more interesting...

- Goal bases (e.g.  $\{\varphi_1, \dots, \varphi_m\}$ ):
  - cardinality
  - inclusion
  - distances
- **Prioritized** goal bases (e.g.  $\{\langle \varphi_1, k_1 \rangle, \dots, \langle \varphi_m, k_m \rangle\}$ ):
  - discrimin
  - leximin
  - best-out
- **Weighted** goal bases



## A language based on weighted goals

Preference representation:

- A propositional language  $L_{\mathcal{O}}$ ...
  - a set of propositional symbols  $\mathcal{O}$ ,
  - the usual connectives  $\neg, \wedge, \vee$
- ...and some weights  $w \in \mathcal{V}$ .



## A language based on weighted goals

Preference representation:

- A propositional language  $L_{\mathcal{O}}$ ...
  - a set of propositional symbols  $\mathcal{O}$ ,
  - the usual connectives  $\neg, \wedge, \vee$
- ...and some weights  $w \in \mathcal{V}$ .

### Example

- $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}$ .
- Agent 1's requests:
  - $\langle \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}), 110 \rangle$ ,
  - $\langle \text{DVD}, -10 \rangle$ ,
  - $\langle \text{camera} \wedge \text{printer}, 50 \rangle$ .



## Individual aggregation

The utility is obtained by **aggregating** the weights of the satisfied formulas, using  $\oplus$ .

### Individual utility

Given an agent  $i$ , her requests  $\Delta_i$ , a set  $\pi$ , her individual utility is:

$$u_i(\pi) = \bigoplus \{w \mid \langle \varphi, w \rangle \in \Delta_i \text{ and } \pi \models \varphi\}.$$

Two reasonable choices for  $\oplus$  : + or max.



## Individual aggregation

### Example

- $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}$ .
- Agent 1's requests:
  - $\langle \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}), 110 \rangle$ ,
  - $\langle \text{DVD}, -10 \rangle$ ,
  - $\langle \text{camera} \wedge \text{printer}, 50 \rangle$ .



## Individual aggregation

### Example

•  $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}.$

• Agent 1's requests:

•  $\langle \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}), 110 \rangle,$

•  $\langle \text{DVD}, -10 \rangle,$

•  $\langle \text{camera} \wedge \text{printer}, 50 \rangle.$

Computation of individual utility ( $\oplus = +$ ) :

$$\pi_1 = \{ \text{DVD}, \text{printer}, \text{laptop}, \text{camera} \}$$



## Individual aggregation

### Example

•  $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}.$

• Agent 1's requests:

•  $\langle \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}), 110 \rangle,$

•  $\langle \text{DVD}, -10 \rangle,$

•  $\langle \text{camera} \wedge \text{printer}, 50 \rangle.$

Computation of individual utility ( $\oplus = +$ ) :

$$\pi_1 = \{ \text{DVD}, \text{printer}, \text{laptop}, \text{printer} \} \Rightarrow u_1(\pi_1) = \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}) = 110$$



## Individual aggregation

### Example

•  $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}$ .

• Agent 1's requests:

•  $\langle \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}), 110 \rangle$ ,

•  $\langle \text{DVD}, -10 \rangle$ ,

•  $\langle \text{camera} \wedge \text{printer}, 50 \rangle$ .

Computation of individual utility ( $\oplus = +$ ) :

$$\pi_1 = \{ \text{DVD}, \text{printer}, \text{laptop} \} \Rightarrow u_1(\pi_1) = 110 - 10$$



## Individual aggregation

### Example

•  $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}.$

• Agent 1's requests:

•  $\langle \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}), 110 \rangle,$

•  $\langle \text{DVD}, -10 \rangle,$

•  $\langle \text{camera} \wedge \text{printer}, 50 \rangle.$

Computation of individual utility ( $\oplus = +$ ) :

$$\pi_1 = \{ \text{DVD}, \text{printer}, \text{laptop}, \text{camera} \} \Rightarrow u_1(\pi_1) = 110 - 10 + \cancel{0}$$



## Individual aggregation

### Example

•  $\mathcal{O} = \{ \text{phone}, \text{laptop}, \text{camera}, \text{monitor}, \text{printer}, \text{DVD} \}$ .

• Agent 1's requests:

•  $\langle \text{DVD} \wedge ((\text{printer} \wedge \text{monitor}) \vee \text{laptop}), 110 \rangle$ ,

•  $\langle \text{DVD}, -10 \rangle$ ,

•  $\langle \text{camera} \wedge \text{printer}, 50 \rangle$ .

Computation of individual utility ( $\oplus = +$ ) :

$$\pi_1 = \{ \text{DVD}, \text{printer}, \text{laptop} \} \Rightarrow u_1(\pi_1) = 110 - 10 + 0 = 100$$



## Weighted goals: expressive power

- **Unrestricted language:** fully expressive.
- **Positive cubes only:** fully expressive (see Möbius transform).
- **Positive cubes and positive weights only:** only monotonic functions.
- **Literals only:** additive functions

See [Coste-Marquis et al., 2004, Chevalyere et al., 2006, Uckelman, 2008] for more details



**Chevalyere, Y., Endriss, U., and Lang, J. (2006).**

Expressive power of weighted propositional formulas for cardinal preference modelling.  
In *Proc. of KR-06*.



**Coste-Marquis, S., Lang, J., Liberatore, P., and Marquis, P. (2004).**

Expressive power and succinctness of propositional languages for preference representation.  
In *Proc. of KR-04*.



**Uckelman, J. (2008).**

*More Than the Sum of Its Parts. Compact Preference Representation of Combinatorial Domains.*  
PhD thesis, Universiteit van Amsterdam.



## Weighted logic: succinctness

- **Unrestricted language:** fully expressive.
- **Positive cubes only:** fully expressive.



## Weighted logic: succinctness

- **Unrestricted language:** fully expressive.
- **Positive cubes only:** fully expressive.

But... Unrestricted language strictly more succinct than the restriction to positive cubes only.

### Example

$u$  defined as  $\langle o_1 \vee o_2 \vee \dots \vee o_n, 1 \rangle$ .



## Weighted logic: succinctness

- **Unrestricted language:** fully expressive.
- **Positive cubes only:** fully expressive.

But... Unrestricted language strictly more succinct than the restriction to positive cubes only.

### Example

$u$  defined as  $\langle o_1 \vee o_2 \vee \dots \vee o_n, 1 \rangle$ .

With **positive cubes only**:

- $\langle o_1, 1 \rangle, \dots, \langle o_n, 1 \rangle$
- $\langle o_1 \wedge o_2, -1 \rangle, \langle o_1 \wedge o_3, -1 \rangle, \dots, \langle o_{n-1} \wedge o_n, -1 \rangle$
- $\langle o_1 \wedge o_2 \wedge o_3, 2 \rangle, \langle o_1 \wedge o_2 \wedge o_4, 2 \rangle, \dots, \langle o_{n-2} \wedge o_{n-1} \wedge o_n, 2 \rangle$
- ...



## Dominance and optimisation

- [DOMINANCE] can be solved in **polynomial time**.
- [OPTIMISATION] is **NP-hard** in the general case.
- [OPTIMISATION] is **polynomial** for the monotonic fragment of weighted logic (no negation, positive weights).



## Auctions

### A special multiagent resource allocation problem:

- Numerical preferences: **money**.
- **Usual criterion**: maximize the revenue of the auctioneer.



## Auctions

### A special multiagent resource allocation problem:

- Numerical preferences: **money**.
- **Usual criterion**: maximize the revenue of the auctioneer.

### Classical auctions:

- Additive preferences.
- Each object is sold to the bidder proposing the highest price.



## Auctions

### A special multiagent resource allocation problem:

- Numerical preferences: **money**.
- **Usual criterion**: maximize the revenue of the auctioneer.

### Classical auctions:

- Additive preferences.
- Each object is sold to the bidder proposing the highest price.

If the bidder has:

- **supermodular preferences** (complements → left and right shoes);
- **submodular preferences** (substitutes → two different pairs of shoes);

*how should she bid in a traditional auction ?*



## Bidding languages

**Combinatorial auctions** are auctions where agents can bid on **subsets** of objects. . .

*Compact language needed  
to allow bidders to transmit their valuations to the auctioneer.*



## Bidding languages

**Combinatorial auctions** are auctions where agents can bid on **subsets** of objects. . .

*Compact language needed  
to allow bidders to transmit their valuations to the auctioneer.*

- Bids are combinations of atomic bids  $b = \langle \pi, w \rangle$ : bundle/price. The valuation of a bundle  $\pi'$  is  $w$  if  $\pi \subseteq \pi'$ , and 0 otherwise.
- In the XOR-language, atomic bids by the same bidder are assumed to be mutually exclusive:

$$(b_1 \text{ XOR } b_2)(\pi) = \max\{b_1(\pi), b_2(\pi)\}$$

- In the OR-language, the valuation is taken to be the maximal value that can be obtained by accepting disjoint bids

$$(b_1 \text{ OR } b_2)(\pi) = \max_{\pi' \subseteq \pi} \{b_1(\pi'), b_2(\pi \setminus \pi')\}$$

See [Cramton et al., 2006] for more details



## Examples

**XOR language:**  $(\pi_1, w_1) \text{ XOR } \dots \text{ XOR } (\pi_n, w_n)$

$(\{\text{DVD}, \text{hard drive}, \text{printer}\}, 400\text{€}) \text{ XOR } (\{\text{camera}, \text{printer}\}, 700\text{€}) \text{ XOR } (\{\text{phone}\}, 100\text{€})$

---

$\{\text{DVD}, \text{hard drive}, \text{printer}, \text{camera}, \text{phone}\} \Rightarrow u =$



## Examples

**XOR language:**  $(\pi_1, w_1) \text{ XOR } \dots \text{ XOR } (\pi_n, w_n)$

$(\{\text{DVD}, \text{hard drive}, \text{printer}\}, 400\text{€}) \text{ XOR } (\{\text{camera}, \text{printer}\}, 700\text{€}) \text{ XOR } (\{\text{phone}\}, 100\text{€})$

---

$\{\text{DVD}, \text{hard drive}, \text{printer}, \text{camera}, \text{phone}\} \Rightarrow u = 700\text{€}$



## Examples

**OR language:**  $(\pi_1, w_1) \text{ OR } \dots \text{ OR } (\pi_n, w_n)$

$(\{\text{DVD}, \text{hard drive}, \text{printer}\}, 400\text{€}) \text{ OR } (\{\text{camera}, \text{printer}\}, 700\text{€}) \text{ OR } (\{\text{phone}\}, 100\text{€})$

---

$\{\text{DVD}, \text{hard drive}, \text{printer}, \text{camera}, \text{phone}\} \Rightarrow u =$



## Examples

**OR language:**  $(\pi_1, w_1)$  OR ... OR  $(\pi_n, w_n)$

$(\{\text{DVD}, \text{hard drive}, \text{printer}\}, 400\text{€})$  OR  $(\{\text{camera}, \text{printer}\}, 700\text{€})$  OR  $(\{\text{phone}\}, 100\text{€})$

---

$\{\text{DVD}, \text{hard drive}, \text{printer}, \text{camera}, \text{phone}\} \Rightarrow u = 800\text{€}$



## Other languages

- Combinations of OR and XOR (OR-of-XOR / XOR-of-OR...)
- The **OR\*-language** is like the OR-language, but dummy items can be used to express exclusiveness constraints. Example:

$\langle \{a, dummy\}, 3 \rangle$  OR  $\langle \{b, dummy\}, 3 \rangle$  OR  $\langle \{a, b, dummy\}, 5 \rangle$ .

- Weighted logic.



## Bidding language: features

### Expressive power:

- XOR can represent all (monotonic) valuations.
- OR can only represent supermodular valuations.

### Succinctness:

- OR is strictly more succinct than XOR.



## Bidding language: features

### Expressive power:

- XOR can represent all (monotonic) valuations.
- OR can only represent supermodular valuations.

### Succinctness:

- OR is strictly more succinct than XOR.

### Proof sketch:

- The XOR representation of every supermodular function is also an OR representation of the same function.
- Function  $u(\pi) = |\pi|$  needs an exponential size XOR formula (whereas it only needs a linear size OR formula).



## Bidding languages: Complexity

- Compute the utility of a given XOR bundle: **polynomial**.
- Compute the utility of a given OR bundle: **NP-hard** (for the associated decision problem) !



