Lifting Rationality Assumptions in Binary Aggregation

Umberto Grandi   Ulle Endriss

Institute for Logic, Language and Computation
University of Amsterdam

12 April 2010

Lifting Rationality Assumptions

Axiomatic Method	Collective Rationality
Independence,	Transitivity,
Neutrality,	Completeness,
...	Consistency of judgments...

Lifting Rationality Assumptions

Axiomatic Method		Collective Rationality
Independence,	$\Rightarrow$	Transitivity,
Neutrality,		Completeness,
...	$\Leftarrow$	Consistency of judgments...

There is a correlation between the two colums:

Depending on the shape of the requirement (shape? use a logical language) different axioms are necessary to preserve this property in the aggregation.

Binary Aggregation

The setting:

- $\mathcal{I} = \{1, \ldots, m\}$ a set of issues;
- A set N of individuals.
- Boolean *combinatorial* domain: $\mathcal{D} = D_1 \times \cdots \times D_m$ with $|D_i| = 2$;

Definition

An aggregation procedure is a function $F : \mathcal{D}^N \rightarrow \mathcal{D}$ *mapping each profile of ballots* $\underline{B} = (\underline{B}_1, \ldots, \underline{B}_n)$ *to an element of the domain* $\mathcal{D}$*.*

Binary Aggregation

The setting:

- $\mathcal{I} = \{1, \ldots, m\}$ a set of issues;
- A set N of individuals.
- Boolean *combinatorial* domain: $\mathcal{D} = D_1 \times \cdots \times D_m$ with $|D_i| = 2$;

Definition

An aggregation procedure is a function $F : \mathcal{D}^N \to \mathcal{D}$ mapping each profile of ballots $\underline{B} = (\underline{B}_1, \ldots, \underline{B}_n)$ to an element of the domain $\mathcal{D}$.

Many frameworks can be expressed as binary aggregation problems:

- Pairwise preference aggregation: issues are $'a > b'$ for all alternatives a, b;
- Judgment aggregation: the agenda is the set of issues;
- Voting for referenda;
- etcetc...

Languages for Integrity Constraints

We define a language to express properties of ballots (elements of $\mathcal{D}$):

- One propositional symbol for every issue: $PS = \{p_1, \dots, p_m\}$
- $\mathcal{L}_{PS}$ is the propositional language (closing under connectives $\wedge$, $\vee$,$\neg$, $\rightarrow$) generated from the propositional symbols PS.

An element of the domain $\mathcal{D}$ is a model for $\mathcal{L}_{PS}$: $\mathcal{D} = \{0, 1\}^m$.

Languages for Integrity Constraints

We define a language to express properties of ballots (elements of $\mathcal{D}$):

- One propositional symbol for every issue: $PS = \{p_1, \ldots, p_m\}$
- $\mathcal{L}_{PS}$ is the propositional language (closing under connectives $\wedge$, $\vee$,$\neg$, $\rightarrow$) generated from the propositional symbols PS.

An element of the domain $\mathcal{D}$ is a model for $\mathcal{L}_{PS}$: $\mathcal{D} = \{0, 1\}^m$.

Example: voting for a referendum.

Two bills between b_1, b_2 and b_3 have to be approved/disproved
Budget constraint: IC $= \neg(p_1 \wedge p_2 \wedge p_3)$, there is budget only for two of them

Languages for Integrity Constraints

We define a language to express properties of ballots (elements of $\mathcal{D}$):

- One propositional symbol for every issue: $PS = \{p_1, \ldots, p_m\}$
- $\mathcal{L}_{PS}$ is the propositional language (closing under connectives $\wedge$, $\vee$,$\neg$, $\rightarrow$) generated from the propositional symbols PS.

An element of the domain $\mathcal{D}$ is a model for $\mathcal{L}_{PS}$: $\mathcal{D} = \{0, 1\}^m$.

Example: voting for a referendum.

Two bills between b_1, b_2 and b_3 have to be approved/disproved
Budget constraint: IC $= \neg(p_1 \wedge p_2 \wedge p_3)$, there is budget only for two of them

Individual 1 submit $B_1 = (1, 1, 0)$: it satisfies IC ✓

Languages for Integrity Constraints

We define a language to express properties of ballots (elements of $\mathcal{D}$):

- One propositional symbol for every issue: $PS = \{p_1, \dots, p_m\}$
- $\mathcal{L}_{PS}$ is the propositional language (closing under connectives $\wedge$, $\vee$,$\neg$, $\rightarrow$) generated from the propositional symbols PS.

An element of the domain $\mathcal{D}$ is a model for $\mathcal{L}_{PS}$: $\mathcal{D} = \{0, 1\}^m$.

Example: voting for a referendum.

