An Analysis of smart voting in liquid democracy

Giannis Tyrovolas

February 26, 2022

Contents

1 Preliminaries

1.1 Ballots

An election consists of a finite set of voters, voting on a finite set of issues. For each issue there is a finite set of alternatives. A special alternative is the abstention represented by $*$. Finally, there is an aggregation function that decides what the result of the election is. Formally:

Definition 1.1 (Election)**.** An *election* consists of a tuple $\langle N, I, D, \mathcal{G}\rangle$ where $N = \{1, ..., n\}$ is a finite non-empty set. The finite non-empty set I holds represents the issues of the election. For each $i \in I$, $D(i)$ is a finite non-empty set. The function $\mathcal{G}$ is an aggregation function that inputs the votes on every voter on every issue and outputs the resulting vote of the group on each issue.

For single-issue elections we omit the reference to I and abbreviate $D(i)$ to D. So, that function $\mathcal{G}$ is $\mathcal{G}\colon D^N \longrightarrow D$. We also focus on non-trivial issues where $|D| > 1$.

The model we will consider allows each voter to submit a smart ballot. A smart ballot is a preference list of smart votes. Each smart vote is a function with domain the other voters. A special requirement is that the final preference in the preference list is a direct vote on an alternative in D. Formally:

Definition 1.2 (Smart Ballots)**.** A smart ballot of an agent a on issue $i \in I$ is an ordering $((S^0, F^0) > \ldots > (S^{k-1}, F^{k-1}) > d)$ where $k \geq 0$. Each S^h for $h \leq k$ is a subset of N and $F^h\colon D(i)^{S^h} \longrightarrow D(i)$ is a resolute non-trivial aggregation function. We also have that $d \in D(i)$.

Further when relevant we will consider F^k to be the constant function with output d. Now, in most cases the sets S^h are implicit and we will drop any mention to them. That is supported by the fact that we will treat two functions F, G as identical if they are extensionally equal. Additionally we will disallow a voter to delegate to themselves. This is formalised by the following definition:

Definition 1.3 (Valid Smart Ballot)**.** A valid smart ballot of an agent a is a smart ballot B_a such that for all $0 \leq s < t \leq k$ F^s is not equivalent to F^t. Additionally $a \notin S_t$.

We collect the n smart ballots into a smart profile $\mathbf{B}$.

1.2 Unravellings

Now that we have defined each agents preferences we need to formalise how to make sense of these preferences. To do so we use *unravelling procedures* to determine each agent's vote.

Definition 1.4 (Unravelling Procedure)**.** An unravelling procedure is any computable function $\mathcal{U}$ where $\mathbf{B} \mapsto^{\mathcal{U}} \mathbf{d}$ with $\mathbf{d} \in D^n$.

Now, every agent would like to know which preference level was used to compute their vote. To do so we introduce the notion of a certificate:

Definition 1.5 (Certificate)**.** A certificate $\mathbf{c} \in \mathbb{N}^n$ for a profile $\mathbf{B}$ is a vector where for each $a \in N$ such that $B_a = (B_a^0 > \ldots > B_a^{k_a})$, the entry $\mathbf{c}_a \in [0, k_a]$ corresponds to the preference level for agent a.

Until now we have not actually used the agent's preferences. So, we need to introduce the concept of a consistent certificate. Consistent certificates are certificates where the vote of each agent is determined by the votes of other agents using the functions in their smart ballots. Formally:

Definition 1.6 (Consistent certificate)**.** For a profile $\mathbf{B}$, a certificate $\mathbf{c}$ is consistent if there is an ordering $\sigma\colon N \longrightarrow N$ of agents which starting from vector $X^0 = \{\Delta\}^n$ with placeholder values Δ for all agents, iteratively constructs an outcome vector of direct votes $X \in D^n$ as follows for $\sigma(a) = z \in [1, n]$:

$$X_a^z = F_a^{\mathbf{c}_a}(X^{z-1} \restriction_{S_a^{\mathbf{c}_a}})$$

where X_a represents agent a's entry in X and $X \restriction_S = \prod_{s \in S} X_s$.

Now what allows the computation of F on a partial input is the concept of a *necessary winner*. A necessary winner occurs when the available input of F is enough to decide the output of F. For example, $Maj(1, 1, a)$ will always resolve to 1 regardless of agent a's vote. So, in this case the necessary winner is 1.

Now, we are only interested in consistent certificates as these are the ones that respect the agents' votes. We will denote the set of consistent certificates of a profile $\mathbf{B}$ as $C(\mathbf{B})$

It is good that there is only one way to unravel consistent certificates. The following proposition is proven in .

Proposition 1.7. If a consistent certificate $\mathbf{c}$ can be given by two orderings σ and σ' of the agents, then the orderings yield the same outcome $X_{\mathbf{c}} \in D^n$.