## Bidding languages: Complexity

- Compute the utility of a given XOR bundle: **polynomial**.
- Compute the utility of a given OR bundle: **NP-hard** (for the associated decision problem) !

### Proof sketch

- Membership to NP is easy.
- Hardness by reduction from [SET PACKING]

#### [Set Packing]

- **Input:** Collection  $\mathcal{C}$  of finite sets and  $K \in \mathbb{N}$ .
- **Question:** Is there a collection of disjoint sets  $\mathcal{C}' \subseteq \mathcal{C}$  s.t.  $|\mathcal{C}'| > K$  ?



## Outline

- 1 Languages for compact preference representation**
- 2 Logical and bidding languages**
  - Propositional logic
  - Bidding languages
- 3 Graphical languages for ordinal preferences**
  - An introduction to graphical languages
  - Preferential independence
  - CP-nets
  - Outcome optimisation with CP-nets
  - CP-nets: extensions and variants
- 4 Graphical languages for cardinal preferences**
  - Additivity generalized
  - Some other languages
- 5 Collective decision making**
  - Voting
  - MultiAgent Resource Allocation
- 6 Conclusion**



## A short introduction

*[Probabilistic] graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering – uncertainty and complexity.*

*The Economist, 03/22/2001*



## A short introduction

*[Probabilistic] graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering – uncertainty and complexity.*

*The Economist, 03/22/2001*

More generally, a natural tool for dealing with complexity and:

- **uncertainty**: Bayesian networks, Markov random fields [Pearl, 1988]
- **decision under uncertainty**: influence diagrams [Howard and Matheson, 1984]
- **(un)satisfaction** → constraint networks [Montanari, 1974]
- **costs** → valued constraint networks (cost function networks) [Schiex et al., 1995]
- **ordinal preferences** → CP-nets... [Boutilier et al., 2004a]
- **utilities** → GAI-nets... [Bacchus and Grove, 1995, Gonzales and Perny, 2004]



## A short introduction

Graphical models / languages are based upon:

- a **graphical component** describing (un)directed dependencies between variables
- collection of **local ordinal or numerical statements** on single variables or small subsets of variables, compatible with the dependence structure

Why graphs ?

- A very natural and intuitive way of representing interactions between variables
- Efficient algorithms



## Preferential independence

**Conditional preferential independence** [Keeney and Raiffa, 1976]:

$\mathcal{V}$  set of variables;  $\{\mathcal{X}, \mathcal{Y}, \mathcal{Z}\}$  partition of  $\mathcal{V}$ .

$D_{\mathcal{X}} = \times_{x_i \in \mathcal{X}} D_i$  etc.

$\mathcal{X}$  is preferentially independent from  $\mathcal{Y}$  (given  $\mathcal{Z}$ ) iff

$$\text{for all } x, x' \in D_{\mathcal{X}}, y, y' \in D_{\mathcal{Y}}, z \in D_{\mathcal{Z}}, \\ (x, y, z) \succeq (x', y, z) \text{ iff } (x, y', z) \succeq (x', y', z)$$

*given any fixed value  $z$  of  $\mathcal{Z}$ , the preferences over the possible values of  $\mathcal{X}$  are independent from the value of  $\mathcal{Y}$*

### Example

Two binary variables  $A, B$  with domains  $\{a, \bar{a}\}, \{b, \bar{b}\}$

Preference relation:  $ab \succ a\bar{b} \succ \bar{a}\bar{b} \succ \bar{a}b$



## Preferential independence

**Conditional preferential independence** [Keeney and Raiffa, 1976]:

$\mathcal{V}$  set of variables;  $\{\mathcal{X}, \mathcal{Y}, \mathcal{Z}\}$  partition of  $\mathcal{V}$ .

$D_{\mathcal{X}} = \times_{x_i \in \mathcal{X}} D_i$  etc.

$\mathcal{X}$  is preferentially independent from  $\mathcal{Y}$  (given  $\mathcal{Z}$ ) iff

$$\text{for all } x, x' \in D_{\mathcal{X}}, y, y' \in D_{\mathcal{Y}}, z \in D_{\mathcal{Z}}, \\ (x, y, z) \succeq (x', y, z) \text{ iff } (x, y', z) \succeq (x', y', z)$$

*given any fixed value  $z$  of  $\mathcal{Z}$ , the preferences over the possible values of  $\mathcal{X}$  are independent from the value of  $\mathcal{Y}$*

### Example

Two binary variables  $A, B$  with domains  $\{a, \bar{a}\}, \{b, \bar{b}\}$

Preference relation:  $ab \succ a\bar{b} \succ \bar{a}\bar{b} \succ \bar{a}b$

$A$  preferentially independent from  $B$



## Preferential independence

**Conditional preferential independence** [Keeney and Raiffa, 1976]:

$\mathcal{V}$  set of variables;  $\{\mathcal{X}, \mathcal{Y}, \mathcal{Z}\}$  partition of  $\mathcal{V}$ .

$D_{\mathcal{X}} = \times_{x_i \in \mathcal{X}} D_i$  etc.

$\mathcal{X}$  is preferentially independent from  $\mathcal{Y}$  (given  $\mathcal{Z}$ ) iff

$$\text{for all } x, x' \in D_{\mathcal{X}}, y, y' \in D_{\mathcal{Y}}, z \in D_{\mathcal{Z}}, \\ (x, y, z) \succeq (x', y, z) \text{ iff } (x, y', z) \succeq (x', y', z)$$

*given any fixed value  $z$  of  $\mathcal{Z}$ , the preferences over the possible values of  $\mathcal{X}$  are independent from the value of  $\mathcal{Y}$*

### Example

Two binary variables  $A, B$  with domains  $\{a, \bar{a}\}, \{b, \bar{b}\}$

Preference relation:  $ab \succ a\bar{b} \succ \bar{a}\bar{b} \succ \bar{a}b$

$A$  preferentially independent from  $B$

$B$  preferentially dependent on  $A$



## Outline

- 1 Languages for compact preference representation**
- 2 Logical and bidding languages**
  - Propositional logic
  - Bidding languages
- 3 Graphical languages for ordinal preferences**
  - An introduction to graphical languages
  - Preferential independence
  - CP-nets
  - Outcome optimisation with CP-nets
  - CP-nets: extensions and variants
- 4 Graphical languages for cardinal preferences**
  - Additivity generalized
  - Some other languages
- 5 Collective decision making**
  - Voting
  - MultiAgent Resource Allocation
- 6 Conclusion**



## CP-nets: definition

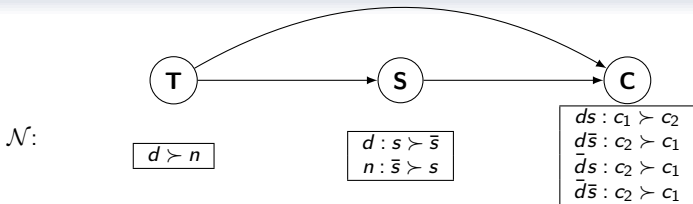
- first reference: [Boutilier et al., 1999]
- one of the most popular preference representation languages on combinatorial domains.
- based on conditional preferential independence.

A CP-net is composed of

- a *directed graph* representing the preferential dependencies between variables
- for each variable, a *conditional preference table* expressing the local preference on the values of its domain given the possible combination of values of its parents.



## CP-nets: an example



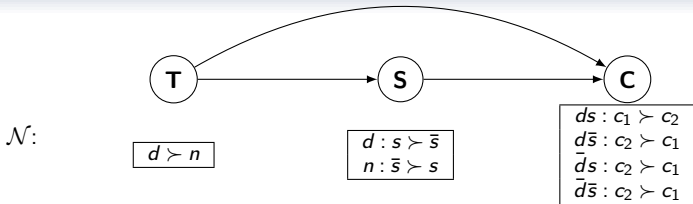
**T** independent of **S** and **C**; **S** independent of **C**

$d \succ n$  |  $T = d$  preferred to  $T = n$   
everything else ( $S, C$ ) being equal (**ceteris paribus**)

$dsc_1 \succ nsc_1$ ;  $dsc_2 \succ nsc_2$ ;  $d\bar{s}c_1 \succ n\bar{s}c_1$ ;  $d\bar{s}c_2 \succ n\bar{s}c_2$



## CP-nets: an example



**T** independent of **S** and **C**; **S** independent of **C**

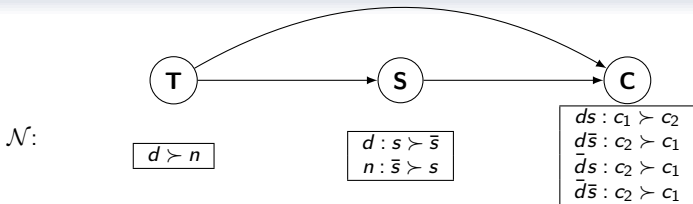
$d : s \succ \bar{s}$  | if  $T = d$   
 then  $S = s$  preferred to  $S = \bar{s}$   
 everything else (**C**) being equal (**ceteris paribus**)

$$dsc_1 \succ d\bar{s}c_1; \quad dsc_2 \succ d\bar{s}c_2$$

and likewise:  $n\bar{s}c_1 \succ nsc_1; \quad n\bar{s}c_2 \succ \bar{x}sc_2$



## CP-nets: an example



**T** independent of **S** and **C**; **S** independent of **C**

$$\succ_T: \{dsc_1 \succ nsc_1, dsc_2 \succ nsc_2, d\bar{s}c_1 \succ n\bar{s}c_1, d\bar{s}c_2 \succ n\bar{s}c_2\};$$

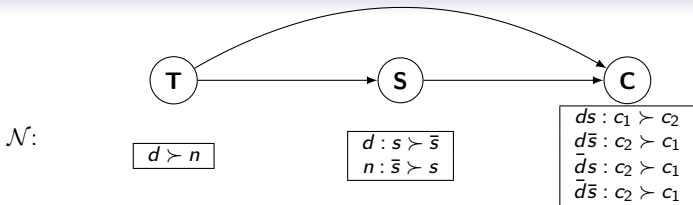
$$\succ_S: \{dsc_1 \succ d\bar{s}c_1, dsc_2 \succ d\bar{s}c_2, n\bar{s}c_1 \succ nsc_1, n\bar{s}c_2 \succ nsc_2\};$$

$$\succ_C: \{dsc_1 \succ dsc_2, d\bar{s}c_2 \succ d\bar{s}c_1, nsc_2 \succ nsc_1, n\bar{s}c_2 \succ n\bar{s}c_1\}$$

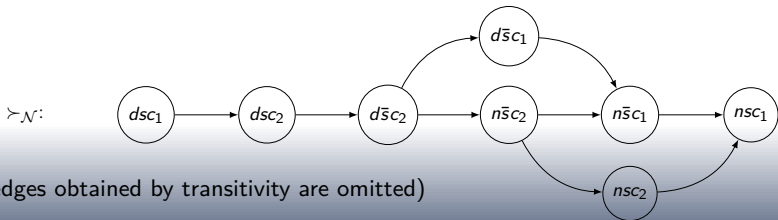
$$\begin{aligned} \succ_{\mathcal{N}} &= \text{preference relation induced by the CP-net } \mathcal{N} \\ &= \text{transitive closure of } \succ_T \cup \succ_S \cup \succ_C \end{aligned}$$



## CP-nets: an example



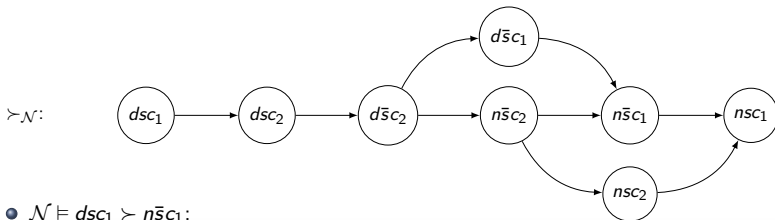
**T** independent of **S** and **C**; **S** independent of **C**





## CP-nets: semantics

- $\succ_N$  generally not complete
- the complete preference relations extending  $\succ_N$  can be viewed as *possible models* of the user's true preference relation
- *consequence* in CP-nets:  $\mathcal{N} \models o \succ o'$  if  $o \succ o'$  holds in all complete preference relations extending  $\succ_N$

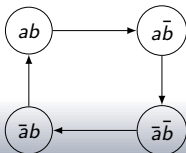
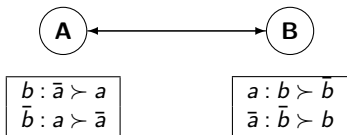


- $\mathcal{N} \models dsc_1 \succ n\bar{s}c_1$ ;
- $\mathcal{N} \not\models d\bar{s}c_1 \succ n\bar{s}c_2$  and  $\mathcal{N} \not\models n\bar{s}c_2 \succ d\bar{s}c_1$



## CP-nets: consistency

- a CP-net  $\mathcal{N}$  is consistent if its associated preference relation  $\succ_{\mathcal{N}}$  is acyclic
- if the dependency graph  $G$  of  $\mathcal{N}$  is acyclic then  $\mathcal{N}$  is consistent
- this is not always true if  $G$  is cyclic:





## Outline

### 1 Languages for compact preference representation

### 2 Logical and bidding languages

Propositional logic

Bidding languages

### 3 Graphical languages for ordinal preferences

An introduction to graphical languages

Preferential independence

CP-nets

Outcome optimisation with CP-nets

CP-nets: extensions and variants

### 4 Graphical languages for cardinal preferences

Additivity generalized

Some other languages

### 5 Collective decision making

Voting

MultiAgent Resource Allocation

### 6 Conclusion



## Dominance and optimisation

Main purposes of a preference representation language: answer queries such as

- *comparison*: given two outcomes, determine if one of them dominates the other;
- *optimisation*: determine some undominated outcome.
- *outcome optimality*: given an outcome, determine if it is undominated.
- *consistency*: determine if the CP-net is consistent.



