# Preference Representation with Weighted Formulas

Joel Uckelman

Institute for Logic, Language, and Computation
University of Amsterdam
juckelma@illc.uva.nl

Computational Social Choice, 19 March 2007

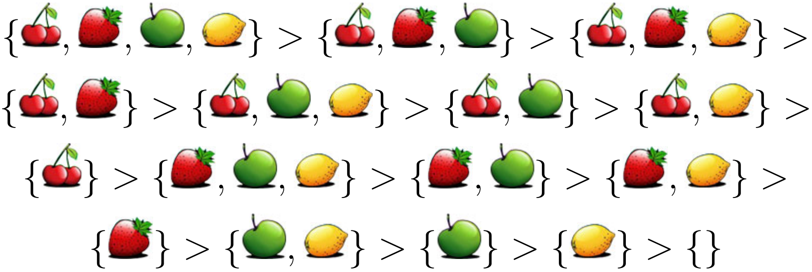
# Overview

| | |
|---|---|
| Introduction | Describe *weighted formulas* and *goal bases*. |
| | Review of properties of utility functions. |
| Expressivity | Discussion of *restrictions* on goal base languages, and their *correspondence* with properties of utility functions. |
| Uniqueness | Demonstrate the *uniqueness* of representations in some languages. |
| Succinctness | Consider the relative *succinctness* (efficiency of representation) of several pairs of languages. |
| Complexity | Review *NP-hardness* and *-completeness*. |
| | Consider the difficulty of *finding optimal assignments* of goods in some languages (efficiency of computation). |
| Applications | An application of goal base languages to *committee voting*. |

# Preferences and the Combinatorial Explosion

Preference orders on sets of items have compact representations:

But many kinds of resource allocation problems require agents to have preference orderings over subsets of items:



# Efficient Representations

Given a set $F$ of fruits there will be $2^{|F|}$ subsets, which rapidly becomes too large to handle. If we need full preferences from agents, we have to do something which takes advantage of the structure of those preferences.

For example:

{(, 8), (, 4), (, 2), (, 1)}

So whenever I have , it's worth 4 to me, and so on. Since my preferences are *modular*, we can write them in a concise way which takes advantage of that.

(Note that we've moved from ordinal to cardinal preferences, which will be the subject of the rest of the lecture.)

# Weighted Formulas and Goal Bases

## Definitions

- A *weighted formula* is a pair $(\varphi, w)$, where $\varphi$ is a propositional formula and $w \in \mathbb{R}$.
- A *goal base* is a set of weighted satisfiable formulas.

## Examples

Goal bases:

$$\emptyset \qquad \{(p, 42)\} \qquad \{(\top, -2)\} \qquad \{(a, 1), (a \wedge a, 1)\}$$

$$\left\{(a \wedge b, -5), \left(\neg a \vee d, \frac{22}{7}\right)\right\}$$

Not a goal base:

$$\{(\bot, 3)\}$$

# Weighted Formulas and Goal Bases

## Definitions

- A *weighted formula* is a pair $(\varphi, w)$, where $\varphi$ is a propositional formula and $w \in \mathbb{R}$.
- A *goal base* is a set of weighted satisfiable formulas.

## Examples

Goal bases:

$$\emptyset \qquad \{(p, 42)\} \qquad \{(\top, -2)\} \qquad \{(a, 1), (a \wedge a, 1)\}$$

$$\left\{(a \wedge b, -5), \left(\neg a \vee d, \frac{22}{7}\right)\right\}$$

Not a goal base:

$$\{(\bot, 3)\}$$

# Goal Bases and Utility Functions

## Definitions

- $\mathcal{PS}$ is a finite set of propositional variables.
- A *utility function* is a mapping $u : 2^{\mathcal{PS}} \to \mathbb{R}$.
- A *model* is a set $M \subseteq \mathcal{PS}$ (i.e., just the true atoms).
- Every goal base $G$ generates a unique utility function $u_G$:

$$u_G(M) = \sum \{w : (\varphi, w) \in G \text{ and } M \models \varphi\}$$

# Expressivity: What Can I Say?

We can form a *goal base language* by taking any desired set of goal bases.

Given a goal base language, what utility functions can it *express*?

The goal base formalism suggests some subsets of goal bases to investigate:

- goal bases which use only a particular sort of formula, e.g., clauses or literals
- goal bases which use only a particular sort of weight, e.g., positive

Are there interesting *correspondences* between goal base languages and classes of utility functions?