So, that when we refer to consistent certificates we do not need to specify the votes of each agent. Now that we have settled the framework it's time to consider some "good" certificates. There is a very natural "cost" in this scenario and that is using a lot of the later preferences of each agent. So, we can set two very natural ways of minimising this cost. We can attempt to minimise the sum of the cost or in a more utilitarian setting attempt to minimise the maximum of the cost. Formally:

Definition 1.8 (MinSum)**.** For a given profile $\mathbf{B}$, the MinSum unravelling procedure is define as:

$$\textsc{MinSum}(\mathbf{B}) = \{X_{\mathbf{c}} \mid \arg\min_{\mathbf{c} \in C(\mathbf{B})} \sum_{i=1}^{n} c_i\}$$

Definition 1.9 (MinMax)**.** For a given profile $\mathbf{B}$, the MinMax unravelling procedure is define as:

$$\textsc{MinMax}(\mathbf{B}) = \{X_{\mathbf{c}} \mid \arg\min_{\mathbf{c} \in C(\mathbf{B})} \max(\mathbf{c})\}$$

2 Results

We introduce the axiom of cast participation. That is that voters that have a preference for an outcome benefit from voting directly for it. We denote that agent a prefers outcome d to e as $d >_a e$. We also infer that if agent a votes directly for outcome d then they weakly prefer d over any other outcome. Formally if $B_a = (d)$ then for all $d' \in D$, $d \geq_a d'$. Now, an issue is that irresolute unravelling procedures might return multiple outcomes. So MinSum can return outcomes for 0 and for 1. So, we need to consider the set of outcomes from irresolute unravelling procedures. The simplest ordering of these sets of outcomes is that a voter a that votes for d prefers every set of outcomes that includes d to every set of outcomes that doesn't. Further, $\{d\}$ is the preferred outcome of voter a. We assume that voters are indifferent between other outcomes.

Then we can formalise cast-participation for irresolute procedures:

Definition 2.1 (Cast-Participation for irresolute procedures)**.** A resolute voting rule r and a irresolute unravelling procedure $\mathcal{U}$ satisfy cast-participation if for all valid smart profiles $\mathbf{B}$ and agents $a \in N$ such that $B_a \in D \setminus \{*\}$ we have that for all $B'_a \neq B_a$:

$$r(\mathcal{U}(\mathbf{B})) \geq_a r(\mathcal{U}(\mathbf{B}_{-a}, B'_a))$$

Where $\mathbf{B}_{-a}, B'_a$ denotes replacing B_a with B'_a.

Cast participation seems like a desirable property of a voting system. Unfortunately in several cases it doesn't hold.

We now prove a series of negative results for cast participation.

Lemma 2.2. If r is not monotone then cast participation doesn't hold.

Proof. Suppose r is not monotone. Then for some vector $\mathbf{u} \in D^n$ with $\mathbf{u}_i = 0$ there exists a $\mathbf{u}' = (\mathbf{u}_1, \ldots, \mathbf{u}_{i-1}, 1, \mathbf{u}_{i+1}, \ldots, \mathbf{u}_n)$ with $r(\mathbf{u}) = 1$ and $r(\mathbf{u}') = 0$. This violates cast participation by the following profile: let $B_a = (\mathbf{u}_a)$, so in particular voter i prefers 0 to 1. Then voter i strictly prefers to vote for 1 than 0. So, to achieve cast participation, r must be monotone. □

Lemma 2.3. Let r be a rule such that for $n \geq 5$ voters if $n-2$ voters vote for d then r assigns d. Let $\mathcal{F}$ contain LIQUID. Then cast-participation doesn't hold for unravelling with MinMax.

Proof. Let $N = \{a, b, c, d, e\}$ and r as described. Then, let $B_a = (0), B_b = (a > 0), B_c = (1), B_d = B_e = (c > 0)$. Then MinMax would simply assign each individual to first preferences and the majority votes for 1. So the outcome set is $\{1\}$.

But, if $B_a = (b > 0)$ then a cycle is formed and so necessarily, MinMax will have to use some second preferences. Hence $(2, 1, 1, 2, 2)$ would be a valid preference level where a, b, d, e vote for 0 so r votes for 0. Note that $(2, 1, 1, 1, 1)$ is a valid preference level and would result to the majority again voting for 1. So the set of outcomes is $\{0, 1\}$ which is better than $\{1\}$ for agent a. □

We can further prove that MinSum doesn't hold when using three symbols. To do this set $D = \{0, 1, *\}$ where $*$ denote abstentions. In the case where abstentions are allowed we denote $Maj(0, *) = 0, Maj(1, *) = 1$ and $Maj(0, 1) = *$.

Lemma 2.4. Let $D = \{0, 1, *\}$ and r such that if a strict majority votes for outcome d then r supports outcome d. Let voters that directly vote for 0 have preference $0 > * > 1$. Then if voters are allowed to delegate to even majorities, cast-participation doesn't hold for MinSum.

Proof. Let $N = \{a, b, c, d, e\}$. Let $B_a = (0), B_b = (1), B_c = (Maj(a, b) > 0), B_d = B_e = (c > 0)$. Then we can resolve this smart profile by assigning everyone to first preferences. Then r delegates to $*$. So the set of outcomes is $\{*\}$.

Now if $B'_a = (c > d > 0)$ then the unique result of MinSum is c votes for 0. Further, a, d, e delegate to c using first preferences and vote for 0 as well. This results in a strict majority for 0. So the set of outcomes is $\{0\}$ which is a preferred set of outcomes. □

Note, that in the above case *Maj* is a monotone rule. So, it is not the case that monotone rules guarantee cast participation.

3 Proposals