Computational Social Choice: Autumn 2011

Ulle Endriss
Institute for Logic, Language and Computation
University of Amsterdam

Plan for Today

So far, we have (almost) always modelled *preferences* as linear orders over the set of alternatives. But:

- Other *preference structures* may be relevant as well: weak orders, partial orders, interval orders, utility functions, ...
- Particularly for large sets of alternatives, we need to clarify what language we want to use to actually *represent preferences*.

Today we will therefore focus on preferences themselves, rather than on their use within social choice theory. Topics to be covered:

- *ordinal* and *cardinal* preference structures
- the challenge of preference modelling in *combinatorial domains*
- several *compact preference representation languages*, namely: CP-nets, prioritised goals, weighted goals,
- research questions regarding preference representation languages, such as *expressivity* and *succinctness* (exemplified for weighted goals)

Ordinal and Cardinal Preferences

A *preference structure* represents an agent's preferences over a (finite) set of alternatives $\mathcal{X}$. Two types of preference structures:

- An *ordinal* preference structure is a *binary relation* $\succeq$ on $\mathcal{X}$.
  Read $x \succeq y$ as "x is at least as good as y". Define:
  - $x \succ y$ ("x is strictly better than y"): $x \succeq y$ but not $y \succeq x$
  - $x \sim y$ ("x is equally good as y"): both $x \succeq y$ and $y \succeq x$

  People often assume (at least) *transitivity* and *completeness* of $\succeq$.
- A *cardinal* preference structure is a (*utility* or *valuation*) function $u : \mathcal{X} \to Val$, where Val usually is a set of numerical values such as $\mathbb{R}$.

Every utility function u *induces* a preference relation $\succeq$; and every complete and transitive preference relation $\succeq$ is *representable* by a (in fact, more than one) utility function u: $x \succeq y$ iff $u(x) \geqslant u(y)$.

Most of voting theory and preference aggregation is based on ordinal preferences. Fair division (which we'll only see in the final lecture) mostly uses cardinal preferences. But there are exceptions (both ways).

Preorders

In economics (including social choice theory), transitivity is often taken to be a central requirement for preferences. Thus, ordinal preferences are usually modelled as (special kinds of) preorders on the set of alternatives $\mathcal{X}$.

A *preorder* on $\mathcal{X}$ is a binary relation $\succeq$ that is *reflexive* and *transitive*.
For any two alternatives $x, y \in \mathcal{X}$ exactly one of the following is true:

- $x \succ y$, i.e., $x \succeq y$ and not $y \succeq x$ ("I think x is better than y")
- $x \prec y$, i.e., $y \succeq x$ and not $x \succeq y$ ("I think x is worse than y")
- $x \sim y$, i.e., $x \succeq y$ and $y \succeq x$ ("I'm indifferent between x and y")
- $x \bowtie y$, i.e., neither $x \succeq y$ nor $y \succeq x$ ("I cannot compare x and y")

Some important classes of preorders on $\mathcal{X}$:

- A *partial order* $\succeq$ is a preorder that is *antisymmetric* (excludes $\sim$).
- A *weak order* $\succeq$ is a preorder that is *complete* (excludes $\bowtie$).
- A *total order* $\succeq$ is a preorder that is both *antisymmetric* and *complete*.
  The strict part $\succ$ of a total order is called a *linear order*.

In practice, the terms *total order* and *linear order* are used interchangeably.

Appropriateness of Representation

What preference structure is *appropriate* for a given application?

- Real-world/cognitive considerations: how close should the representation be to how humans express their preferences?
  - Can we assume that an agent can assign a real number reflecting her preference to each and every alternative?
  - Are preferences really complete (a common assumption)?
- Technical considerations:
  - Maybe it will be more convenient to work with utility functions?
  - Maybe we can prove more attractive results using them?

Two important features of preference that can be modelled using utility functions, but not ordinal preference structures:

- *Interpersonal comparison:* "Ann likes x more than Bob likes y."
- *Preference intensity:* "Ann's preference of x over y is stronger than her preference of y over z."

Transitivity of the Indifference Relation

Two observations regarding the *indifference relation* $\sim$ induced by $\succeq$:

- If $\succeq$ is a total order, then $\sim$ is the identity relation $=$.
- If $\succeq$ is a weak order, then $\sim$ is *transitive*.