# Classes of Goal Bases

### Definition

$\mathcal{U}(H, H')$ is the class of utility functions generated by goal bases meeting restrictions $H$ and $H'$.

Here, we let $H \subseteq \mathcal{L}_{\mathcal{PS}}$ restrict the formulas of a goal base, and $H' \subseteq \mathbb{R}$ restrict the weights.

### Examples

$$
\begin{array}{lcl}
\mathcal{U}(\mathit{atoms}, \mathit{pos}) & = & \text{atoms with positive weights} \\
\mathcal{U}(\mathit{literals}, \{0,1\}) & = & \text{literals with binary weights} \\
\mathcal{U}(\mathit{cubes}, \mathit{all}) & = & \text{cubes with arbitrary weights} \\
\mathcal{U}(\mathit{pclauses}, \mathit{neg}) & = & \text{positive clauses with negative weights}
\end{array}
$$

(Cubes and clauses are con- and disjunctions of literals, resp.)

# A Correspondence Theorem

## Theorem

*$\mathcal{U}(cubes, all)$ contains all utility functions.*

## Proof.

Given arbitrary $u$, define a corresponding $G$ by states:

$$G = \left\{ \begin{array}{l} (\;\;p_0 \wedge \;\;p_1 \wedge \;\;p_2 \wedge \ldots \wedge \;\;p_n,\; u(\mathcal{PS}) \qquad\qquad ), \\ (\neg p_0 \wedge \;\;p_1 \wedge \;\;p_2 \wedge \ldots \wedge \;\;p_n,\; u(\mathcal{PS} \setminus \{p_0\}) \quad ), \\ (\;\;p_0 \wedge \neg p_1 \wedge \;\;p_2 \wedge \ldots \wedge \;\;p_n,\; u(\mathcal{PS} \setminus \{p_1\}) \quad ), \\ (\neg p_0 \wedge \neg p_1 \wedge \;\;p_2 \wedge \ldots \wedge \;\;p_n,\; u(\mathcal{PS} \setminus \{p_0, p_1\})), \\ \qquad \vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ (\neg p_0 \wedge \neg p_1 \wedge \neg p_2 \wedge \ldots \wedge \neg p_n,\; u(\emptyset) \qquad\qquad ) \end{array} \right\}$$

$\square$

## Corollary

*$\mathcal{U}(all, all)$ is fully expressive.*

# $k$-Additivity

A utility function $u$ is *$k$-additive* if there is a mapping $m : [\mathcal{PS}]^k \to \mathbb{R}$ such that

$$u(X) = \sum\{m(Y) : Y \subseteq X \text{ and } Y \in [\mathcal{PS}]^k\}$$

$k$-additive utility functions are those where there are no interactions among subsets containing more than $k$ items.

E.g., if $u$ is 1-additive, then

$$u(\{a,b,c\}) = u(\emptyset) + u(\{a\}) - u(\emptyset) + u(\{b\}) - u(\emptyset) + u(\{c\}) - u(\emptyset)$$

which is the same as

$$u(\{a,b,c\}) = m(\emptyset) + m(\{a\}) + m(\{b\}) + m(\{c\})$$

$m(Y)$ is just the utility that the set $Y$ contributes whenever present. ($m$ is unique for each $u$. The map $u \mapsto m$ is called the *Möbius inversion*.)

# Another Correspondence Theorem

## Theorem (Chevaleyre, Endriss, & Lang, 2006)

$\mathcal{U}$(*positive k-cubes*, *all*) *is the class of k-additive utility functions.*

## Proof.

If $m$ is the $k$-additive mapping for $u$, define a goal base $G$ from it:

$$G = \{(p_1 \wedge \ldots \wedge p_j, w) : m(\{p_1, \ldots, p_j\}) = w \text{ and } j \leq k\}$$

Clearly, $u_G = u$.

Conversely, if $G \in \mathcal{U}$(*positive k-cubes*), then define $m$ from it:

$$m(X) = w \text{ for each } (\bigwedge X, w) \in G$$

Since every $\bigwedge X$ in $G$ is a $k$-clause, $m$ defines a $k$-additive function. □

Many expressivity results may be derived from this one...

