Voting Theory in the Lean Theorem Prover

Chase Norman
University of California, Berkeley

Joint with Wesley Holliday (University of California, Berkeley) and
Eric Pacuit (University of Maryland)

October 11, 2021

Social Choice Theory

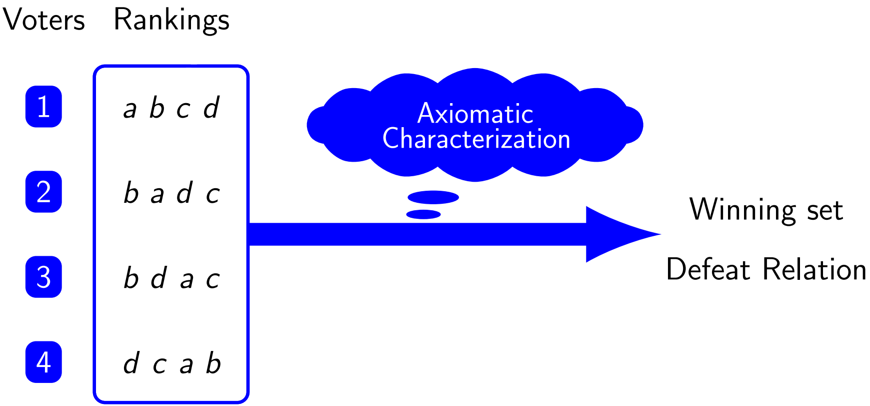
Voters	Rankings
1	a b c d
2	b a d c
3	b d a c
4	d c a b

Aggregation Method

Winning set

Defeat Relation

Social Choice Theory



social choice theory turns out to be perfectly suitable for mechanical theorem proving...

F. Wiedijk. *Arrow's impossibility theorem*. Formalized Mathematics, 15:171–174, 2007.

T. Nipkow. *Social choice theory in HOL: Arrow and Gibbard-Satterthwaite*. Journal of Automated Reasoning, 43:289–304, 2009.

M. Eberl. *Verifying Randomised Social Choice*. International Symposium on Frontiers of Combining Systems, FroCoS 2019: Frontiers of Combining Systems pp 240-256.

F. Brandt, M. Eberl, C. Saile and C. Stricker. *The Incompatibility of Fishburn-Strategyproofness and Pareto-Efficiency*. `https://www.isa-afp.org/entries/Fishburn_Impossibility.html`.

Lean

The Lean Theorem Prover aims to bridge the gap between interactive and automated theorem proving, by situating automated tools and methods in a framework that supports user interaction and the construction of fully specified axiomatic proofs. The goal is to support both mathematical reasoning and reasoning about complex systems, and to verify claims in both domains.

`https://leanprover.github.io/`

Profiles of Preferences

Profiles

Definition

For $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$, a (V, X)-*profile* is a map $\mathrm{P} : V \to \mathcal{B}(X)$.

Given a (V, X)-profile P, let $V(\mathrm{P})$ be V and $X(\mathrm{P})$ be X.

We then define a function Prof that assigns to each pair (V, X) of $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$ the set $\mathrm{Prof}(V, X)$ of all (V, X)-profiles.

Profiles

Definition

For $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$, a (V, X)-*profile* is a map $\mathrm{P} : V \to \mathcal{B}(X)$.

Given a (V, X)-profile P, let $V(\mathrm{P})$ be V and $X(\mathrm{P})$ be X.

We then define a function Prof that assigns to each pair (V, X) of $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$ the set $\mathrm{Prof}(V, X)$ of all (V, X)-profiles.

```
def Prof :  Type → Type → Type :=
λ (V X : Type), V → X → X → Prop
```

Majority Preferred

Definition

Given a profile P and $x, y \in X(\mathrm{P})$, we say that x *is majority preferred to* y *in* P if more voters rank x above y than rank y above x.