## Outcome optimisation

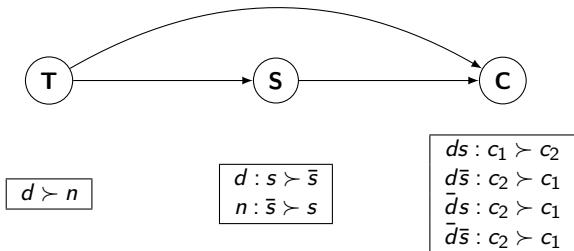
When  $G$  is acyclic:

- the CP-net  $\mathcal{N}$  is consistent;
- there is a unique nondominated (and dominating) outcome
- this outcome can be found in polynomial time, using the *forward sweep algorithm*



## Outcome optimisation

**Forward sweep:** consider variables in sequence in some order compatible with  $G$ , and choose the best value given the values of parents variables (already assigned).



- *step 1:* preferred value of  $T$ :  $d \Rightarrow T := d$
- *step 2:* preferred value of  $S$  given  $T = d$ :  $s \Rightarrow S := s$
- *step 3:* preferred value of  $C$  given  $T = d, S = s$ :  $c_1 \Rightarrow C := c_1$

optimal outcome:  $dsc_1$



## Outcome optimisation

Consequence of forward sweep: when  $G$  is acyclic, optimisation is in P.

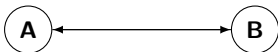
More generally:

	$G$ hypertree	$G$ acyclic	any $G$
optimisation	P	in P	NP-hard
comparison	P	NP-hard (in NP?)	PSPACE-complete
outcome optimality	P	P	P
consistency	trivial	trivial	PSPACE-complete



## Outcome optimisation

General case: translation into a *propositional satisfiability problem*  
[Domshlak et al., 2006, Brafman and Dimopoulos, 2004]



$$\begin{array}{l} b : a \succ \bar{a} \\ \bar{b} : \bar{a} \succ a \end{array}$$

$$\begin{array}{l} a : b \succ \bar{b} \\ \bar{a} : \bar{b} \succ b \end{array}$$

every entry  $u : x \succ \bar{x}$  (resp.  $u : \bar{x} \succ x$ ) of every table  
is translated into  $u \rightarrow x$  (resp.  $u \rightarrow \neg x$ )

$$\Phi_{\mathcal{N}} = (b \rightarrow a) \wedge (\neg b \rightarrow \neg a) \wedge (a \rightarrow b) \wedge (\neg a \rightarrow \neg b) \equiv (a \leftrightarrow b)$$

models of  $\Phi_{\mathcal{N}} = \{ab, \bar{a}\bar{b}\} =$  undominated outcomes of  $\mathcal{N}$

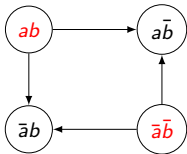


## Outcome optimisation



$$\begin{array}{l} b : a \succ \bar{a} \\ \bar{b} : \bar{a} \succ a \end{array}$$

$$\begin{array}{l} a : b \succ \bar{b} \\ \bar{a} : \bar{b} \succ b \end{array}$$

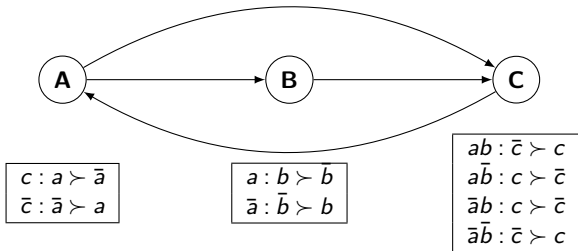


undominated outcomes of  $\mathcal{N}$ :  $\{ab, \bar{a}\bar{b}\}$



## Outcome optimisation

$\Phi_{\mathcal{N}}$  maybe consistent even when  $\mathcal{N}$  is inconsistent



$\succ_{\mathcal{N}}$  has a cycle:  $abc \succ \bar{a}bc \succ \bar{a}\bar{b}\bar{c} \succ ab\bar{c} \succ abc$

$$\Phi_{\mathcal{N}} \equiv (a \leftrightarrow c) \wedge (b \leftrightarrow a) \wedge (c \leftrightarrow (a \leftrightarrow b)) \equiv \neg a \wedge \neg b \wedge c$$

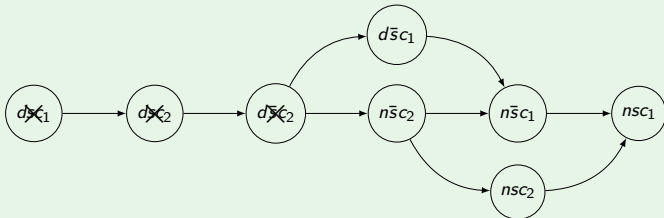
models of  $\Phi_{\mathcal{N}} = \{\bar{a}\bar{b}c\}$ :  $\bar{a}\bar{b}c$  is non-dominated.



## Constrained optimisation

- sometimes not all outcomes are feasible.
- *constrained CP-net*: CP-net  $\mathcal{N}$  + set of constraints  $\Gamma$ ;
- find an outcome  $\alpha$  both feasible and undominated, *i.e.*, such that there is no feasible outcome  $\beta$  such that  $\beta \succ_{\mathcal{N}} \alpha$  [Boutilier et al., 2004b].

### Example



Constraint:  $\Gamma = \{T = d \Rightarrow S = \bar{s}, C = c_2 \Rightarrow T = n\}$

Undominated feasible outcomes:  $d\bar{s}c_1$  and  $n\bar{s}c_2$ .

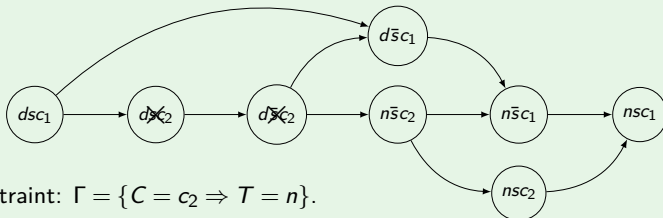


## Constrained optimisation

A different way of defining optimal solutions for constrained CSPs  
[Prestwich et al., 2004] :

- $\alpha$  dominates  $\beta$  if there is a flipping sequence from  $\alpha$  to  $\beta$  that goes through feasible outcomes only;
- again we look for non-dominated feasible outcomes.

### Example



Constraint:  $\Gamma = \{C = c_2 \Rightarrow T = n\}$ .

- according to [Prestwich et al., 2004]:  $dsc_1$  and  $n\bar{s}c_2$  are non-dominated;
- according to [Boutilier et al., 2004b]: only  $dsc_1$  is non-dominated.



## Outline

### 1 Languages for compact preference representation

### 2 Logical and bidding languages

Propositional logic

Bidding languages

### 3 Graphical languages for ordinal preferences

An introduction to graphical languages

Preferential independence

CP-nets

Outcome optimisation with CP-nets

CP-nets: extensions and variants

### 4 Graphical languages for cardinal preferences

Additivity generalized

Some other languages

### 5 Collective decision making

Voting

MultiAgent Resource Allocation

### 6 Conclusion



## TCP-nets

- [Brafman et al., 2006]
- enrich CP-nets by allowing importance statements between variables

A TCP-net contains

- conditional preference statements just like in CP-nets
- unconditional importance statements such as  $A \triangleright B$  ( $A$  is more important than  $B$ )
- conditional importance statements such as  $A = a : B \triangleright C$  (if  $A = a$  then  $B$  is more important than  $C$ )

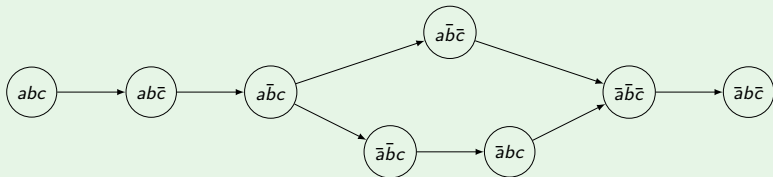
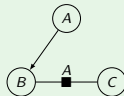


## TCP-nets

### Example

Importance statements:  $a : B \triangleright C$   
 $\bar{a} : C \triangleright B$

Preference statements:  $a \succ \bar{a}$      $a : b \succ \bar{b}$      $c \succ \bar{c}$   
 $\bar{a} : \bar{b} \succ b$





## CP-theories

[Wilson, 2004]

Example:

if  $a$  then  $b \succ \bar{b}$  whatever the value of  $C$ ,  
*ceteris paribus* (the value of  $D$ , etc. being fixed)

This preference statement entails

$$\begin{array}{cccc} abcd \succ a\bar{b}cd & abcd \succ a\bar{b}\bar{c}d & ab\bar{c}d \succ a\bar{b}\bar{c}d & a\bar{b}\bar{c}d \succ a\bar{b}\bar{c}\bar{d} \\ abcd \succ a\bar{b}c\bar{d} & abcd \succ a\bar{b}c\bar{d} & ab\bar{c}\bar{d} \succ a\bar{b}\bar{c}\bar{d} & a\bar{b}\bar{c}\bar{d} \succ a\bar{b}\bar{c}\bar{d} \end{array}$$

CP-theories are more general than TCP-nets (and *a fortiori*, than CP-nets);

- [Wilson, 2009]: more general language allowing preference statements of the form  
if  $a$  then  $bc \succ \bar{b}\bar{c}$  (...)
- [Bienvenu et al., 2010]: even more general language allowing preference statements between arbitrary propositional formulas given a set of fixed formulas.



## CI-nets

- [Bouveret et al., 2009]
- allow to express importance statements between sets of variables

Example:

if  $a$  and  $\bar{b}$  then  $C, D, E$  together are more important than  $F, G$  together,  
*ceteris paribus*

CI-networks are useful for expressing preferences over sets of goods (compact preference language for fair division)



## CI-nets: the language

### Conditional importance statement

$\mathcal{S}^+, \mathcal{S}^- : \mathcal{S}_1 \triangleright \mathcal{S}_2$  (with  $\mathcal{S}^+, \mathcal{S}^-, \mathcal{S}_1$  and  $\mathcal{S}_2$  pairwise-disjoint).