# Expressivity Summary

| Formulas | Weights | | Class of Utility Functions | Reference |
|---|---|---|---|---|
| cubes | all | = | all | [CEL06, Prop. 4] |
| clauses | all | = | all | [CEL06, Prop. 4] |
| all | all | = | all | [CEL06, Prop. 4] |
| positive cubes | all | = | all | [CEL06, Prop. 4] |
| positive formulas | all | = | all | [CEL06, Prop. 4] |
| Horn | all | = | all | U&E |
| positive clauses | all | = | normalized | [CEL06, Prop. 5] |
| strictly positive formulas | all | = | normalized | [CEL06, Prop. 6] |
| $k$-cubes | all | = | $k$-additive | [CEL06, Prop. 2] |
| $k$-clauses | all | = | $k$-additive | [CEL06, Prop. 2] |
| $k$-formulas | all | = | $k$-additive | [CEL06, Prop. 2] |
| positive $k$-cubes | all | = | $k$-additive | [CEL06, Props. 1 & 2] |
| positive $k$-formulas | all | = | $k$-additive | [CEL06, Props. 1 & 2] |
| positive $k$-clauses | all | = | normalized $k$-additive | [CEL06, Prop. 3] |
| literals | all | = | modular | [CEL06, Prop. 7] |
| atoms | all | = | normalized modular | [CEL06, Prop. 8] |
| cubes | positive | = | nonnegative | [CEL06, Prop. 9] |
| formulas | positive | = | nonnegative | [CEL06, Prop. 9] |
| clauses | positive | $\subset$ | nonnegative | [CEL06, Prop. 10] |
| strictly positive formulas | positive | = | normalized monotonic | [CEL06, Prop. 11] |
| positive clauses | positive | $\subset$ | normalized concave monotonic | U&E |
| positive formulas | positive | = | nonnegative monotonic | U&E |

# Uniqueness of Representations

A language has *unique representations* if every utility function it can represent is generated by exactly one goal base in the language.

Languages with the uniqueness property are minimal with respect to the class of utility functions to which they correspond. *Any* further restrictions will reduce their expressivity.

Are there any such languages?

Yes, any language formed from a singleton class of goal bases is like this.

Are there any such (nontrivial!) languages?

# A Uniqueness Proof

### Theorem

$\mathcal{U}(pclauses, all)$ *has unique representations.*

### Proof

There are $2^{|\mathcal{PS}|} - 1$ nonequivalent positive clauses, enumerated

$$\varphi_j = \bigvee \{p_k : j \,\&\, 2^k = 1\}$$

Each model $i \in 2^{\mathcal{PS}}$ defines a constraint

$$a_{i1} w_1 + \ldots + a_{im} w_m = b_i$$

where $a_{ij} \in \{0, 1\}$ depending on whether clause $j$ is true in state $i$, and $b_i = u(X_i)$. Neglecting state $\emptyset$ (since $\bigvee \emptyset = \bot$ is not a positive clause), we have...

# A Uniqueness Proof II

...this system

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Any such system has a single unique solution when $\det(A) \neq 0$.

# A Uniqueness Proof III

So A looks like this, for $n = 1, 2, 3, \ldots$:

$$A_1 \qquad \begin{matrix} \boxed{1} \end{matrix}$$

$$A_2 \qquad \begin{matrix} \boxed{1} & 0 & \boxed{1} \\ 0 & 1 & 1 \\ \boxed{1} & 1 & 1 \end{matrix}$$

$$A_3 \qquad \begin{matrix} \boxed{1} & 0 & \boxed{1} & 0 & \boxed{1} & 0 & \boxed{1} \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \boxed{1} & 1 & 1 & 0 & \boxed{1} & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ \boxed{1} & 0 & \boxed{1} & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ \boxed{1} & 1 & 1 & 1 & 1 & 1 & 1 \end{matrix}$$

We can show that this pattern yields a nonzero determinant at all sizes, hence the system has exactly one solution at all sizes. Thus, $\mathcal{U}(\mathit{pclauses}, \mathit{all})$ has unique representations. □

# What Good Is Uniqueness?

- Some languages have multiple languages with the uniqueness property: $\mathcal{U}(pcubes, all)$ is also fully expressive and has unique representations (similar proof).
- Uniqueness is useful when examining succinctness of languages.

# Succinctness

A given utility function may be represented by different goal bases:
E.g., for $u(X) = 1$:

$$\{(\top, 1)\} \qquad \{(a \wedge b, 1), (a \wedge \neg b, 1), (\neg a \wedge b, 1), (\neg a \wedge \neg b, 1)\}$$

One goal base is more *succinct* than the other.