Two bills between b_1, b_2 and b_3 have to be approved/disproved
Budget constraint: $\text{IC} = \neg(p_1 \wedge p_2 \wedge p_3)$, there is budget only for two of them

Individual 1 submit $B_1 = (1, 1, 0)$: it satisfies IC ✓
Individual 2 submit $B_2 = (0, 1, 1)$: $B_2 \models \text{IC}$ ✓
Individual 3 submit $B_3 = (1, 0, 1)$: $B_3 \models \text{IC}$ ✓

Languages for Integrity Constraints

We define a language to express properties of ballots (elements of $\mathcal{D}$):

- One propositional symbol for every issue: $PS = \{p_1, \ldots, p_m\}$
- $\mathcal{L}_{PS}$ is the propositional language (closing under connectives $\wedge$, $\vee$,$\neg$, $\rightarrow$) generated from the propositional symbols PS.

An element of the domain $\mathcal{D}$ is a model for $\mathcal{L}_{PS}$: $\mathcal{D} = \{0, 1\}^m$.

Example: voting for a referendum.

Two bills between b_1, b_2 and b_3 have to be approved/disproved
Budget constraint: $\text{IC} = \neg(p_1 \wedge p_2 \wedge p_3)$, there is budget only for two of them

Individual 1 submit $B_1 = (1, 1, 0)$: it satisfies IC ✓
Individual 2 submit $B_2 = (0, 1, 1)$: $B_2 \models \text{IC}$ ✓
Individual 3 submit $B_3 = (1, 0, 1)$: $B_3 \models \text{IC}$ ✓

Majority approves all three bills: IC not satisfied!

Definition

A language for integrity constraints over a domain $\mathcal{D}$ is a subset $\mathcal{L} \subset \mathcal{L}_{PS}$.

IC of previous examples in the language $\mathcal{L}_{3\text{-}cubes}$: disjunction of lenght 3.

Collective Rationality

Definition

A language for integrity constraints over a domain $\mathcal{D}$ is a subset $\mathcal{L} \subset \mathcal{L}_{PS}$.

IC of previous examples in the language $\mathcal{L}_{\textit{3-cubes}}$: disjunction of lenght 3.

We suppose every individual satisfies the same rationality assumption, i.e., submits ballots B satisfying the same integrity constraint IC.

Collective Rationality

Definition

A language for integrity constraints over a domain $\mathcal{D}$ is a subset $\mathcal{L} \subset \mathcal{L}_{PS}$.

IC of previous examples in the language $\mathcal{L}_{\textit{3-cubes}}$: disjunction of lenght 3.

We suppose every individual satisfies the same rationality assumption, i.e., submits ballots B satisfying the same integrity constraint IC.

Definition

Call an aggregation procedure F collectively rational for IC $\in \mathcal{L}_{PS}$ if for all profiles $\underline{B}$ such that $\underline{B}_i \models$ IC for all $i \in N$ we have that $F(\underline{B}) \models$ IC.

F is collectively rational if it lifts the rationality assumption given by IC from the individual to the collective level.

Axioms

Aggregation procedures have been studied using the axiomatic method, listing axioms as desirable properties of the functions.

Classical axioms from social choice theory can be translated in this framework:

Unanimity (U): For any profile $\underline{B} \in X^N$ and any $x \in \{0, 1\}$, if $\underline{B}_{i,j} = x$ for all $i \in N$, then $F(\underline{B})_j = x$.

Independence (I): For any issue $j \in \mathcal{I}$ and any two profiles $\underline{B}, \underline{B}' \in X^N$, if $\underline{B}_{i,j} = \underline{B}'_{i,j}$ for all $i \in N$, then $F(\underline{B})_j = F(\underline{B}')_j$.

Axioms

Aggregation procedures have been studied using the axiomatic method, listing axioms as desirable properties of the functions.

Classical axioms from social choice theory can be translated in this framework:

Unanimity (U): For any profile $\underline{B} \in X^N$ and any $x \in \{0,1\}$, if $\underline{B}_{i,j} = x$ for all $i \in N$, then $F(\underline{B})_j = x$.

Independence (I): For any issue $j \in \mathcal{I}$ and any two profiles $\underline{B}, \underline{B}' \in X^N$, if $\underline{B}_{i,j} = \underline{B}'_{i,j}$ for all $i \in N$, then $F(\underline{B})_j = F(\underline{B}')_j$.

New axioms are also defined, like the following generalisation of May (1952) neutrality axiom:

Domain-Neutrality ($\mathrm{N}^{\mathcal{D}}$): For any two issues $j, j' \in \mathcal{I}$ and any profile $\underline{B} \in X^N$, if $\underline{B}_{i,j} = 1 - \underline{B}_{i,j'}$ for all $i \in N$, then $F(\underline{B})_j = 1 - F(\underline{B})_{j'}$.