**Example:**  $\overline{ad} : b \triangleright ce$  implies for example  $ab \succ ace$ ,  $abfg \succ acefg$ , ...

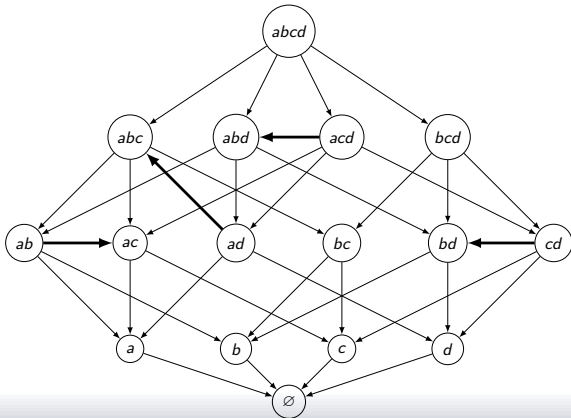
### CI-net

A CI-net on  $\mathcal{V}$  is a set  $\mathcal{N}$  of conditional importance statements on  $\mathcal{V}$ .



## Semantics

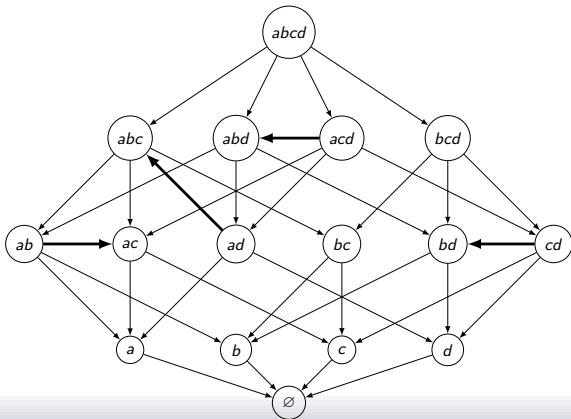
A CI-net of 4 objects  $\{a, b, c, d\}$ :  $\{a : d \triangleright bc, \overline{ad} : b \triangleright c, d : c \triangleright b\}$





## Semantics

A CI-net of 4 objects  $\{a, b, c, d\}$ :  $\{a : d \triangleright bc, \overline{ad} : b \triangleright c, d : c \triangleright b\}$



Induced preference relation  $\succ_{\mathcal{N}}$ : the smallest preference **monotonic** relation compatible with all CI-statements.



## More on CI-nets

- CI-nets can express all strict monotonic preference relations on  $2^{\mathcal{V}}$ .
- characterization of dominance by flipping sequences
- dominance and satisfiability in the general case PSPACE-complete; in P for precondition-free, singleton-comparing CI-statements (such as  $\{a \triangleright c, b \triangleright c, e \triangleright d\}$ ).



## Outline

### 1 Languages for compact preference representation

### 2 Logical and bidding languages

Propositional logic

Bidding languages

### 3 Graphical languages for ordinal preferences

An introduction to graphical languages

Preferential independence

CP-nets

Outcome optimisation with CP-nets

CP-nets: extensions and variants

### 4 Graphical languages for cardinal preferences

Additivity generalized

Some other languages

### 5 Collective decision making

Voting

MultiAgent Resource Allocation

### 6 Conclusion



## Outline

### 1 Languages for compact preference representation

### 2 Logical and bidding languages

Propositional logic

Bidding languages

### 3 Graphical languages for ordinal preferences

An introduction to graphical languages

Preferential independence

CP-nets

Outcome optimisation with CP-nets

CP-nets: extensions and variants

### 4 Graphical languages for cardinal preferences

Additivity generalized

Some other languages

### 5 Collective decision making

Voting

MultiAgent Resource Allocation

### 6 Conclusion



## Back to binary variables. . .

**Additive** utility functions are probably the simplest and the most intuitive ones. If  $\mathcal{O}$  is of size  $p$ , an additive utility function is only defined by  $p$  weights.

**Example:**  $u = 3o_1 + 7o_2 - 2o_3$  is an additive function.  $u(\{o_1, o_2\}) = 10$ .



## Back to binary variables. . .

**Additive** utility functions are probably the simplest and the most intuitive ones. If  $\mathcal{O}$  is of size  $p$ , an additive utility function is only defined by  $p$  weights.

**Example:**  $u = 3o_1 + 7o_2 - 2o_3$  is an additive function.  $u(\{o_1, o_2\}) = 10$ .

But. . . Additive utility functions are unable to represent dependencies (synergies) between variables (objects).

*Is there a way to represent these dependencies while keeping the representation not too big ?*



## The $k$ -additive form

- A utility function is called  **$k$ -additive** iff the utility assigned to a bundle  $\pi$  can be represented as the sum of basic utilities assigned to subsets of  $\pi$  with cardinality  $\leq k$  (limited synergies).
- The  **$k$ -additive form** of representing utility functions:

$$u(\pi) = \sum_{\pi' \subseteq \pi} \alpha^{\pi'} \text{ with } \alpha^{\pi'} = 0 \text{ whenever } |\pi'| > k$$

**Example:**  $u = 3o_1 + 7o_2 - 2o_2o_3$  is a 2-additive function

- Specifying a utility function = specifying the **coefficients**  $\alpha^\pi$  for bundles  $\pi \subseteq \mathcal{O}$ .
- $\alpha^\pi$  = additional benefit from having objects in  $\pi$  **together**.
  - $\alpha^\pi > 0 \Rightarrow$  objects in  $\pi$  are **complements**
  - $\alpha^\pi < 0 \Rightarrow$  objects in  $\pi$  are **substitutes**
- **Remark:**  $k$ -additive form = weighted propositional language restricted to **cubes only**



## Example

$$\mathcal{O} = \{o_1, o_2, o_3, o_4\}$$

$$u = 2o_1 + 3o_2 + 2o_2o_3 + 2o_3o_4$$

	$\emptyset$	$o_1$	$o_2$	$o_3$	$o_4$	$o_1o_2$	$o_1o_3$	$o_1o_4$	$o_2o_3$	$o_2o_4$
$u$	0	2	3	0	0	5	2	2	5	3

	$o_3o_4$	$o_1o_2o_3$	$o_1o_2o_4$	$o_2o_3o_4$	$o_1o_2o_3o_4$
$u$	2	7	5	7	9



## The $k$ -additive form – expressivity

The  $k$ -additive form is **fully expressive**, if we choose  $k$  large enough:

### Proposition

Any utility function is representable in the  $k$ -additive form for some  $k \leq |\mathcal{O}|$ .

**Proof:** For any utility function  $u$ , we can define coefficients  $\alpha^\pi$ :

- $\alpha^\emptyset = u(\emptyset)$ ;
- $\alpha^\pi = u(\pi) - \sum_{\pi' \subsetneq \pi} \alpha^{\pi'}$  for all  $\pi \subseteq \mathcal{O}$ .

The function  $u \mapsto (\pi \mapsto \alpha^\pi)$  is called the **Möbius transform** of  $u$  [Grabisch, 1997].



**Grabisch, M. (1997).**

*k*-order additive discrete fuzzy measure and their representation.  
*Fuzzy Sets and Systems*, 92:167–189.



## Explicit vs. $k$ -additive Form

Among the  $k$ -additive form and the explicit one, which one is the **most succinct** ?



## Explicit vs. $k$ -additive Form

Among the  $k$ -additive form and the explicit one, which one is the **most succinct** ?

### Proposition

The explicit and the  $k$ -additive form of representing utility functions are incomparable with respect to succinctness.

- $u_1(\pi) = |\pi|$ : size  $|\mathcal{O}|$  in the  $k$ -additive form, but  $2^{|\mathcal{O}|} - 1$  in the explicit form.
- $u_2(\pi) = 1$  for  $|\pi| = 1$  and  $u_2(\pi) = 0$  otherwise:  $|\mathcal{O}|$  non-zero values in the explicit form, but  $2^{|\mathcal{O}|} - 1$  in the  $k$ -additive:
  - $\alpha^\pi = 1$  for  $|\pi| = 1$
  - $\alpha^\pi = -2$  for  $|\pi| = 2$
  - ...



## $k$ -additive form: complexity

- [DOMINANCE] is in P.
- [OPTIMISATION] is NP-hard, but in P if all the weights are positive.

[Chevaleyre et al., 2008]



## Generalized Additive Independence

$\mathcal{X}$  set of variables (binary or not),  $v$  instantiation.

### Definition [Fishburn, 1970, Bacchus and Grove, 1995]

Let  $\mathcal{X}_1, \dots, \mathcal{X}_k$  be a family of subsets of  $\mathcal{X}$  such that  $\bigcup_i \mathcal{X}_i = \mathcal{X}$ .

$u$  is GAI-decomposable with respect to  $\mathcal{X}_1, \dots, \mathcal{X}_k$  iff  $\exists u_1 \dots u_k$  utility functions  $\mathcal{X}_i \rightarrow \mathbb{R}$  such that  $u(v) = \sum_{i=1}^k u_i(v \downarrow \mathcal{X}_i)$

### Remarks:

- For binary variables, additivity = GAI-decomposability, with  $|\mathcal{X}_i| = 1, \forall i$ .
- For binary variables,  $k$ -additivity = GAI-decomposability, with  $|\mathcal{X}_i| \leq k, \forall i$ .



## GAI-decomposition: example

{**house, tractor, lawn mower, fields**}

<i>F</i>	0	1
$u_1$	0	10

<i>H</i>	0	1
$u_2$	0	15

<i>TR</i>	0	1
$u_3$	0	2

<i>LM</i>	0	1
$u_4$	0	1

<i>F</i>	0	0	1	1
<i>TR</i>	0	1	0	1
$u_5$	0	0	0	8

<i>TR</i>	0	0	1	1
<i>LM</i>	0	1	0	1
$u_6$	0	0	0	-8

<i>H</i>	0	0	1	1
<i>TR</i>	0	1	0	1
$u_7$	0	0	0	7

<i>H</i>	0	0	1	1
<i>LM</i>	0	1	0	1
$u_8$	0	0	0	10



## GAI-nets

- [Gonzales and Perny, 2004]
- A graphical language to represent GAI-decomposable utility functions
- Similar to **junction graphs** in Bayesian networks [Jensen et al., 1994]

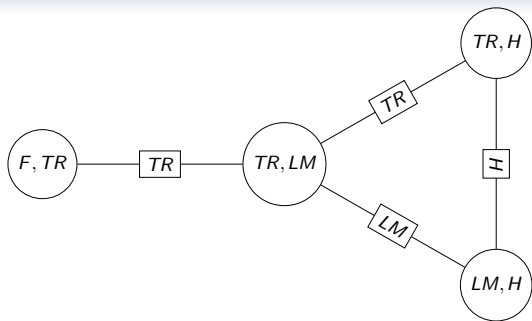
### GAI-net

Let  $u$  be a utility function decomposable over  $\mathcal{X}_1, \dots, \mathcal{X}_m$ . A GAI-net representing  $u$  is an undirected graph  $(G, E)$ , where:

- $V = \{X_{\mathcal{X}_1}, \dots, X_{\mathcal{X}_p}\}$  ;
- $\forall (X_{\mathcal{X}_i}, X_{\mathcal{X}_j}) \in E, \mathcal{X}_i \cap \mathcal{X}_j \neq \emptyset$  ;
- $\forall \mathcal{X}_i \cap \mathcal{X}_j = T_{ij} \neq \emptyset$ , there is a path from  $X_{\mathcal{X}_i}$  to  $X_{\mathcal{X}_j}$  whose nodes  $X_{\mathcal{X}_k}$  are all such that  $T_{ij} \subset \mathcal{X}_k$  (**running intersection property**).

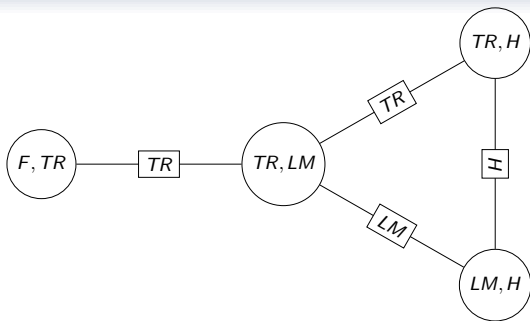


## GAI-nets





## GAI-nets



- GAI-nets are fully expressive (provided that we allow decomposition subsets of any size)

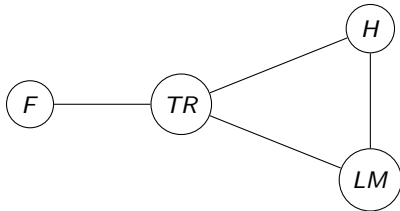
**DOMINANCE** is in P

**OPTIMISATION** is NP-hard in the general case



## Dependency graph

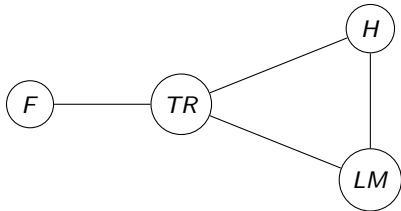
Another possible representation. . .