For some pairs of languages, *any* goal base representable in one will have a shorter representation in the other.

Succinctness measures how space-efficient languages are.

# A Definition of Succinctness

## Definition

$\mathcal{L} \preceq \mathcal{L}'$ ($\mathcal{L}'$ is at least as succinct as $\mathcal{L}$) iff there exist

- a function $f : \mathcal{L} \to \mathcal{L}'$, and
- a polynomial $p$

such that for all $G \in \mathcal{L}$

- $u_G = u_{f(G)}$, and
- $\text{size}(f(G)) \leq p(\text{size}(G))$

This is fine when $\mathcal{L}$ and $\mathcal{L}'$ are equally expressive, but using this definition we can't compare them otherwise.

# A More General Definition

## Definitions

- $\mathcal{U}(\mathcal{L}) = \{u_G : G \in \mathcal{L}\}$ (the u.f. represented in the language)
- $\mathrm{Rep}_{\mathcal{L}}(\mathcal{U}) = \{G \in \mathcal{L} : u_G \in \mathcal{U}\}$ (the $\mathcal{L}$-reps. of a class of u.f.)
- $\mathcal{L}_{\cap\mathcal{L}'} = \mathrm{Rep}_{\mathcal{L}}(\mathcal{U}(\mathcal{L}) \cap \mathcal{U}(\mathcal{L}'))$
  (the *expressive* intersection of $\mathcal{L}$ and $\mathcal{L}'$ in $\mathcal{L}$)

$\mathcal{L} \preceq \mathcal{L}'$ ($\mathcal{L}'$ is at least as succinct as $\mathcal{L}$) iff there exist

- a function $f : \mathcal{L}_{\cap\mathcal{L}'} \to \mathcal{L}'_{\cap\mathcal{L}}$, and
- a polynomial $p$

such that

- $u_G = u_{f(G)}$, and
- $\mathrm{size}(f(G)) \leq p(\mathrm{size}(G))$

for all $G \in \mathcal{L}_{\cap\mathcal{L}'}$.

# A Simple Strict Succinctness Result

## Theorem

$\mathcal{U}(\mathit{pclauses}, \mathit{all}) \prec \mathcal{U}(\mathit{clauses}, \mathit{all})$

## Proof.

$\mathcal{U}(\mathit{pclauses}, \mathit{all}) \preceq \mathcal{U}(\mathit{clauses}, \mathit{all})$: Every pclause is a clause.

Consider the family $u_n$ where $u_n(X) = \begin{cases} 1 & \text{if } X = \mathcal{PS} \\ 0 & \text{otherwise} \end{cases} \qquad (|\mathcal{PS}| = n)$

There is a linear clauses representation: $\{(\top, 1), (\bigvee\{\neg p : p \in \mathcal{PS}\}, -1)\}$

Here is an exponential representation in pclauses:

$$\left\{\left(\bigvee X, w_{\bigvee X}\right) : \emptyset \subset X \subseteq \mathcal{PS}\right\} \qquad w_{\bigvee X} = \begin{cases} 1 & \text{if } |X| \text{ is odd} \\ -1 & \text{if } |X| \text{ is even} \end{cases}$$

By uniqueness, this is the sole pclauses representation of $u$. $\square$

# A Not-Quite-So-Simple Equivalence Result

## Theorem

*If $\mathcal{L}_{cubes} \subseteq \Phi$ or $\mathcal{L}_{clauses} \subseteq \Phi$, $\mathcal{L}_{cubes} \subseteq \Psi$ or $\mathcal{L}_{clauses} \subseteq \Psi$, and $\Phi, \Psi \subseteq \mathcal{L}_{cubes \cup clauses}$, then $\mathcal{U}(\Phi, all) \sim \mathcal{U}(\Psi, all)$.*

## Proof

Suppose that $G \in \mathcal{U}(\Phi, \text{all})$. Enumerate $(\phi_i, w_i) \in G$ and construct an equivalent goal base $G'$:

$$G_0 = G$$

$$G_{i+1} = \begin{cases} (G_i \setminus \{(\phi_i, w_i)\}) \cup \{(\neg\phi_i, -w_i), (\top, w_i)\} & \text{if } \phi_i \notin \Psi \\ G_i & \text{otherwise} \end{cases}$$

and let $G' = G_{|G|}$.