But even if we accept that $\succeq$ and $\succ$ should be transitive, we may not always want $\sim$ to be transitive:

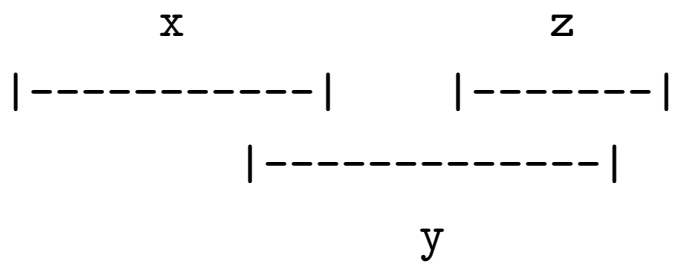
> For any $k \in \mathbb{N}$, I'm indifferent between a cup of coffee with k grains of sugar and a cup with $k+1$ grains of sugar. But that does not mean that I'm indifferent between a cup of coffee without sugar and one with 1,000,000 grains of sugar.

How can we deal with this?

Interval Orders and Semiorders

A binary relation $\succ$ on $\mathcal{X}$ is called an *interval order* if there exists a mapping from $\mathcal{X}$ to intervals on the real line such that $x \succ y$ *iff* the interval corresponding to x lies strictly to the left from that corresponding to y.

An interval order $\succ$ induces a (non-transitive) *indifference* relation $\sim$:
$x \sim y$ *iff* neither $x \succ y$ nor $y \succ x$ (symmetric complement).
(Note that $\succeq := \succ \cup \sim$ is complete but need not be transitive.)



$x \sim y$ and $y \sim z$
but $x \succ z$

A *semiorder* is an interval order where all intervals have the same length.

For more on interval- and semiorders, including applications in social choice theory, consult the book by Pirlot and Vincke (1997).

M. Pirlot and Ph. Vincke. *Semiorders: Properties, Representations, Applications*. Kluwer Academic Publishers, 1997.

Explicit Representation

How costly is it to *represent* a given preference structure? It depends on the *language* used. For now, consider *explicit representations* only':

- We can represent a *utility function* $u : \mathcal{X} \to Val$ as a table, listing the utility for every element of $\mathcal{X}$.

  It often is reasonable to assume that we need only a constant number of bits to represent any given value (e.g., if possible utility values are integers between 0 and 100, then we need 7 bits).

  $\Rightarrow O(|\mathcal{X}|)$: *linear* in the number of alternatives

- We can represent a *preorder* $\succeq$ as a list of pairs in $\mathcal{X} \times \mathcal{X}$, listing all those pairs that belong to $\succeq$.

  $\Rightarrow O(|\mathcal{X}|^2)$: *quadratic* in the number of alternatives

Both are fine as long as $\mathcal{X}$ is small, and problematic otherwise.

Combinatorial Domains

A *combinatorial domain* is a Cartesian product $\mathcal{D} = D_1 \times \cdots \times D_p$ of p finite domains. Many collective decision-making problems of practical interest have a combinatorial structure:

- During a *referendum* (in Switzerland, California, places like that), voters may be asked to vote on p different propositions.
- Elect a *committee* of k members from amongst n candidates.
- Find a good *allocation* of p indivisible goods to agents.

Seemingly small problems generate huge numbers of alternatives:

- $\binom{10}{3} = 120$ possible 3-member committees from 10 candidates; i.e., $120! \approx 6.7 \times 10^{198}$ possible linear (more weak) orders
- Allocating 10 goods to 5 agents: $5^{10} = 9765625$ allocations and $2^{10} = 1024$ bundles for each agent to think about

Therefore: we need good *languages* for representing preferences!

Conditional Preference Networks: Partial Orders

Associate each D_i in the combinatorial domain with a variable X_i.
A *CP-net* over a set of variables $\{X_1, \ldots, X_p\}$ consists of

- a *directed graph* G over $\{X_1, \ldots, X_p\}$ and
- a *conditional preference table* (CPT) for each X_i, fixing a total order on values of X_i for each instantiation of X_i's parents in G.