## Dependency graph

Another possible representation. . .



- Similarity with constraint graph in **constraint networks**
- Can be used to transform the GAI-net into a GAI-tree [Gonzales and Perny, 2004]:
  - Build the dependency graph
  - Triangulate the graph [Robertson and Seymour, 1986]
  - Construct the new GAI-tree
- Having a GAI-tree can ease **elicitation**, **reasoning** and **preference aggregation** [Gonzales et al., 2006]
- Link with **tree-decomposition-based** algorithms in constraint networks [Jégou and Terrioux, 2003]



## Outline

### 1 Languages for compact preference representation

### 2 Logical and bidding languages

Propositional logic

Bidding languages

### 3 Graphical languages for ordinal preferences

An introduction to graphical languages

Preferential independence

CP-nets

Outcome optimisation with CP-nets

CP-nets: extensions and variants

### 4 Graphical languages for cardinal preferences

Additivity generalized

Some other languages

### 5 Collective decision making

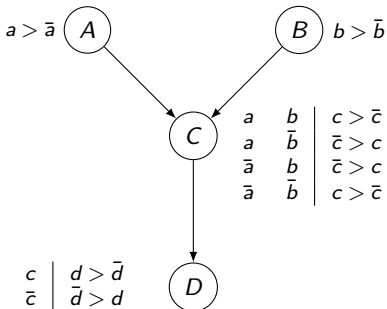
Voting

MultiAgent Resource Allocation

### 6 Conclusion

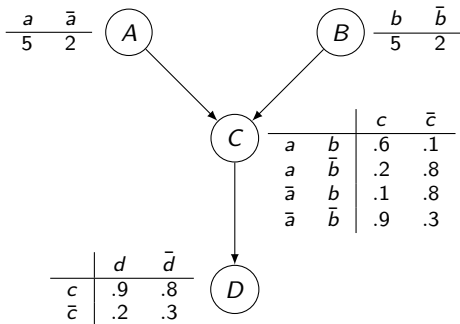


## From CP-nets to UCP-nets



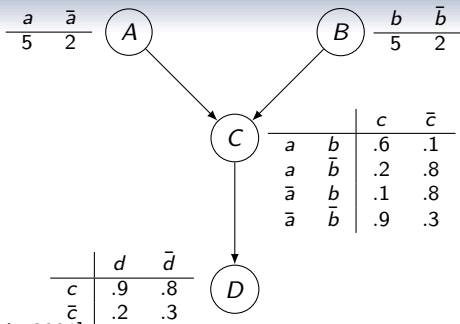


## From CP-nets to UCP-nets





## From CP-nets to UCP-nets



- [Boutilier et al., 2001]
- Combines generalized additivity and CP-nets
- UCP-net = GAI decomposition over clusters  $X \cup Pa(X)$  and a DAG  $G$  which is a valid CP-net.

OPTIMISATION : easy (forward-sweep procedure)

DOMINANCE : easy (GAI computation)



## Expected utility networks

### Expected Utility networks [La Mura and Shoham, 1999]

**Idea:** Representing probabilistic and utility conditional independence in the same way.

Representation based on:

- a **reference point**  $(x_1^0, \dots, x_n^0)$

- a probability ratio function  $q(x_{\mathcal{M}}|x_{\mathcal{K}}) = \frac{p(x_{\mathcal{M}}, x_{\mathcal{K}}, x_{\mathcal{V} \setminus \mathcal{M} \cup \mathcal{K}}^0)}{p(x_{\mathcal{M}}^0, x_{\mathcal{K}}, x_{\mathcal{V} \setminus \mathcal{M} \cup \mathcal{K}}^0)}$

- a utility ratio function  $w(x_{\mathcal{M}}|x_{\mathcal{K}}) = \frac{u(x_{\mathcal{M}}, x_{\mathcal{K}}, x_{\mathcal{V} \setminus \mathcal{M} \cup \mathcal{K}}^0)}{u(x_{\mathcal{M}}^0, x_{\mathcal{K}}, x_{\mathcal{V} \setminus \mathcal{M} \cup \mathcal{K}}^0)}$



## Utility Difference networks

### Utility Difference networks [Brafman and Engel, 2009]

- **Idea:** use a notion of utility independence close to probabilistic independence, as in EUN
- but... taking the "log"  $\rightarrow$  the ratio becomes a **difference**
- all the operations are isomorphic to those in the Bayesian networks
- We also have a **reference point**  $\rightarrow$  reference utility function  $u_r(\mathcal{K}) = u(\mathcal{K}x_{\mathcal{V} \setminus \mathcal{K}}^0)$
- **Conditional utility function:**  $u_r(\mathcal{K}|\mathcal{L}) = u(\mathcal{K}\mathcal{L}) - u(v_{\mathcal{K}}^0\mathcal{L})$
- **Conditional independence:**  $CDI_r(\mathcal{K}, \mathcal{L}|\mathcal{M})$  iff for all  $v \in D_{\mathcal{M}}$ ,  
 $u_r(\mathcal{K}|\mathcal{L}v) = u_r(\mathcal{K}|v)$



## Conditional Utility Independence networks

### Conditional Utility Independence networks [Engel, 2008]

- Based on a weaker notion of independence than CAI (on which are based GAI-nets)
- $CAI(\mathcal{L}, \mathcal{K} | \mathcal{M})$  (with  $\mathcal{M} = \mathcal{V} \setminus (\mathcal{K} \cup \mathcal{L})$ ):  $u(\mathcal{K}, \mathcal{L}, \mathcal{M}) = f(\mathcal{K}, \mathcal{L}) + g(\mathcal{L}, \mathcal{M})$  (**undirectional**)
- $CUI(\mathcal{L}, \mathcal{K} | \mathcal{M})$  (with  $\mathcal{M} = \mathcal{V} \setminus (\mathcal{K} \cup \mathcal{L})$ ):  
 $u(\mathcal{K}, \mathcal{L}, \mathcal{M}) = f(\mathcal{K}, \mathcal{L}) + h(\mathcal{K}, \mathcal{M}).g(\mathcal{L}, \mathcal{M})$  (**directional**)
- **Graphical model:** DAG where  $CUI(\mathcal{V} \setminus (\{X\} \cup Pa(X)), X | Pa(X))$



## PFU framework

### Plausibility Feasibility Utility framework [Pralet et al., 2007]

- **Idea:** Mixing constraints, uncertainties, and preferences in an homogeneous framework
- Several different queries can be directly expressed in the language
- Generic algorithms have been developed to answer to these queries



## Outline

### 1 Languages for compact preference representation

### 2 Logical and bidding languages

Propositional logic

Bidding languages

### 3 Graphical languages for ordinal preferences

An introduction to graphical languages

Preferential independence

CP-nets

Outcome optimisation with CP-nets

CP-nets: extensions and variants

### 4 Graphical languages for cardinal preferences

Additivity generalized

Some other languages

### 5 Collective decision making

Voting

MultiAgent Resource Allocation

### 6 Conclusion



## Voting

- $\mathcal{A} = \{1, \dots, n\}$  finite set of **voters** ;
- $\mathcal{X}$  finite set of **candidates** (= alternatives);
- **profile:**

$$P = (\gamma_1, \dots, \gamma_n)$$

$\gamma_i$  = linear order on  $\mathcal{X}$  (vote) expressed by voter  $i$ .

A **voting rule** maps every profile to a candidate.



## Voting

An important problem in social choice: **voting in a combinatorial domain**.

Examples:

**Example 1** choosing a common menu:

$$\begin{aligned}\mathcal{X} = & \quad \{\text{soup, salad}\} \\ & \times \quad \{\text{fish, meat}\} \\ & \times \quad \{\text{apple pie, ice cream}\}\end{aligned}$$

**Example 2** multiple referendum: a local community has to decide on several interrelated issues (should we build a swimming pool or not? should we build a tennis court or not?)

**Example 3** recruiting committee (3 positions, 6 applicants):

$$\mathcal{X} = \{A \mid A \subseteq \{a, b, c, d, e, f\}, |A| \leq 3\}.$$

**Combinatorial domains:**  $\mathcal{V} = \{X_1, \dots, X_p\}$  set of **variables**, or **issues**;  
 $\mathcal{X} = D_1 \times \dots \times D_p$  (where  $D_i$  is a finite value domain for variable  $X_i$ )

**Voting in a combinatorial domain:** choose collectively a value for each variable.



## Voting in combinatorial domains

A naive solution: **don't bother and vote separately on each variable.**  
⇒ **multiple election paradoxes**

### Example

2 binary variables  $S$  (build a new swimming pool),  $T$  (build a new tennis court)

voters 1 and 2     $S\bar{T} \succ \bar{S}T \succ \bar{S}\bar{T} \succ ST$

voters 3 and 4     $\bar{S}T \succ S\bar{T} \succ \bar{S}\bar{T} \succ ST$

voter 5             $ST \succ S\bar{T} \succ \bar{S}T \succ \bar{S}\bar{T}$

**Problem 1:** voters 1-4 feel ill at ease reporting a preference on  $\{S, \bar{S}\}$  and  $\{T, \bar{T}\}$



## Voting in combinatorial domains

A naive solution: **don't bother and vote separately on each variable.**  
⇒ **multiple election paradoxes**

### Example

2 binary variables  $S$  (build a new swimming pool),  $T$  (build a new tennis court)

voters 1 and 2     $S\bar{T} \succ \bar{S}T \succ \bar{S}\bar{T} \succ ST$

voters 3 and 4     $\bar{S}T \succ S\bar{T} \succ \bar{S}\bar{T} \succ ST$

voter 5             $ST \succ S\bar{T} \succ \bar{S}T \succ \bar{S}\bar{T}$

**Problem 1:** voters 1-4 feel ill at ease reporting a preference on  $\{S, \bar{S}\}$  and  $\{T, \bar{T}\}$

**Problem 2:** suppose they do so by an “optimistic” projection

- voters 1, 2 and 5:  $S$ ; voters 3 and 4:  $\bar{S} \Rightarrow$  decision =  $S$ ;
- voters 3,4 and 5:  $T$ ; voters 1 and 2:  $\bar{T} \Rightarrow$  decision =  $T$ .

Alternative  $ST$  is chosen although it is the worst alternative for all but one voter.



## Voting and graphical languages

The input of a voting problem is a collection of preference relations

⇒ CP-nets graphical languages for ordinal preferences are relevant for voting in combinatorial domains

Two ways of using graphical languages in this context.

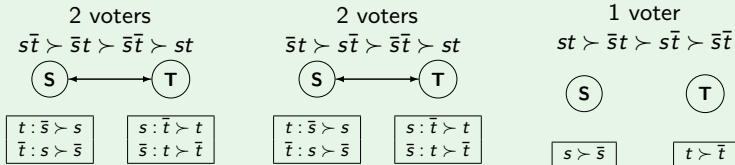
- 1 **elicit preferences globally, and then aggregate them:**
- 2 **proceed sequentially, interleaving elicitation and aggregation:**



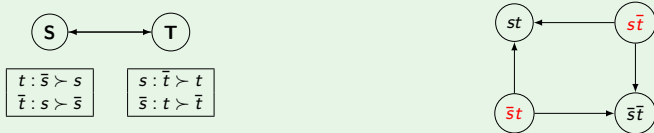
# Voting and CP-nets: aggregating CP-nets

## Example

Swimming pool and tennis: 5 voters, 2 binary variables  $S, T$



Aggregate locally (by majority) for each pair of adjacent outcomes:

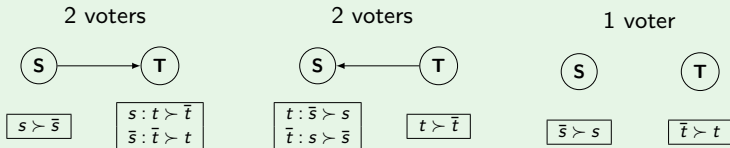




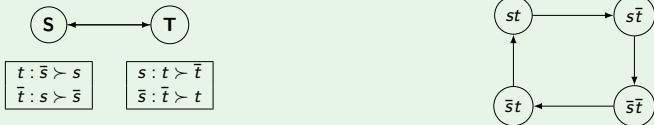
## Voting and CP-nets: aggregating CP-nets

### Example

Another example: again 5 voters and 2 binary variables  $S, T$



Aggregate locally (by majority) for each pair of adjacent outcomes:





## Voting and CP-nets: aggregating CP-nets

- + always applicable, because any preference relation is compatible with some CP-net (possibly with cyclic dependencies).
- elicitation cost: in the worst case, exponential number of queries to each voter
- computation cost: dominance in CP-nets with cyclic dependencies is PSPACE-complete
- there might be no winner; there might be several winners

[Xia et al., 2008, Conitzer et al., 2011, Li et al., 2011]



## Voting and CP-nets: sequential voting

**Assumption:** there exists an order on variables, say  $X_1 > \dots > X_p$ , such that for every voter and for every  $i$ ,  $X_i$  is preferentially independent of  $X_{i+1}, \dots, X_p$  given  $X_1, \dots, X_{i-1}$ .