Majority Preferred

Definition

Given a profile P and $x, y \in X(\mathrm{P})$, we say that x *is majority preferred to* y *in* P if more voters rank x above y than rank y above x.

```
def majority_preferred {V X : Type} :
Prof V X → X → X → Prop := λ P x y,
cardinal.mk {v : V // P v x y} >
cardinal.mk {v : V // P v y x}
```

Margin

Definition

Given a profile P and $x, y \in X(\mathrm{P})$, the *margin of x over y in* P, denoted $Margin_{\mathrm{P}}(x, y)$, is $|\{i \in V(\mathrm{P}) \mid x\mathrm{P}_i y\}| - |\{i \in V(\mathrm{P}) \mid y\mathrm{P}_i x\}|$.

Margin

Definition

Given a profile P and $x, y \in X(\mathrm{P})$, the *margin of x over y in* P, denoted $Margin_{\mathrm{P}}(x, y)$, is $|\{i \in V(\mathrm{P}) \mid x\mathrm{P}_i y\}| - |\{i \in V(\mathrm{P}) \mid y\mathrm{P}_i x\}|$.

```
def margin {V X : Type} [fintype V] :
Prof V X → X → X → ℤ
:= λ P x y, ↑(finset.univ.filter (λ v, P v x y)).card -
↑(finset.univ.filter (λ v, P v y x)).card
```

Simple Example

Condorcet Winner and Majority Winner

Definition

Given a profile P and $x \in X(\mathrm{P})$, x is a *Condorcet winner in* P if for all $y \in X(\mathrm{P})$ with $y \neq x$, x is majority preferred to y in P.

We say that x is a *majority winner in* P if the number of voters who rank x (and only x) in first place is greater than the number of voters who do not rank x in first place.

Condorcet Winner and Majority Winner

Definition

Given a profile P and $x \in X(\mathrm{P})$, x is a *Condorcet winner in* P if for all $y \in X(\mathrm{P})$ with $y \neq x$, x is majority preferred to y in P.

We say that x is a *majority winner in* P if the number of voters who rank x (and only x) in first place is greater than the number of voters who do not rank x in first place.

```
def condorcet_winner {V X : Type} (P : Prof V X) (x :  X) :
Prop := ∀ y ≠ x, majority_preferred P x y
```

```
def majority_winner {V X : Type} (P : Prof V X) (x :  X) :
 Prop := cardinal.mk {v :  V // ∀ y ≠ x, P v x y} >
 cardinal.mk {v :  V // ∃ y ≠ x, P v y x}
```

Lemma. For any profile P, for all $x \in X(P)$, if x is a majority winner in P, then x is a Condorcet winner in P.

```
lemma condorcet_of_majority_winnner {V X : Type}
(P : Prof V X) [fintype V] (x :  X) :
majority_winner P x → condorcet_winner P x :=
```

We make use of the following theorem from mathlib:

```
theorem cardinal.mk_subtype_mono {α : Type u}
{φ ψ : α → Prop} (h : ∀ x, φ x → ψ x) :
cardinal.mk {x // φ x} ≤ cardinal.mk {x // ψ x}
```

```
lemma condorcet_of_majority_winnner {V X : Type}
(P : Prof V X) [fintype V] (x :  X) :
majority_winner P x → condorcet_winner P x :=
begin
1.   intros majority z z_ne_x,
2.   have imp1 :  ∀ v, (∀ y ≠ x, P v x y) → P v x z :=
     by finish,
3.   refine lt_of_lt_of_le _ (cardinal.mk_subtype_mono imp1),
4.   have imp2 :  ∀ v, P v z x → (∃ y ≠ x, P v y x) :=
     by finish,
5.   apply lt_of_le_of_lt (cardinal.mk_subtype_mono imp2),
6.   exact majority,
end
```

Functions on Profiles

Definition

For $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$, a *social choice correspondence for* (V, X), or (V, X)-SCC, is a function $F : \mathrm{Prof}(V, X) \to \wp(X)$.

Let SCC be a function that assigns to each pair (V, X) of $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$ the set of all (V, X)-SCCs.

Definition

For $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$, a *social choice correspondence for* (V, X), or (V, X)-SCC, is a function $F : \mathrm{Prof}(V, X) \to \wp(X)$.