This induces a (partial) preference order:

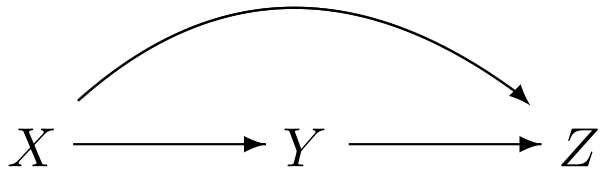
(1) If $\vec{y}$ is an instantiation of the parents of X, and $\vec{y} : x \succ x'$ is part of the CPT for X, then prefer x to x' given $\vec{y}$, *ceteris paribus*, i.e., prefer $x\vec{y}\vec{z}$ to $x'\vec{y}\vec{z}$ for any instantiation $\vec{z}$ of all other variables.

(2) Take the transitive closure of the above.

C. Boutilier, R.I. Brafman, C. Domshlak, H.H. Hoos, and D. Poole. CP-nets: A Tool for Representing and Reasoning with Conditional *Ceteris Paribus* Preference Statements. *Journal of Artificial Intelligence Research*, 21:135–191, 2004.

Example

Consider the following CP-net, consisting of a graph on $\{X, Y, Z\}$ and conditional preference tables for X, Y and Z:

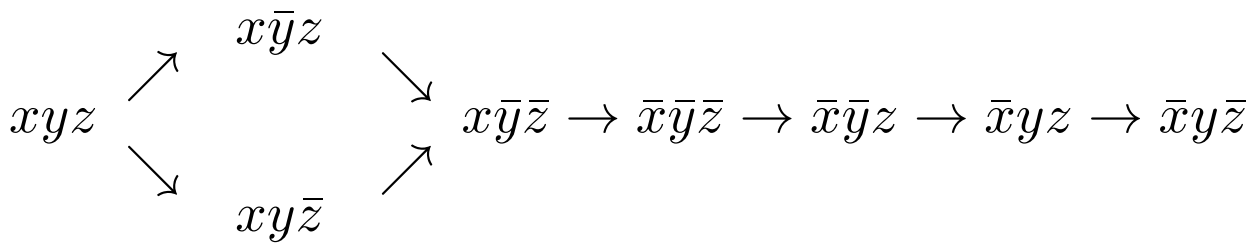


$$x \succ \bar{x}$$

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

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

This CP-net induces the following (partial) preference order:



Related Languages

- *CP-theories* are a generalisation of CP-nets by relaxing the "everything else being equal" condition (Wilson, 2004).
- Conditional importance networks (*CI-nets*) define preference orders by allowing the user to specify the relative importance of variables, given certain conditions (Bouveret et al., 2009).

N. Wilson. Extending CP-nets with Stronger Conditional Preference Statements. Proc. AAAI-2004.

S. Bouveret, U. Endriss, and J. Lang. Conditional Importance Networks: A Graphical Language for Representing Ordinal, Monotonic Preferences over Sets of Goods. Proc. IJCAI-2009.

Prioritised Goals: Weak Orders

Consider a *binary* comb. domains $\mathcal{D} = D_1 \times \cdots \times D_p$, with $|D_i| = 2$. Associate $\mathcal{D}$ with a set $PS = \{X_1, \ldots, X_p\}$ of propositional variables, i.e., identify truth assignments to PS (models) with elements of $\mathcal{D}$.

Use propositional formulas to express *goals* and use *numbers* to indicate their importance. This induces a weak order:

> Suppose (φ, k_1) has higher *priority* than (ψ, k_2) if $k_1 > k_2$.
>
> Under the *lexicographic* form of aggregation, we prefer M to M' if there exists a k such that for all $j > k$ both M and M' satisfy the same number of goals of rank j, and M satisfies more goals of rank k.

Other forms of aggregation are possible.

J. Lang. Logical Preference Representation and Combinatorial Vote. *Annals of Mathematics and Artificial Intelligence*, 42(1–3):37–71, 2004.

Weighted Goals: Utility Functions

Consider again a binary combinatorial domains and the propositional language $\mathcal{L}_{PS}$ with propositional variables PS declared on it.

We now want to model utility functions $u : 2^{PS} \to \mathbb{R}$.

Use formulas in $\mathcal{L}_{PS}$ to express *goals*. Give each goal a numerical *weight*. A *goalbase* $G = \{(\varphi_i, w_i)\}_i$ is a set of weighted goals.

Each goalbase G generates a utility function u_G:

$$u_G(M) \quad = \quad \sum_{(\varphi,w)\in G[M]} w \quad \text{where } G[M] = \{(\varphi, w) \in G \mid M \models \varphi\}$$

That is, the utility of a model M is the sum of the weights of the formulas it satisfies.