Equivalently: if the dependencies graphs of the agents are  $G_1, \dots, G_n$  then  $G_1 \cup \dots \cup G_n$  is acyclic.

**Sequential voting:** apply local voting rules, one variable after the other, in an order compatible with  $G$ .

At every step:

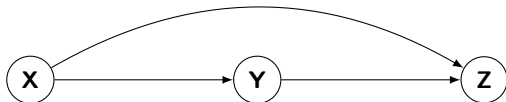
- we elicit the voters' preferences about a single variable;
- a local rule is used to compute the value chosen for this variable;
- this value is communicated to the voters.

We don't need to know the whole preference relations of the voters but only a part of their CP-nets.

[Lang and Xia, 2009]



## Voting and CP-nets: sequential voting



- 1 elicit voters' preferences on  $\mathbf{X}$  (possible because their preferences on  $\mathbf{X}$  are unconditional);
- 2 apply local voting rule  $r_X$  and determine the "local" winner  $x^*$ ;
- 3 elicit voters' preferences on  $\mathbf{Y}$  given  $\mathbf{X} = x^*$  (possible because their preferences on  $\mathbf{Y}$  depend only on  $\mathbf{X}$ );
- 4 apply local voting rule  $r_Y$  and determine  $y^*$ ;
- 5 elicit voters' preferences on  $\mathbf{Z}$  given  $\mathbf{X} = x^*$  and  $\mathbf{Y} = y^*$ .
- 6 apply local voting rule  $r_Z$  and determine  $z^*$ .
- 7 winner:  $(x^*, y^*, z^*)$



## Voting and CP-nets: sequential voting

Example:  $r_X = r_Y =$  majority rule

3 voters

$\bar{x} \succ x$
$x : \bar{y} \succ y$
$\bar{x} : y \succ \bar{y}$

2 voters

$x \succ \bar{x}$
$x : y \succ \bar{y}$
$\bar{x} : \bar{y} \succ y$

2 voters

$x \succ \bar{x}$
$\bar{y} \succ y$

order:  $X > Y$ .

4 voters out of 7 prefer  $x$  to  $\bar{x} \Rightarrow x^* = r_X(\succ_1, \dots, \succ_7) = x$



## Voting and CP-nets: sequential voting

Example:  $r_X = r_Y =$  majority rule

3 voters	2 voters	2 voters
$\bar{x} \succ x$	$x \succ \bar{x}$	$x \succ \bar{x}$
$x : \bar{y} \succ y$	$x : y \succ \bar{y}$	$\bar{y} \succ y$
$\bar{x} : y \succ \bar{y}$	$\bar{x} : \bar{y} \succ y$	

order:  $X > Y$ .

4 voters out of 7 prefer  $x$  to  $\bar{x} \Rightarrow x^* = r_X(\succ_1, \dots, \succ_7) = x$

given  $X = x$ , 5 voters out of 7 prefer  $\bar{y}$  to  $y \Rightarrow y^* = \bar{y}$

$$\text{Seq}(r_X, r_Y)(P) = x\bar{y}$$



## Outline

- 1 Languages for compact preference representation**
- 2 Logical and bidding languages**
  - Propositional logic
  - Bidding languages
- 3 Graphical languages for ordinal preferences**
  - An introduction to graphical languages
  - Preferential independence
  - CP-nets
  - Outcome optimisation with CP-nets
  - CP-nets: extensions and variants
- 4 Graphical languages for cardinal preferences**
  - Additivity generalized
  - Some other languages
- 5 Collective decision making**
  - Voting
  - MultiAgent Resource Allocation
- 6 Conclusion**

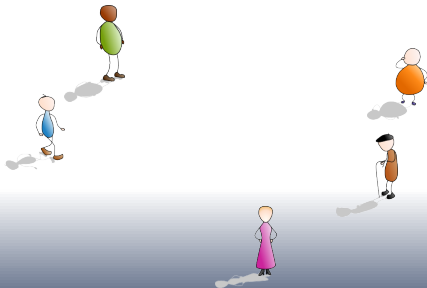


## The resource allocation problem...



## The resource allocation problem...

- A finite set  $\mathcal{N}$  of agents .





## The resource allocation problem...

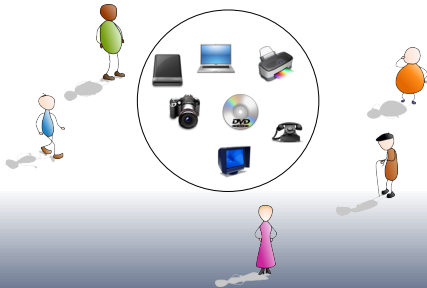
- A finite set  $\mathcal{N}$  of **agents** .
- A limited common **resource**.





## The resource allocation problem...

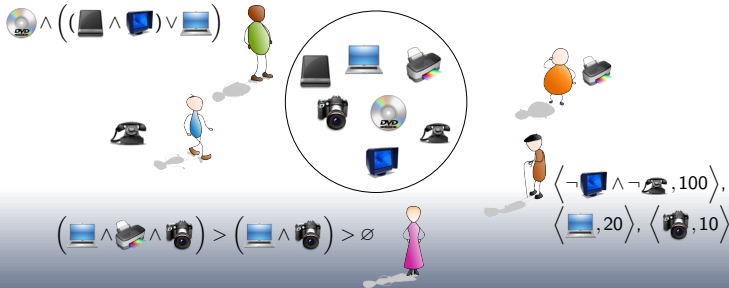
- A finite set  $\mathcal{N}$  of **agents** .
- A limited common **resource**.





## The resource allocation problem...

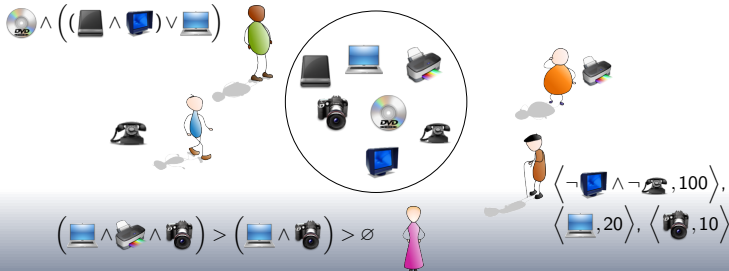
- A finite set  $\mathcal{N}$  of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.





## The resource allocation problem...

- A finite set  $\mathcal{N}$  of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral, ...).



A bundle cannot exceed the transport capacity of an agent.



## The resource allocation problem...

- A finite set  $\mathcal{N}$  of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral, ...).
- An **optimisation** or **decision** criterion.



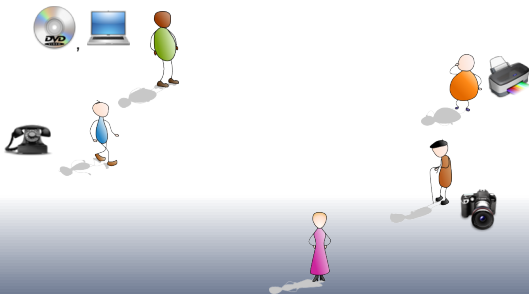
A bundle cannot exceed the transport capacity of an agent.



## The resource allocation problem...

- A finite set  $\mathcal{N}$  of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral, ...).
- An **optimisation** or **decision** criterion.

*How to allocate a part of or the whole resource to each agent such that no constraint is violated, and the criterion is optimized or verified.*

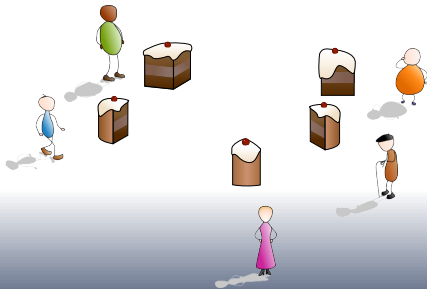




## The resource allocation problem...

- A finite set  $\mathcal{N}$  of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral, ...).
- An **optimisation** or **decision** criterion.

*How to allocate a part of or the whole resource to each agent such that no constraint is violated, and the criterion is optimized or verified.*





## Real-world applications

### An ubiquitous problem. . .

- Fair share of Earth Observation Satellites [Lemaître et al., 1999]
- Tasks or subjects allocation.
- Combinatorial auctions problems [Cramton et al., 2006].
- Allocation of waste water treatment rights [Murillo Espinar, 2010].
- Computer network sharing, rostering problems, allocation of take-off and landing slots in airports [Faltings, 2005],. . . .



## A resource to share

- Resource may be **continuous** (e.g. energy) or **discrete** (e.g. candy)
- It can be **sharable / non-rival** (e.g. information, numerical picture) or **non sharable / rival** (e.g. candy, physical object)
- Other dichotomies: **static** vs. **time-dependent**, **single-unit** vs. **multi-unit**, **tasks** vs. **goods**



## A resource to share

- Resource may be **continuous** (e.g. energy) or **discrete** (e.g. candy)
- It can be **sharable / non-rival** (e.g. information, numerical picture) or **non sharable / rival** (e.g. candy, physical object)
- Other dichotomies: **static** vs. **time-dependent**, **single-unit** vs. **multi-unit**, **tasks** vs. **goods**

In the following, we focus on **discrete, static, single-unit** resource.

- Elements of  $\mathcal{O}$  are called (indivisible) **items, goods** or **objects**.
- A **bundle** is an element  $\pi \in 2^{\mathcal{O}}$ .
- an **allocation** is a vector  $\vec{\pi} = \langle \pi_1, \dots, \pi_n \rangle \in 2^{\mathcal{O}^n}$
- $\pi_i \cap \pi_j = \emptyset$  if the goods are non shareable.



## A resource to share

- Resource may be **continuous** (e.g. energy) or **discrete** (e.g. candy)
- It can be **sharable** / **non-rival** (e.g. information, numerical picture) or **non sharable** / **rival** (e.g. candy, physical object)
- Other dichotomies: **static** vs. **time-dependent**, **single-unit** vs. **multi-unit**, **tasks** vs. **goods**

In the following, we focus on **discrete, static, single-unit** resource.

- Elements of  $\mathcal{O}$  are called (indivisible) **items, goods** or **objects**.
- A **bundle** is an element  $\pi \in 2^{\mathcal{O}}$ .
- an **allocation** is a vector  $\vec{\pi} = \langle \pi_1, \dots, \pi_n \rangle \in 2^{\mathcal{O}^n}$
- $\pi_i \cap \pi_j = \emptyset$  if the goods are non shareable.

Preferences are assumed to be:

- **non-exogenous**: the set of alternatives is  $2^{\mathcal{O}}$ ;
- **monotonic**:  $\pi \subseteq \pi' \Rightarrow \pi \preceq \pi'$



## Auctions

Auctions are a special case of multiagent resource allocation problem.

- Numerical preferences: **money**.
- **Usual criterion**: maximize the revenue of the auctioneer.
- **Combinatorial auctions** [Cramton et al., 2006]
  - preferential dependencies ( $\rightarrow$  bidding languages)
  - *allocate the objects*  $\rightarrow$  **Winner Determination Problem**
  - First reference [Rassenti et al., 1982]
  - Extensively studied in the last 15 years



## The WDP

### Winner Determination Problem

- **Input** : A set of bids (XOR, OR, or other...)
- **Solution** : An allocation that maximizes the sum of utilities (*i.e* the auctioneer's payment).