Let SCC be a function that assigns to each pair (V, X) of $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$ the set of all (V, X)-SCCs.

```
def SCC := λ (V X : Type), Prof V X → set X
```

Definition

For $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$, a *social choice correspondence for* (V, X), or (V, X)-SCC, is a function $F : \mathrm{Prof}(V, X) \to \wp(X)$.

Let SCC be a function that assigns to each pair (V, X) of $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$ the set of all (V, X)-SCCs.

```
def SCC := λ (V X : Type), Prof V X → set X
```

```
def universal_domain_SCC {V X : Type} (F : SCC V X) : Prop :=
∀ P : Prof V X, F P ≠ ∅
```

Example

The Condorcet SCC:

```
def condorcet_SCC {V X : Type} : SCC V X := λ P,
{x : X | condorcet_winner P x ∨ ¬ ∃ y, condorcet_winner P y}
```

Variable-Election Framework

Definition

A *variable-election social choice correspondence* (VSCC) is a function F that assigns to each pair (V, X) of a $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$ a (V, X)-SCC.

```
def VSCC : Type 1 := Π (V X : Type), SCC V X
```

Variable-Election Framework

Definition

A *variable-election social choice correspondence* (VSCC) is a function F that assigns to each pair (V, X) of a $V \subseteq \mathcal{V}$ and $X \subseteq \mathcal{X}$ a (V, X)-SCC.

```
def VSCC : Type 1 := Π (V X : Type), SCC V X
```

Example: Condorcet VSCC

```
def condorcet_VSCC : VSCC := λ V X, condorcet_SCC
```

Other Functions on Profiles

```
def SCC := λ (V X : Type), Prof V X → set X

def VSCC : Type 1 := Π (V X : Type), SCC V X
```

Other Functions on Profiles

```
def SCC := λ (V X : Type), Prof V X → set X
def VSCC : Type 1 := Π (V X : Type), SCC V X

def CCR := λ (V X : Type), Prof V X → X → X → Prop
def VCCR := Π (V X : Type), CCR V X
```

Other Functions on Profiles

```
def SCC := λ (V X : Type), Prof V X → set X
def VSCC : Type 1 := Π (V X : Type), SCC V X
```

```
def CCR := λ (V X : Type), Prof V X → X → X → Prop
def VCCR := Π (V X : Type), CCR V X
```

Given an asymmetric VCCR f, we define the *maximal-element induced* VSCC f_M:

```
def max_el_VSCC : VCCR → VSCC := λ f V X P,
{x : X | ∀ y : X, ¬ f V X P y x}
```

Formalized Proofs

We verified all the results about a new voting method, *Split Cycle*, from

W. Holliday and E. Pacuit. *Split Cycle: A New Condorcet Consistent Voting Method Independent of Clones and Immune to Spoilers* . `https://arxiv.org/abs/2004.02350`.

	Split Cycle	Ranked Pairs	Beat Path	Mini-max	GETCHA/ GOCHA	Ranked Choice	Plurality
Condorcet Winner	✓	✓	✓	✓	✓	–	–
Condorcet Loser	✓	✓	✓	–	✓	✓	–
Pareto	✓	✓	✓	✓	–	✓	✓
Monotonicity	✓	✓	✓	✓	✓	–	✓
Independence of Clones	✓	✓	✓	–	✓	✓	–
Strong Stability for Winners	✓	–	–	–	✓	–	–
Reversal Symmetry	✓	✓	✓	–	✓	–	–
Positive Involvement	✓	–	–	✓	✓/–	✓	✓
Negative Involvement	✓	–	–	✓	✓/–	–	✓

Future Work

- Verify axioms of other voting methods: Not just margin-based methods (e.g., Split Cycle and Beat Path), but also scoring rules (e.g., Plurality and Borda), and *recursive* voting methods (e.g., Instant Runoff).

- Formalize characterization theorems (e.g., Arrow's Theorem characterizing dictatorship, May's Theorem characterizing majority rule, Young's Theorem characterizing scoring rules, ...).

Thank you!

`https://github.com/chasenorman/Formalized-Voting`