The transformation produces an equivalent goal base at each stage: Notice that we can always replace a cube with a clause and $\top$, or a clause with a cube and $\top$. The negation of a cube is a clause (and vice versa), and $\top$ is both a cube ($\bigwedge \emptyset$) and a clause ($p \vee \neg p$). The final goal base, $G'$, is in the target language.

# A Not-Quite-So-Simple Equivalence Result II

The transformation produces a goal base as succinct as the original: If $\phi$ is a cube, then $\phi$ requires the same number of atoms and binary connectives as as $\neg\phi$ (written as a clause); similarly, if $\phi$ is a clause. The only increase in size from $G$ to $G'$ can come from the addition of $\top$, so we have that $|G'| \leq |G| + 1$.

Therefore, $\mathcal{U}(\Phi, \text{all}) \succeq \mathcal{U}(\Psi, \text{all})$, and by the same argument $\mathcal{U}(\Psi, \text{all}) \succeq \mathcal{U}(\Phi, \text{all})$; hence $\mathcal{U}(\Phi, \text{all}) \sim \mathcal{U}(\Psi, \text{all})$. $\square$

## Corollary

$\mathcal{U}(\mathit{cubes}, \mathit{all}) \sim \mathcal{U}(\mathit{clauses}, \mathit{all})$

# Succinctness Summary

| **Result** | | | | | **Reference** |
|---|---|---|---|---|---|
| positive clauses | all | $\prec$ | clauses | all | U&E |
| positive clauses | all | $\perp$ | positive cubes | all | U&E |
| cubes | all | $\prec$ | all | all | Chevaleyre |
| clauses | all | $\prec$ | all | all | U&E |
| clauses | all | $\sim$ | cubes | all | U&E |
| positive cubes | all | $\prec$ | cubes | all | [CEL06, Prop. 13] |
| complete cubes | all | $\perp$ | positive cubes | all | [CEL06, p. 150] |

# Finding Optimal Allocations

## Definition

The decision problem $\textsc{Max-Utility}(H, H')$ is defined as: Given a goal base $G \in \mathcal{U}(H, H')$ and an integer $K$, check whether there is a model $M \in 2^{\mathcal{PS}}$ where $u_G(M) \geq K$.

## Uses

- Finding an agent's most preferred state
- Finding an optimal state overall, by goal base summation
- Similar to Winner Determination Problem in auctions and voting

# $\textsc{Max-Util}$ Is Easy, Sometimes

First, notice that the complexity of testing whether $u_G(M) \geq K$ is linear in the size of $G$.

### Theorem

$\textsc{Max-Util}(\textit{positive}, \textit{positive}) \in P$

### Proof.

If $G \in \mathcal{U}(\textit{positive}, \textit{positive})$, then $\mathcal{PS}$ is at least as good as any other state, since $\mathcal{PS} \models \varphi$ and $w > 0$ for all $(\varphi, w) \in G$. Check whether $u_G(\mathcal{PS}) \geq K$. This is linear in the size of $G$. $\square$

### Theorem

$\textsc{Max-Util}(\textit{literals}, \textit{all}) \in P$

### Proof.

Suppose $G \in \mathcal{U}(\textit{literals}, \textit{all})$. Make one pass over $G$, keeping a running tally of the difference between $p$ and $\neg p$ weights for each $p \in \mathcal{PS}$. When done, put any $p$ with a positive difference in $M$. Check whether $u_G(M) \geq K$. This is polynomial in the size of $G$. $\square$

# NP-Completeness

A quick computational complexity refresher:

- *NP* is the class of decision problems for which solutions may be checked in polynomial time. That is, if I guess a solution, you can verify whether it's correct in polynomial time (where the polynomial is in the size of the input). Membership is an upper bound on complexity.
- A decision problem $D$ is *NP-hard* if it is at least as hard as every other problem in NP. Hardness is a lower bound on complexity.
- NP-hardness of a problem $D$ is usually demonstrated by transforming (polynomially!) a known NP-hard problem into $D$. This tells you that your problem is at least as hard as the known NP-hard problem, and so is NP-hard itself.
- A decision problem $D$ is *NP-complete* if both $D \in NP$ and $D$ is NP-hard.

# MAX-UTIL Is Hard, Sometimes

## Definition

The decision problem MAX $k$-CONSTRAINT SAT is defined as: Given a set $C$ of $k$-cubes in $\mathcal{PS}$ and an integer $K$, check whether there is a model $M \in 2^{\mathcal{PS}}$ which satisfies at least $K$ of the $k$-cubes in $C$.