### Example

- **Agent 1** : ( $\{\text{DVD}, \text{Laptop}, \text{Printer}\}, 400\text{€}$ )
- **Agent 2** : ( $\{\text{Camera}, \text{Printer}\}, 700\text{€}$ ) *OR* ( $\{\text{Phone}\}, 100\text{€}$ )



## The WDP

### Winner Determination Problem

- **Input** : A set of bids (XOR, OR, or other...)
- **Solution** : An allocation that maximizes the sum of utilities (*i.e* the auctioneer's payment).

### Example

- **Agent 1** : ( $\{\text{DVD}, \text{Laptop}, \text{Printer}\}, 400\text{€}$ )
- **Agent 2** : ( $\{\text{Camera}, \text{Printer}\}, 700\text{€}$ ) OR ( $\{\text{Phone}\}, 100\text{€}$ )

Solution :

- **Agent 1** : Nothing
- **Agent 2** :  $\{\text{Camera}, \text{Printer}, \text{Phone}\} \rightarrow u = 800 \text{€}$ .



## The WDP: solving

- NP-complete problem (for most languages)
- Efficient solving:

- linear formulation:  $\mathbf{x}_i$  0-1 variables, bids  $(\pi_i, w_i)$

$$\begin{aligned} & \text{maximize } \sum_i w_i \times \mathbf{x}_i \\ & \forall i \neq j \text{ s.t. } \pi_i \cap \pi_j \neq \emptyset, \mathbf{x}_i + \mathbf{x}_j \leq 1 \end{aligned}$$

- branching algorithms (objects / bids),
- approximated algorithms...



## Auctions and fairness

### Example

- **Agent 1** : ( $\{\text{DVD}, \text{Laptop}, \text{Printer}\}$ , 400€)
- **Agent 2** : ( $\{\text{Camera}, \text{Printer}\}$ , 700€) *OR* ( $\{\text{Phone}\}$ , 100€)

Solution :

- **Agent 1** : Nothing
- **Agent 2** :  $\{\text{Camera}, \text{Printer}, \text{Phone}\} \rightarrow u = 800 \text{ €}$ .



## Auctions and fairness

### Example

- Agent 1 : ( $\{\text{DVD}, \text{hard drive}, \text{printer}\}, 400\text{€}$ )
- Agent 2 : ( $\{\text{camera}, \text{printer}\}, 700\text{€}$ ) OR ( $\{\text{phone}\}, 100\text{€}$ )

Solution (fairer) :

- Agent 1 :  $\{\text{DVD}, \text{hard drive}, \text{printer}\} \rightarrow u = 400 \text{ €}$ .
- Agent 2 :  $\{\text{phone}\} \rightarrow u = 100 \text{ €}$ .



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- Egalitarian ordering
- Leximin egalitarian ordering
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages, . . .



## Usual Social Welfare Orderings

- **Classical utilitarian ordering**
- Egalitarian ordering
- Leximin egalitarian ordering
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...

### Classical utilitarianism [Harsanyi]

$$\vec{u} \preceq \vec{v} \Leftrightarrow \sum_{i=1}^n u_i \leq \sum_{i=1}^n v_i.$$

### Features

Conveys the sum-fitness principle (resource goes to who makes the best use of it).

Indifferent to inequalities (Pigou-Dalton)  $\rightsquigarrow$  can lead to huge inequalities between the agents.



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- **Egalitarian ordering**
- Leximin egalitarian ordering
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...

### **Egalitarianism [Rawls, 1971]**

$$\vec{u} \preceq \vec{v} \Leftrightarrow \min_{i=1}^n u_i \leq \min_{i=1}^n v_i.$$

### **Features**

Conveys the compensation principle: the least well-off must be made as well-off as possible (justice according to needs)  $\rightsquigarrow$  tends to equalize the utility profile.



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- **Egalitarian ordering**
- Leximin egalitarian ordering
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...

### **Egalitarianism [Rawls, 1971]**

$$\vec{u} \preceq \vec{v} \Leftrightarrow \min_{i=1}^n u_i \leq \min_{i=1}^n v_i.$$

### **Features**

Conveys the compensation principle: the least well-off must be made as well-off as possible (justice according to needs)  $\rightsquigarrow$  tends to equalize the utility profile.

**However, it can lead to non Pareto-efficient decisions (drowning effect).**



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- **Egalitarian ordering**
- Leximin egalitarian ordering
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...

### Egalitarianism [Rawls, 1971]

$$\vec{u} \preceq \vec{v} \Leftrightarrow \min_{i=1}^n u_i \leq \min_{i=1}^n v_i.$$

### Egalitarian SWO and Pareto-efficiency

$\langle 1, 1, 1, 1 \rangle \sim \langle 1000, 1, 1000, 1000 \rangle$ , whereas  $\langle 1, 1, 1, 1 \rangle$  and  $\langle 1000, 1, 1000, 1000 \rangle$  are very different!



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- Egalitarian ordering
- **Leximin egalitarian ordering**
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...

### Leximin egalitarianism [Sen, 1970, Kolm, 1972]

Let  $\vec{x}$  be a vector. We write  $\vec{x}^\uparrow$  the sorted version of  $\vec{x}$ .  
 $\vec{u} \succ_{\text{leximin}} \vec{v} \Leftrightarrow \exists k$  such that  $\forall i \leq k, u_i^\uparrow = v_i^\uparrow$  and  $u_{k+1}^\uparrow > v_{k+1}^\uparrow$ .

**This is a lexicographical comparison over sorted vectors.**

### Perform a leximin comparison...

Two vectors to compare:  $\vec{u} = \langle 4, 10, 3, 5 \rangle$  and  $\vec{v} = \langle 4, 3, 6, 6 \rangle$ .

- We sort the two vectors: 
$$\begin{cases} \vec{u}^\uparrow = \langle 3, 4, 5, 10 \rangle \\ \vec{v}^\uparrow = \langle 3, 4, 6, 6 \rangle \end{cases}$$
- We lexicographically sort the ordered vectors:  $\vec{u}^\uparrow \prec_{\text{lexico}} \vec{v}^\uparrow$



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- Egalitarian ordering
- **Leximin egalitarian ordering**
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...

### **Leximin egalitarianism [Sen, 1970, Kolm, 1972]**

Let  $\vec{x}$  be a vector. We write  $\vec{x}^\uparrow$  the sorted version of  $\vec{x}$ .  
 $\vec{u} \succ_{leximin} \vec{v} \Leftrightarrow \exists k$  such that  $\forall i \leq k, u_i^\uparrow = v_i^\uparrow$  and  $u_{k+1}^\uparrow > v_{k+1}^\uparrow$ .

**This is a lexicographical comparison over sorted vectors.**

### **Features**

This SWO both refines the egalitarian SWO and the Pareto relation  $\rightsquigarrow$  it inherits of the fairness features of egalitarianism, while overcoming drowning effect.



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- Egalitarian ordering
- **Leximin egalitarian ordering**
- Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...

### Leximin egalitarianism [Sen, 1970, Kolm, 1972]

Let  $\vec{x}$  be a vector. We write  $\vec{x}^\uparrow$  the sorted version of  $\vec{x}$ .  
 $\vec{u} \succ_{leximin} \vec{v} \Leftrightarrow \exists k$  such that  $\forall i \leq k, u_i^\uparrow = v_i^\uparrow$  and  $u_{k+1}^\uparrow > v_{k+1}^\uparrow$ .

**This is a lexicographical comparison over sorted vectors.**

### Leximin SWO leximin and Pareto-efficiency

$\langle 1, 1, 1, 1 \rangle \prec \langle 1000, 1, 1000, 1000 \rangle$  (the second value of the two vectors is discriminating).



## Usual Social Welfare Orderings

- Classical utilitarian ordering
- Egalitarian ordering
- Leximin egalitarian ordering
- **Compromises between classical utilitarianism and egalitarianism: Nash ( $\times$ ), families OWA and generalized averages,...**



## Allocation and numerical preferences

Resource allocation with numerical preferences is not so straightforward. . .



## Allocation and numerical preferences

Resource allocation with numerical preferences is not so straightforward. . .

### **Complexity:**

With most logical based languages, generalized additivity, OR or XOR bids, with most social welfare orderings. . .

. . . **NP**-hard problem, except for very simple cases [Bouveret et al., 2005].



## Allocation and numerical preferences

Resource allocation with numerical preferences is not so straightforward. . .

### Complexity:

With most logical based languages, generalized additivity, OR or XOR bids, with most social welfare orderings. . .

. . . **NP**-hard problem, except for very simple cases [Bouveret et al., 2005].

Some works that investigate algorithmical issues:

[Bouveret and Lemaître, 2009, Gonzales et al., 2006, Lesca and Perny, 2010]. . .



## Allocation and numerical preferences

Resource allocation with numerical preferences is not so straightforward. . .

### Complexity:

With most logical based languages, generalized additivity, OR or XOR bids, with most social welfare orderings. . .

. . . **NP**-hard problem, except for very simple cases [Bouveret et al., 2005].

Some works that investigate algorithmical issues:

[Bouveret and Lemaître, 2009, Gonzales et al., 2006, Lesca and Perny, 2010]. . .

**Interpersonal comparison of utilities:** *is a utility of 10 worth the same for agent 1 and 4 ?*

Possible solutions:

- Use a common utility scale (e.g money, like in auctions)
- Use the Nash social welfare ordering (independent of individual utility scales)
- Normalize utilities (e.g the Kalai-Smorodinski solution)



## With ordinal preferences

**Ordinal preferences:** What can we do ?



## With ordinal preferences

### Ordinal preferences: What can we do ?

- *scoring* procedures + additional assumptions for comparing sets of objects [Brams and King, 2005]

$o_4$	$\succ$	$o_2$	$\succ$	$o_3$	$\succ$	$o_1$
$\downarrow$		$\downarrow$		$\downarrow$		$\downarrow$
16		8		4		2



## With ordinal preferences

### Ordinal preferences: What can we do ?

- *scoring* procedures + additional assumptions for comparing sets of objects [Brams and King, 2005]
- Pareto-efficiency

**Agent 1 :**  $o_1 \succ o_2 \succ o_1 o_2 \succ \emptyset$  / **Agent 2 :**  $o_2 \succ o_2 o_1 \succ \emptyset \succ o_1$

$(1 : o_2, 2 : o_1)$  Pareto-dominates  $(1 : o_1, 2 : o_2)$



## With ordinal preferences

### Ordinal preferences: What can we do ?

- *scoring* procedures + additional assumptions for comparing sets of objects [Brams and King, 2005]
- Pareto-efficiency
- Envy-freeness

**Agent 1 :**  $o_1 \succ o_2 \succ o_1 o_2 \succ \emptyset$  / **Agent 2 :**  $o_2 \succ o_2 o_1 \succ \emptyset \succ o_1$

$(1 : o_1, 2 : \emptyset)$  is envy-free whereas  $(1 : o_1, 2 : o_2)$  is not.



## Ordinal criteria: complexity

- **Pareto-efficiency** : in general easy! (give all the objects to one agent)



## Ordinal criteria: complexity

- **Pareto-efficiency** : in general easy! (give all the objects to one agent)
- **Envy-freeness** :
  - **Alone** : easy, but not interesting!



## Ordinal criteria: complexity

- **Pareto-efficiency** : in general easy! (give all the objects to one agent)
- **Envy-freeness** :
  - **Alone** : easy, but not interesting!
  - **Complete allocation** : in general **NP**-complete [Lipton et al., 2004].



## Ordinal criteria: complexity

- **Pareto-efficiency** : in general easy! (give all the objects to one agent)
- **Envy-freeness** :
  - **Alone** : easy, but not interesting!
  - **Complete allocation** : in general **NP**-complete [Lipton et al., 2004].
  - **Pareto-efficiency** : in general  $\Sigma_p^2$ -complete: dichotomous preferences, weighted goal bases [Bouveret and Lang, 2008], (generalized) additives [de Keijzer et al., 2009], etc.



## Incomplete preferences

**Incomplete ordinal preferences:** (like CP-nets) What can we do ?

→ **necessary** or **possible** envy-freeness (resp. Pareto-efficiency)

- **Objects:**  $\mathcal{O} = \{a, b, c\}$ .
- **Agents:**
  - **Agent 1 :**  $bc \succ ab, a \succ c$ , + monotonicity (ex :  $abc \succ ab$ ) + transitivity
  - **Agent 2 :**  $c \succ a, a \succ b$  + monotonicity + transitivity.