Other Languages for Modelling Utility Functions

- We can also use *aggregators other than sum-taking* (e.g., max) to compute a utility value from the weights of the satisfied goals (Lafage and Lang, 2000; Uckelman and Endriss, 2010).
- OR/XOR *bidding languages* developed for combinatorial auctions can express monotonic utility functions (Nisan, 2006).
- Can also represent utility functions as *programs*, with the alternative being the input and the utility value the output (Dunne et al., 2005).

C. Lafage and J. Lang. Logical Representation of Preferences fro Group Decision Making. Proc. KR-2000.

J. Uckelman and U. Endriss. Compactly Representing Utility Functions Using Weighted Goals and the Max Aggregator. *Artif. Intell.*, 174(15):1222–1246, 2010.

N. Nisan. Bidding Languages. In: *Combinatorial Auctions*, MIT Press, 2006.

P.E. Dunne, M. Wooldridge, and M. Laurence. The Complexity of Contract Negotiation. *Artificial Intelligence*, 164(1–2):23–46, 2005.

Criteria for Choosing a Language

At least the following questions should be addressed when choosing a representation language:

- *Cognitive relevance:* How close is a given language to the way in which humans would express their preferences?
- *Elicitation:* How difficult is it to elicit the preferences of an agent so as to represent them in the chosen language?
- *Expressive power:* Can the chosen language encode all the preference structures we are interested in?
- *Succinctness:* Is the representation of (typical) structures succinct? Is one language more succinct than the other?
- *Complexity:* What is the computational complexity of related decision problems, such as comparing two alternatives?

We are going to concentrate on the final three. And we will exemplify these questions for the family of weighted goals.

A Family of Languages

Recall: $\{(\varphi_i, w_i)\}_i$ induces $u : M \mapsto \sum\{w_i \mid M \models \varphi_i\}$ (multiset)

By imposing different restrictions on formulas and/or weights we can design different representation languages.

Regarding *formulas*, we may consider restrictions such as:

- *positive* formulas (no occurrence of $\neg$)
- *clauses* and *cubes* (disjunctions and conjunctions of literals)
- *k-formulas* (formulas of length $\leqslant k$), e.g. 1-formulas = literals
- combinations of the above, e.g. k-pcubes

Regarding *weights*, interesting restrictions would be $\mathbb{R}^+$ or $\{0, 1\}$.

If $H \subseteq \mathcal{L}_{PS}$ is a restriction on formulas and $H' \subseteq \mathbb{R}$ a restriction on weights, then $\mathcal{L}(H, H')$ is the language conforming to H and H'.

Properties

We are interested in the following types of questions:

- Are there restrictions on goalbases such that the utility functions they generate enjoy natural structural properties?
- Are some goalbase languages more succinct than others?
- What is the computational complexity of reasoning about preferences expressed in a given language?

The results on the following slides are from Uckelman et al. (2009). More details in Joel Uckelman's PhD thesis.

J. Uckelman, Y. Chevaleyre, U. Endriss, and J. Lang. Representing Utility Functions via Weighted Goals. *Mathematical Logic Quarterly*, 55(4):341–361, 2009.

J. Uckelman. *More Than the Sum of its Parts: Compact Preference Representation over Combinatorial Domains*. PhD thesis, ILLC, University of Amsterdam, 2009.

Expressive Power

We first give an example for a language that is fully expressive:

Proposition 1 (Expressivity of pcubes) *$\mathcal{L}(pcubes, \mathbb{R})$, the language of positive cubes, can express all utility functions.*

Proof: Let $u : 2^{PS} \to \mathbb{R}$ be any utility function. Define goalbase G:

$(\top, w_\top)$ with $w_\top = u(\emptyset)$
(p, w_p) with $w_p = u(\{p\}) - w_\top$
$(p \wedge q, w_{p,q})$ with $w_{p,q} = u(\{p, q\}) - w_p - w_q - w_\top$ ...
$(\bigwedge P, w_P)$ with $w_P = u(P) - \sum_{Q \subset P} w_Q$

Clearly, G thus defined will generate the function u. ✓

Observe that the proof also demonstrates that $\mathcal{L}(pcubes, \mathbb{R})$ has a *unique* way of representing any given function.