MAX $k$-CONSTRAINT SAT is known to be NP-complete [ACGKMS99].

## Theorem

MAX-UTIL($k$*-cubes, positive*) *is NP-complete for* $k \geq 2$*.*

## Proof.

NP-membership: Given any $M, K$ we can polynomially check whether $u_G(M) \geq K$.

NP-hardness: We exhibit a polynomial reduction of MAX $k$-CONSTRAINT SAT to MAX-UTIL($k$-cubes, positive): Given a set $C$ of $k$-cubes and an integer $K$, construct a goal base $G = \{(c, 1) : c \in C\}$. Then there is a model $M$ satisfying at least $K$ $k$-cubes in $C$ iff there is a model $M$ (actually, the same $M$) for which $u_G(M) \geq K$. □

# Complexity Summary

| **Formulas** | **Weights** | MAX-UTILITY | **Reference** |
|---|---|---|---|
| cubes | all | NP | $\mathbb{R} \supset \mathbb{R}^+$ |
| clauses | all | NP | $\mathbb{R} \supset \mathbb{R}^+$ |
| all | all | NP | [CEL06, p. 151] |
| positive cubes | all | NP | positive cubes $\supset$ positive $k$-cubes |
| positive clauses | all | NP | positive clauses $\supset$ positive $k$-clauses |
| positive formulas | all | NP | positive formulas $\supset$ positive cubes |
| Horn | all | NP | Horn $\supset$ negative clauses |
| strictly positive formulas | all | NP | strictly positive formulas $\supset$ positive cubes |
| $k$-cubes | all | NP, $k \geq 2$ | $\mathbb{R} \supset \mathbb{R}^+$ |
| $k$-clauses | all | NP, $k \geq 2$ | $\mathbb{R} \supset \mathbb{R}^+$ |
| $k$-formulas | all | NP, $k \geq 2$ | $\mathbb{R} \supset \mathbb{R}^+$ |
| positive $k$-cubes | all | NP, $k \geq 2$ | U&E |
| positive $k$-clauses | all | NP, $k \geq 2$ | U&E |
| positive $k$-formulas | all | NP, $k \geq 2$ | $k$-formulas $\supset$ $k$-cubes |
| literals | all | P | [CEL06, p. 151] |
| atoms | all | P | atoms $\subset$ literals |
| cubes | positive | NP | cubes $\supset$ $k$-cubes |
| clauses | positive | NP | clauses $\supset$ $k$-clauses |
| Horn | positive | NP | Horn $\supset$ $k$-Horn |
| all | positive | NP | all $\supset$ $k$-clauses |
| positive cubes | positive | P | positive cubes $\subset$ positive formulas |
| positive formulas | positive | P | [CEL06, p. 151] |
| positive clauses | positive | P | positive clauses $\subset$ positive formulas |
| strictly positive formulas | positive | P | strictly positive formulas $\subset$ positive formulas |
| $k$-cubes | positive | NP, $k \geq 2$ | U&E |
| $k$-clauses | positive | NP, $k \geq 2$ | [CEL06, p. 151] |
| $k$-Horn | positive | NP, $k \geq 2$ | U&E |
| $k$-formulas | positive | NP, $k \geq 2$ | $k$-formulas $\supset$ $k$-clauses |
| positive $k$-cubes | positive | P | positive $k$-cubes $\subset$ positive formulas |
| positive $k$-formulas | positive | P | positive $k$-formulas $\subset$ positive formulas |
| positive $k$-clauses | positive | P | positive $k$-clauses $\subset$ positive formulas |

# So You Want to Elect a Committee... Are You *Sure*?

*We always carry out by committee anything in which any one of us alone would be too reasonable to persist.*

—Frank Moore Colby (1865–1925), American essayist

# Extending Single-Winner Voting Methods

Suppose that we want to elect a committee with $k$ seats from a field of $n$ candidates. How might we do it?

We could extend some single-winner voting method, such as

- *Plurality voting*: Each voter casts one vote for one candidate; the candidate receiving the most votes is the winner.
- *Approval voting*: Each voter casts a maximum of one vote for each candidate; the candidate receiving the most votes is the winner.

A naïve way of extending each would be to make the top $k$ candidates winners.

Neither method is very expressive:

- Plurality can express only preferences where one candidate has utility 1 and the rest utility 0.
- Approval can express preferences where a subset of candidates each has utility 1 and each candidate in the complement has utility 0.