## Incomplete preferences

**Incomplete ordinal preferences:** (like CP-nets) What can we do ?

→ **necessary** or **possible** envy-freeness (resp. Pareto-efficiency)

- **Objects:**  $\mathcal{O} = \{a, b, c\}$ .
- **Agents:**
  - **Agent 1 :**  $bc \succ ab, a \succ c$ , + monotonicity (ex :  $abc \succ ab$ ) + transitivity
  - **Agent 2 :**  $c \succ a, a \succ b$  + monotonicity + transitivity.
- $\langle 1 : a, 2 : bc \rangle$  **necessarily** leads to **envy**
- $\langle 1 : b, 2 : ac \rangle$  is **possibly envy-free** but not **necessarily**.



## Resource allocation and CP-nets

CP-nets are not very well-suited for resource allocation problems. . .

→ Use CI-nets instead

However, most reasoning tasks in CI-nets are very hard. . .

→ Use a tractable fragment of CI-nets: SCI-nets [Bouveret et al., 2010]



## Outline

- 1 **Languages for compact preference representation**
- 2 **Logical and bidding languages**
  - Propositional logic
  - Bidding languages
- 3 **Graphical languages for ordinal preferences**
  - An introduction to graphical languages
  - Preferential independence
  - CP-nets
  - Outcome optimisation with CP-nets
  - CP-nets: extensions and variants
- 4 **Graphical languages for cardinal preferences**
  - Additivity generalized
  - Some other languages
- 5 **Collective decision making**
  - Voting
  - MultiAgent Resource Allocation
- 6 **Conclusion**



## Conclusion

- Graphical languages are a powerful way of expressing preferences in combinatorial domains.
- They are equipped with (sometimes) efficient algorithms for finding an optimal outcome.
- Many applications, such as
  - configuration problems (tour packages, web page configuration etc.)
  - planning
  - information retrieval
  - group decision making
  - distributed decision making / game theory



**Bacchus, F. and Grove, A. (1995).**

Graphical models for preference and utility.

In Besnard, P. and Hanks, S., editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 3–10, Montréal, Canada. Morgan Kaufmann.



**Bienvenu, M., Lang, J., and Wilson, N. (2010).**

From preference logics to preference languages, and back.

In *Proceedings of the Twelfth International Conference on Principles of Knowledge Representation and Reasoning (KR10)*.



**Boutilier, C., Bacchus, F., and Brafman, R. I. (2001).**

UCP-networks: A directed graphical representation of conditional utilities.

In Breese, J. S. and Koller, D., editors, *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 56–64, Washington, DC. Morgan Kaufmann.



**Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., and Poole, D. (2004a).**

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

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



**Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., and Poole, D. (2004b).**

Preference-based constrained optimization with cp-nets.

*Computational Intelligence*, 20(2):137–157.



**Boutilier, C., Brafman, R. I., Hoos, H. H., and Poole, D. (1999).**

Reasoning with conditional ceteris paribus preference statements.

In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-99)*, Stockholm, Sweden.



**Bouveret, S., Endriss, U., and Lang, J. (2009).**

Conditional importance networks: A graphical language for representing ordinal, monotonic preferences over sets of goods.

In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, pages 67–72, Pasadena, California.



**Bouveret, S., Endriss, U., and Lang, J. (2010).**

Fair division under ordinal preferences: Computing envy-free allocations of indivisible goods.

In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI'10)*, Lisbon, Portugal. IOS Press.



**Bouveret, S., Fargier, H., Lang, J., and Lemaître, M. (2005).**

Allocation of indivisible goods: a general model and some complexity results.

In Dignum, F., Dignum, V., Koenig, S., Kraus, S., Singh, M. P., and Wooldridge, M., editors, *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, Utrecht, The Netherlands. ACM.



**Bouveret, S. and Lang, J. (2008).**

Efficiency and envy-freeness in fair division of indivisible goods: Logical representation and complexity.  
*Journal of Artificial Intelligence Research (JAIR)*, 32:525–564.



**Bouveret, S. and Lemaître, M. (2009).**

Computing leximin-optimal solutions in constraint networks.  
*Artificial Intelligence*, 173(2):343–364.



**Brafman, R. and Engel, Y. (2009).**

Directional decomposition of multiattribute utility functions.  
In *Proceedings of the 1st International Conference on Algorithmic Decision Theory (ADT'09)*, Lecture Notes in Artificial Intelligence, Venice, Italy. Springer Verlag.



**Brafman, R. I. and Dimopoulos, Y. (2004).**

Extended semantics and optimization algorithms for cp-networks.  
*Computational Intelligence*, 20(2):218–245.



**Brafman, R. I. and Domshlak, C. (2007).**

Representing, eliciting, and reasoning with preferences.  
AAAI 2007 tutorial notes.



**Brafman, R. I., Domshlak, C., and Shimony, S. E. (2006).**

On graphical modeling of preference and importance.  
*J. Artif. Intell. Res. (JAIR)*, 25:389–424.



**Brams, S. J. and King, D. (2005).**

Efficient fair division—help the worst off or avoid envy?  
*Rationality and Society*, 17(4):387–421.



**Chen, L. and Pu, P. (2004).**

Survey of preference elicitation methods.  
Technical report, École Polytechnique Fédérale de Lausanne.



**Chevalyere, Y., Endriss, U., Estivie, S., and Maudet, N. (2008).**

Multiagent resource allocation in  $k$ -additive domains.  
*Annals of Operations Research*, 163(1):49–62.



**Chevalyere, Y., Endriss, U., and Lang, J. (2006).**

Expressive power of weighted propositional formulas for cardinal preference modelling.

In *Proc. of KR-06*.



**Conitzer, V., Lang, J., and Xia, L. (2011).**

Hypercube-wise preference aggregation in multi-issue domains.

In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*, Barcelona, Spain.



**Coste-Marquis, S., Lang, J., Liberatore, P., and Marquis, P. (2004).**

Expressive power and succinctness of propositional languages for preference representation.

In *Proc. of KR-04*.



**Cramton, P., Shoham, Y., and Steinberg, R., editors (2006).**

*Combinatorial Auctions*.

MIT Press.



**de Keijzer, B., Bouveret, S., Klos, T., and Zhang, Y. (2009).**

On the complexity of efficiency and envy-freeness in fair division of indivisible goods with additive preferences.

In *Proceedings of the 1st International Conference on Algorithmic Decision Theory (ADT'09)*, Lecture Notes in Artificial Intelligence, Venice, Italy. Springer Verlag.



**Domshlak, C., Prestwich, S., Rossi, F., and Venable, K. B. (2006).**

Hard and soft constraints for reasoning about qualitative conditional preferences.

*Journal of Heuristics*, 12:263–285.



**Engel, Y. (2008).**

Cui networks: a graphical representation for conditional utility independence.

*Journal of Artificial Intelligence Research*, 31:83–112.



**Faltings, B. (2005).**

A budget-balanced, incentive-compatible scheme for social choice.

In Faratin, P. and Rodriguez-Aguilar, J. A., editors, *Agent-Mediated Electronic Commerce VI*, volume 3435 of *LNAI*, pages 30–43. Springer.



**Fishburn, P. C. (1970).**

*Utility Theory for Decision-Making*.

John Wiley & Sons, New York.



**Gonzales, C. and Perny, P. (2004).**

Gai networks for utility elicitation.

In Dubois, D., Welty, C. A., and Williams, M.-A., editors, *Proceedings of the 9th International Conference on Principles of Knowledge Representation and Reasoning (KR-04)*, pages 224–234, Whistler, Canada. AAAI Press.



**Gonzales, C., Perny, P., and Queiroz, S. (2006).**

Preference aggregation in combinatorial domains using GAI-nets.

In Bouyssou, D., Roberts, F., and Tsoukiàs, A., editors, *Proceedings of the Dimacs-Lamsade workshop on voting theory and preference modelling*, number 6 in Annales du LAMSADE, pages 165–179, Paris, France. CNRS.



**Howard, R. A. and Matheson, J. E. (1984).**

Influence diagrams.

In Howard, R. A. and Matheson, J. E., editors, *Readings on the principles and applications of decision analysis*, volume 2, pages 720–761. Strategic decision group.



**Jégou, P. and Terrioux, C. (2003).**

Hybrid backtracking bounded by tree-decomposition of constraint networks.

*Artificial Intelligence*, 146(1):43–75.



**Jensen, F., Jensen, F., and Dittmer, S. (1994).**

From influence diagrams to junction trees.

In de Mántaras, R. L. and Poole, D., editors, *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 367–37, Seattle, WA. Morgan Kaufmann.



**Keeney, R. L. and Raiffa, H. (1976).**

*Decisions with Multiple Objectives: Preferences and Value Tradeoffs*.

John Wiley and Sons.



**Kolm, S.-C. (1972).**

*Justice et Équité*.

Cepremap, CNRS Paris.

(english translation: *Justice and Equity*, MIT Press 1998.



**La Mura, P. and Shoham, Y. (1999).**

Expected utility networks.

In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-99)*.



**Lang, J. (2004).**

Logical preference representation and combinatorial vote.

*Annals of Mathematics and Artificial Intelligence*, 42(1):37–71.



**Lang, J. and Xia, L. (2009).**

Sequential composition of voting rules in multi-issue domains.

*Mathematical Social Sciences*, 57(3):304–324.



**Lemaître, M., Verfaillie, G., and Bataille, N. (1999).**

Exploiting a common property resource under a fairness constraint: a case study.

In Dean, T., editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 206–211, Stockholm, Sweden. Morgan Kaufmann.



**Lesca, J. and Perny, P. (2010).**

Lp solvable models for multiagent fair allocation problems.

In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI'10)*, Lisbon, Portugal. IOS Press.



**Li, M., Vo, Q. B., and Kowalczyk, R. (2011).**

Majority-rule-based preference aggregation on multi-attribute domains with cp-nets.

In *Proceedings of AAMAS'11*.



**Lipton, R., Markakis, E., Mossel, E., and Saberi, A. (2004).**

On approximately fair allocations of divisible goods.

In *Proceedings of the 5th ACM Conference on Electronic Commerce (EC-04)*, New York, NY. ACM.



**Montanari, U. (1974).**

Network of constraints: Fundamental properties and applications to picture processing.

*Inf. Sci.*, 7:95–132.



**Murillo Espinar, J. (2010).**

*Egalitarian Behaviour in Multiunit Combinatorial Auctions*.

PhD thesis, Universitat de Girona.



**Pearl, J. (1988).**

*Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*.

Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.



**Pralet, C., Schiex, T., and Verfaillie, G. (2007).**

An algebraic graphical model for decision with uncertainties, feasibilities, and utilities.

*Journal of Artificial Intelligence Research*, 29:421–489.



**Prestwich, S., Rossi, F., Venable, K. B., and Walsh, T. (2004).**

Constrained cpnets.

In *Proceedings of CSCLP'04*.



**Rassenti, S., Smith, V. L., and Bulfin, R. L. (1982).**

A combinatorial auction mechanisms for airport time slot allocation.

*Bell Journal of Economics*, pages 402–417.



**Rawls, J. (1971).**

*A Theory of Justice*.

Harvard University Press, Cambridge, Mass.

Traduction française disponible aux éditions du Seuil.



**Robertson, N. and Seymour, P. D. (1986).**

Graph minors. ii. algorithmic aspects of tree-width.

*J. Algorithms*, 7(3):309–322.



**Schiex, T., Fargier, H., and Verfaillie, G. (1995).**

Valued constraint satisfaction problems: Hard and easy problems.

In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 631–637, Montréal, Canada. Morgan Kaufmann.



**Sen, A. K. (1970).**

*Collective Choice and Social Welfare*.

North-Holland.



**Wilson, N. (2004).**

Extending CP-nets with stronger conditional preference statements.

In *Proceedings of AAAI'04*.



**Wilson, N. (2009).**

An efficient deduction mechanism for expressive comparative preferences languages.

In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, Pasadena, California.



**Xia, L., Conitzer, V., and Lang, J. (2008).**

Voting on multiattribute domains with cyclic preferential dependencies.

In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI-08)*, pages 202–207.