$\mathcal{L}(cubes, \mathbb{R})$, for example, is also fully expressive but *not unique:*
$\{(p \wedge q, 5), (p \wedge \neg q, 5), (\neg p \wedge q, 3), (\neg p \wedge \neg q, 3)\} \equiv \{(\top, 3), (p, 2)\}$

Expressive Power: Modular Functions

A function $u : 2^{PS} \to \mathbb{R}$ is *modular* if for all $M_1, M_2 \subseteq 2^{PS}$ we have:

$$u(M_1 \cup M_2) = u(M_1) + u(M_2) - u(M_1 \cap M_2)$$

Here's a nice characterisation of the modular functions:

Proposition 2 (Expressivity of literals) *$\mathcal{L}(literals, \mathbb{R})$ can express all modular utility functions, and only those.*

Proof sketch: Modular functions are like *additive* functions, i.e., functions u with $u(M) = \sum_{x \in M} u(x)$, except that we also allow non-zero values for $\emptyset$.

Easy to see that $\mathcal{L}(atoms, \mathbb{R})$ expresses exactly the additive functions.

So $\mathcal{L}(atoms \cup \{\top\}, \mathbb{R})$ expresses exactly the modular functions.

To see that adding negation does not increase expressive power, observe that $G \cup \{(\neg\varphi, w)\} \equiv G \cup \{(\top, w), (\varphi, -w)\}$. ✓

Expressive Power: Monotonic Functions

A function $u : 2^{PS} \to \mathbb{R}$ is *monotonic* if for all $M_1, M_2 \subseteq 2^{PS}$:

$$M_1 \subseteq M_2 \text{ implies } u(M_1) \leqslant u(M_2)$$

We state this next result without proof:

Proposition 3 (Expressivity of pforms/pos) $\mathcal{L}(pforms, \mathbb{R}^+)$, *the language of positive formulas with positive weights, can express all nonnegative monotonic utility functions, and only those.*

That positively weighted positive formulas can only generate monotonic functions is clear. The other direction the interesting bit.

Relative Succinctness

If two languages can express the same class of utility functions, which should we use? An important criterion is *succinctness*.

Let $\mathcal{L}$ and $\mathcal{L}'$ be two languages that can define all utility functions belonging to some class $\mathcal{U}$.

We say that $\mathcal{L}'$ is at least as succinct as $\mathcal{L}$ ($\mathcal{L} \preceq \mathcal{L}'$) over $\mathcal{U}$ if there exist a mapping $f : \mathcal{L} \to \mathcal{L}'$ and a *polynomial* function p such that for all expressions $G \in \mathcal{L}$ with the corresponding function $u_G \in \mathcal{U}$:

- $G \equiv f(G)$ (i.e., they represent the same function: $u_G = u_{f(G)}$);
- and $size(f(G)) \leqslant p(size(G))$ (polysize reduction).

$\mathcal{L}$ is *less succinct* than $\mathcal{L}'$ ($\mathcal{L} \prec \mathcal{L}'$) iff $\mathcal{L} \preceq \mathcal{L}'$ and not $\mathcal{L}' \preceq \mathcal{L}$.
Equivalence ($\sim$) and *incomparability* ($\bowtie$) are defined accordingly.

If left implicit, then $\mathcal{U}$ is the intersection of the ranges of $\mathcal{L}$ and $\mathcal{L}'$.

(Note the overloading of the symbols $\succeq$, etc.)

The Effect of Negation

We have seen that positive cubes are fully expressive. Hence, $\mathcal{L}(pcubes, \mathbb{R})$ and $\mathcal{L}(cubes, \mathbb{R})$ have the same expressivity.

Proposition 4 (Succinctness) $\mathcal{L}(pcubes, \mathbb{R}) \prec \mathcal{L}(cubes, \mathbb{R})$.

Proof: Clearly, $\mathcal{L}(pcubes, \mathbb{R}) \preceq \mathcal{L}(cubes, \mathbb{R})$, because any positive cube is also a cube.

Now consider u with $u(\emptyset) = 1$ and $u(M) = 0$ for all $M \neq \emptyset$:

- $G = \{(\neg p_1 \wedge \cdots \wedge \neg p_n, 1)\} \in \mathcal{L}(cubes, \mathbb{R})$ has *linear* size and generates u.
- $G' = \{(\bigwedge X, (-1)^{|X|}) \mid X \subseteq PS\} \in \mathcal{L}(pcubes, \mathbb{R})$ has *exponential* size and also generates u.