Maybe we can do something else to better reflect voter preferences...

# How Similar Are Two Approval Ballots?

Consider approval ballots as vectors of 0s and 1s. The *Hamming distance* between two approval ballots is the number of places in which they differ.

## Example

The Hamming distance between 00111 and 10101 is 2.

Instead of forming the committee from the top $k$ vote-getters, we could make the winning committee the one which minimizes the sum of Hamming distances to the ballots cast:

$$\text{committee } c \text{ is a winner iff } \forall c' \in C, \sum_{b \in B} \mathrm{H}(c, b) \leq \sum_{b \in B} \mathrm{H}(c', b)$$

or which minimizes the maximum Hamming distance to any ballot:

$$\text{committee } c \text{ is a winner iff } \forall c' \in C, \max_{b \in B} \mathrm{H}(c, b) \leq \max_{b \in B} \mathrm{H}(c', b)$$

where $C$ is the set of possible committees and $B$ the set of ballots cast. [BKS06]

Problem: Do voters have the same similarity metric?

# Goal Bases as Ballots

Plurality voting and approval voting may be done with the goal base languages $\mathcal{U}(atom, \{1\})$ and $\mathcal{U}(atoms, \{1\})$, respectively.

E.g., $\{(\text{Gore}, 1)\}$ is a plurality ballot, and $\{(\text{Gore}, 1), (\text{Nader}, 1)\}$ is an approval ballot.

We can find the winner of an election using goal base ballots by summing the goal bases:

$$G \oplus G' = \left\{ \left( \varphi, \sum_{(\varphi,a)\in G} a + \sum_{(\varphi,b)\in G'} b \right) : \varphi \in \text{For}(G \cup G') \right\}$$

and then using $\textsc{Max-Util}$ with increasing $K$ to find an optimal state, disregarding states which contain an inappropriate number of atoms.

(Notice that $\{(p, 1)\} \oplus \{(p \wedge p, 1)\} \neq \{(p, 2)\}$.)

# A Committee Election Example

Common multi-winner voting systems cannot express nonmodular preferences (same problem we had with the fruit basket at the start).

Suppose we have a voter with preferences like this:

$$\text{Alice}, \text{Bob} > \text{neither} > \text{both}$$

What ballot should this voter cast when using plurality or approval voting?

When we express ballots as goal bases, we have an obvious way to get more expressivity: Use more formulas! Use more weights!

A voter with these preferences could express them like so in the full language:

$$\{(a \vee b, 1), (a \wedge b, -2)\}$$

Fine, but we can't let voters submit *any* goal bases as ballots...

# Committee Voting—What Next?

- Isn't Max-Util too hard for committee voting? No, because complexity depends on # of candidates and seats, not # of voters.
- What languages are good for committee voting? In terms of complexity? In terms of expressivity? In terms of ease of use for voters?
- Can we avoid Gibbard-Satterthwaite?

  Recall that G-S says (details omitted) that for $\geq 3$ candidates, every voting rule is dictatorial or manipulable. G-S relies on the assumption that a voter be able to cast a sincere ballot. Suppose that we take a restricted language, so that voters have no sincere option. If we put a distance metric on ballots, we could then call the set of ballots nearest to the voter's true preferences the most sincere ones. *Maybe* we could get a weak form of strategyproofness this way, by expanding the number of ballots which count as sincere.
- How hard is (standard) manipulation? Probably quite hard, given a reasonable voting language.

# References

[ACGKMS99] Ausiello, G.; Crescenzi, P.; Gambosi, G.; Kann, V.; Marchetti-Spaccamela, A.; and Protasi, M.
*Complexity and Approximation*.
Springer-Verlag. 1999.

[BKS06] Steven J. Brams, D. Marc Kilgour, and M. Remzi Sanver.
A minimax procedure for electing committees.
In Denis Bouyssou, Fred Roberts, and Alexis Tsoukiàs, editors, *Proceedings of the DIMACS-LAMSADE Workshop on Voting Theory And Preference Modelling*, volume 6 of *Annales du LAMSADE*, pages 77–104, Paris, October 2006. Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision.

[CEL06] Chevaleyre, Y.; Endriss, U.; and Lang, J.
Expressive power of weighted propositional formulas for cardinal preference modelling.
In *Proc. 10th Intl. Conference on Principles of Knowledge Representation and Reasoning (KR-2006)*, 145–152. 2006.