Results (template)

Different lists of axioms AX define classes of functions:

$$\mathcal{F}[\mathrm{AX}] \;=\; \{F : \mathcal{D}^N \to \mathcal{D} \mid F \text{ satisfies } \mathrm{AX}\}$$

Results (template)

Different lists of axioms AX define classes of functions:

$$\mathcal{F}[\mathrm{AX}] \;=\; \{F : \mathcal{D}^N \to \mathcal{D} \mid F \text{ satisfies } \mathrm{AX}\}$$

Axioms are domain dependent, domains of interest are defines via IC:

$$\mathcal{F}_{\mathcal{L}}[\mathrm{AX}] \;=\; \{F : \mathcal{D}^N \to \mathcal{D} \mid F_{\restriction \mathrm{Mod}(\mathsf{IC})^N} \text{ sat. } \mathrm{AX} \text{ for all } \mathsf{IC} \in \mathcal{L}\}$$

Results (template)

Different lists of axioms AX define classes of functions:

$$\mathcal{F}[\mathrm{AX}] \;=\; \{F : \mathcal{D}^N \rightarrow \mathcal{D} \mid F \text{ satisfies } \mathrm{AX}\}$$

Axioms are domain dependent, domains of interest are defines via IC:

$$\mathcal{F}_{\mathcal{L}}[\mathrm{AX}] \;=\; \{F : \mathcal{D}^N \rightarrow \mathcal{D} \mid F_{\restriction \mathrm{Mod}(\mathsf{IC})^N} \text{ sat. } \mathrm{AX} \text{ for all } \mathsf{IC} \in \mathcal{L}\}$$

The class of procedures that lift integrity constraint in a given language is:

$$\mathcal{CR}[\mathcal{L}] \;=\; \{F : \mathcal{D}^N \rightarrow \mathcal{D} \mid F \text{ is CR for all IC} \in \mathcal{L}\}$$

Results (template)

Different lists of axioms AX define classes of functions:

$$\mathcal{F}[\mathrm{AX}] \;=\; \{F : \mathcal{D}^N \rightarrow \mathcal{D} \mid F \text{ satisfies } \mathrm{AX}\}$$

Axioms are domain dependent, domains of interest are defines via IC:

$$\mathcal{F}_{\mathcal{L}}[\mathrm{AX}] \;=\; \{F : \mathcal{D}^N \rightarrow \mathcal{D} \mid F_{\restriction \mathrm{Mod}(\mathsf{IC})^N} \text{ sat. } \mathrm{AX} \text{ for all } \mathsf{IC} \in \mathcal{L}\}$$

The class of procedures that lift integrity constraint in a given language is:

$$\mathcal{CR}[\mathcal{L}] \;=\; \{F : \mathcal{D}^N \rightarrow \mathcal{D} \mid F \text{ is CR for all } \mathsf{IC} \in \mathcal{L}\}$$

What we seek are results of this form:

$$\mathcal{CR}[\mathcal{L}] = \mathcal{F}_{\mathcal{L}}[\mathrm{AX}]$$

Results (examples)

Proposition

$\mathcal{CR}[cubes] = \mathcal{F}_{cubes}[\mathrm{U}]$.

Proof sketch: Cubes are conjunctions of literals: they induce unanimous profiles. If a function lifts all cubes then it is unanimous and viceversa. $\square$

Since $\mathcal{F}_{cubes}[\mathrm{U}] = \mathcal{F}[\mathrm{U}]$ this result can be interpreted as a characterisation of unanimity in terms of collective rationality with respect to cubes.

Results (examples)

Proposition

$\mathcal{CR}[\mathit{cubes}] = \mathcal{F}_{\mathit{cubes}}[\mathrm{U}]$.

Proof sketch: Cubes are conjunctions of literals: they induce unanimous profiles. If a function lifts all cubes then it is unanimous and viceversa. □

Since $\mathcal{F}_{\mathit{cubes}}[\mathrm{U}] = \mathcal{F}[\mathrm{U}]$ this result can be interpreted as a characterisation of unanimity in terms of collective rationality with respect to cubes.

Call $\mathcal{L}_{\not\leftrightarrow}$ the language of negative bi-implications (i.e. of the form $p_i \leftrightarrow \neg p_j$):

Proposition

$\mathcal{CR}[\mathcal{L}_{\not\leftrightarrow}] = \mathcal{F}_{\mathcal{L}_{\not\leftrightarrow}}[\mathrm{N}^{\mathcal{D}}]$.

For the axiom of independence a negative result holds:

Proposition

There is no language $\mathcal{L} \subseteq \mathcal{L}_{PS}$ such that $\mathcal{CR}[\mathcal{L}] = \mathcal{F}_{\mathcal{L}}[\mathrm{I}]$.

Conclusion and Future Work

In this work we have presented:

- a language to express rationality assumptions as integrity constraints IC over domains in binary aggregation;
- the concept of collective rationality of an aggregator wrt. a constraint IC;
- characterisation results for different propositional languages $\mathcal{L}$:
  Which properties of the aggregator guarantee that a certain IC is lifted.

Conclusion and Future Work

In this work we have presented:

- a language to express rationality assumptions as integrity constraints IC over domains in binary aggregation;
- the concept of collective rationality of an aggregator wrt. a constraint IC;
- characterisation results for different propositional languages $\mathcal{L}$:
  Which properties of the aggregator guarantee that a certain IC is lifted.

This work can be extended in a number of ways:

- using logic not only as a language but also as a tool to derive (im)possibility theorems for different set of axioms;
- extend the language for full combinatorial domains;
- characterise classical axioms in terms of collective rationality;
- study aggregation of more complex logical structures.