But there can be not better way of expressing u in pcubes, because we have seen that pcube representations are *unique*. ✓

More Succinctness Results

Some more examples for succinctness results (without proof):

- Cubes and clauses are equally succinct:

  Proposition 5 $\mathcal{L}(cubes, \mathbb{R}) \sim \mathcal{L}(clauses, \mathbb{R})$

- But *positive* cubes and clauses are incomparable:

  Proposition 6 $\mathcal{L}(pcubes, \mathbb{R}) \bowtie \mathcal{L}(pclauses, \mathbb{R})$

- If we restrict the weights, we get equal succinctness again:

  Proposition 7 $\mathcal{L}(pcubes, \mathbb{R}^+) \sim \mathcal{L}(pclauses, \mathbb{R}^+)$

  In fact, this result is more about expressivity: it is a corollary of other results showing that only modular functions are expressible in *both* $\mathcal{L}(pcubes, \mathbb{R}^+)$ and $\mathcal{L}(pclauses, \mathbb{R}^+)$.

- We gain further succinctness as we enrich the formula language:

  Proposition 8 $\mathcal{L}(cubes, \mathbb{R}) \prec \mathcal{L}(forms, \mathbb{R})$

Computational Complexity

Other interesting questions concern the complexity of reasoning about preferences. Consider the following decision problem:

$\textsc{Max-Utility}(H, H')$
Instance: Goalbase $G \in \mathcal{L}(H, H')$ and $K \in \mathbb{Z}$
Question: Is there an $M \in 2^{PS}$ such that $u_G(M) \geqslant K$?

Some basic results are straightforward:

- $\textsc{Max-Utility}(H, H')$ is *in NP* for any $H \subseteq \mathcal{L}_{PS}$ and $H' \subseteq \mathbb{Q}$, because we can always check $u_G(M) \geqslant K$ in polynomial time.
- $\textsc{Max-Utility}(\mathit{forms}, \mathbb{Q})$ is *NP-complete*, certainly if we do not assume that formulas are satisfiable (reduction from $\textsc{Sat}$).

More interesting questions would be: are there either (1) "large" sublanguages for which $\textsc{Max-Utility}$ is still polynomial, or (2) "small" sublanguages for which it is already NP-hard?

Three Complexity Results

Proposition 9 $\textsc{Max-Utility}(k\text{-}cubes, \mathbb{Q})$ *is NP-complete, even for* $k = 2$ *and assuming all formulas are satisfiable.*

Proof: By reduction from $\textsc{Max2Sat}$ (NP-hard): "Given a set of 2-clauses, is there a satisfiable subset with cardinality $\geqslant K$?". Given a $\textsc{Max2Sat}$ instance, for each clause $p \vee q$ create a goal $(\neg p \wedge \neg q, -1)$. Add $(\top, N)$, where N is the number of clauses. Answer YES for the $\textsc{Max2Sat}$ instance iff maximal utility is $\geqslant K$. ✓

Proposition 10 $\textsc{Max-Utility}(literals, \mathbb{Q})$ *is in P.*

Proof: Assuming that G contains every literal exactly once (possibly with weight 0), making p true *iff* the weight of p is greater than the weight of $\neg p$ results in a model with maximal utility. ✓

Proposition 11 $\textsc{Max-Utility}(pforms, \mathbb{Q}^+)$ *is in P.*

Proof: Making *all* variables true yields maximal utility. ✓

Summary

In the first part of today's lecture we have seen that there are different *preference structures* we might want to work with:

- preorders, including partial orders, weak order, total (linear) orders
- interval orders and semiorders
- utility functions

We may or may not want completeness, transitivity, and the ability to express preference intensity or interpersonal comparison.

In the context of (social choice in) *combinatorial domains*, the choice of *representation language* is crucial. Options we have seen:

- CP-nets (for a subclass of the partial orders)
- Prioritised goals (for a subclass of the weak orders)
- Weighted goals (for the full class of utility functions)

The study of preference representation languages is a research area in its own right. We have seen results on expressivity, succinctness, and complexity.

What next?

Next week we will see several approaches to dealing with the problem of *social choice in combinatorial domains*, some of which will make use the the preference representation languages introduced today.