Computational Social Choice: Spring 2017

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

Plan for Today

Today we are going to discuss a variety of questions that concern the roles of *information* and *communication* in voting:

- The Possible Winner Problem
- Compilation of Intermediate Election Results
- Communication Complexity of Voting Rules (just one slide)

Remark: We focus on positional scoring rules, particularly Borda and plurality, but several other voting rules have been analysed as well.

Remark: Boutilier and Rosenschein (2016) review several other strands of work that also fit under "information and communication".

C. Boutilier and J.S. Rosenschein. Incomplete Information and Communication in Voting. In F. Brandt et al. (eds.), *Handbook of COMSOC*. CUP, 2016.

Possible Winners

Idea: If we only have partial information about the ballots, we may ask which alternatives are *possible winners* (for a given voting rule F).

Let $\mathcal{P}(X)$ be the class of *partial orders* on the set of alternatives X.

The linear order $(\succ^\ell) \in \mathcal{L}(X)$ *refines* the partial order $(\succ^p) \in \mathcal{P}(X)$ if $(\succ^\ell) \supseteq (\succ^p)$, i.e., if $x \succ^\ell y$ whenever $x \succ^p y$. And a profile of linear ballots $\boldsymbol{R}^\ell \in \mathcal{L}(X)^n$ *refines* a profile of partial ballots $\boldsymbol{R}^p \in \mathcal{P}(X)^n$ if R_i^ℓ refines R_i^p for every voter $i \in N$.

Definition: Given a partial profile $\boldsymbol{R} \in \mathcal{P}(X)^n$, an alternative $x^\star \in X$ is called a *possible winner* under voting rule F if $x^\star \in F(\boldsymbol{R}^\star)$ for *some* profile of linear ballots $\boldsymbol{R}^\star \in \mathcal{L}(X)^n$ that refines $\boldsymbol{R}$.

The concept was originally introduced by Konczak and Lang (2005).

K. Konczak and J. Lang. *Voting Procedures with Incomplete Preferences*. Proc. Multidisciplinary Workshop on Advances in Preference Handling 2005.

Necessary Winners and Preference Elicitation

Given a profile of partial ballots $\boldsymbol{R} \in \mathcal{P}(X)^n$, an alternative $x^\star \in X$ is called a *necessary winner* under voting rule F if $x^\star \in F(\boldsymbol{R}^\star)$ for *all* profiles of linear ballots $\boldsymbol{R}^\star \in \mathcal{L}(X)^n$ that refine $\boldsymbol{R}$.

If we *elicit* preferences, either voter-by-voter ("coarse") or pair-by-pair ("fine"), we can stop once possible and necessary winners coincide.

Special Case: Missing Voters

A first natural special case of having only partial ballot information is when some *voters are missing*:

- Some voters have ballots that are *linear orders* (note: a linear order is a special case of a partial order).
- All other voters have *empty relations* as ballots: $x \not\succ y$ for all $x, y \in X$ (also a special case of a partial order)

Possible scenarios:

- We might be in the midst of a coarse elicitation procedure.
- Postal ballots may only arrive a few days after election day.

Exercise: *How do the possible/necessary winner problems relate to the coalitional manipulation problems we had discussed earlier?*

Special Case: Missing Alternatives

A second natural special case of having only partial ballot information is when some *alternatives are missing*:

- Distinguish "old" alternatives X_1 and "new" alternatives X_2.
- All ballots are complete on X_1: $x \succ y$ or $y \succ x$ for all $x, y \in X_1$.
- All ballots are empty on pairs involving a new alternative: $x \not\succ y$ if $x \in X_2$ or $y \in X_2$.

Possible scenario:

- Some alternatives (e.g., a new plan) become available only after voting has started.

The Possible Winner Problem

Let us now analyse the *computational complexity* of the following variant of the problem (for voting rule F):

$\textsc{PossibleWinner}(F)$
Instance: Profile of partial ballots $\boldsymbol{R} \in \mathcal{P}(X)^n$; alternative $x^\star \in X$.
Question: Is $x^\star$ a possible winner under voting rule F?

Note that ballots are *unweighted* and that the number of alternatives is *unbounded* (the crucial parameter for the complexity will be the number of alternatives, not the number of voters).

Possible Winners under Plurality

Even for the very simplest of voting rules, computing possible winners is not trivial. But for plurality it at least is polynomial:

Theorem 1 (Betzler and Dorn, 2010) *Under the plurality rule, the possible winner problem can be decided in polynomial time.*

The original proof of Betzler and Dorn is based on flow networks. Here we instead use a reduction to bipartite matching.

Remark: Computing possible winners under the *antiplurality rule* is also polynomial, and the proof is very similar.

N. Betzler and B. Dorn. Towards a Dichotomy for the Possible Winner Problem in Elections Based on Scoring Rules. *Journal of Computer and System Sciences*, 76(8):812–836, 2010.

Bipartite Matching

A *perfect matching* of a graph $G = (V, E)$ is a set of edges $E' \subseteq E$ such that each vertex in V is adjacent to *exactly* one edge in E'.

A graph $G = (V, E)$ is called *bipartite* if the set of vertices V can be partitioned into sets V_1 and V_2 such that each edge in E is adjacent to both a vertex in V_1 and a vertex in V_2.

BIPARTITE MATCHING
Instance: Bipartite graph G.
Question: Does G have a perfect matching?

BIPARTITE MATCHING can be decided in polynomial time, using for instance the Hopcroft-Karp Algorithm (Hopcroft and Karp, 1973).

J.E. Hopcroft and R.M. Karp. An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

Proof of Theorem 1

Recall: Given a profile of partial ballots, we want to show that we can decide in polynomial time whether $x^\star$ is a possible plurality winner.

An alternative can get a point from a partial ballot iff it is undominated.

Suppose $x^\star$ is undominated in $K \leqslant n$ ballots. So $x^\star$ can get K points.

Can we choose one undominated alternative for each of the other ballots without one alternative getting more than K points?

If $K \cdot (m-1) < n-K$, then one rival must get more points than $x^\star$. ✓

Otherwise, construct a bipartite graph G:

- Lefthand vertices: one for each of the $n-K$ "other" ballots (= sets of undominated alternatives) and $K \cdot (m-1) - (n-K)$ "dummy" vertices
- Righthand vertices: K copies of each alternative other than $x^\star$
- Edges: link each set-vertex to each copy of each of its members; and link each dummy vertex to all vertices on the right

Now we have a one-to-one correspondence between perfect matchings of G and ballot refinements that give each rival of $x^\star$ at most K points. ✓

Possible Winners under Borda

Theorem 2 (Xia and Conitzer, 2008) *Under the Borda rule, deciding whether an alternative is a possible winner is NP-complete.*

Proof: By reduction from EXACT3COVER (details omitted).

In fact, the result of Xia and Conitzer covers many positional scoring rules. Betzler and Dorn (2010) showed that POSSIBLEWINNER is NP-complete for almost all PSR's (except for plurality and antiplurality).

L. Xia and V. Conitzer. Determining Possible and Necessary Winners under Common Voting Rules Given Partial Orders. Proc. AAAI-2008.

N. Betzler and B. Dorn. Towards a Dichotomy for the Possible Winner Problem in Elections Based on Scoring Rules. *Journal of Computer and System Sciences*, 76(8):812–836, 2010.

Overview: Possible Winners under PSR's

We have seen results regarding the computational complexity of the PossibleWinner problem for *positional scoring rules:*

- General case: polynomial for plurality and antiplurality
- General case: NP-complete for Borda (and most PSR's)
- Missing voters only: NP-complete for Borda (not discussed)

  This is equivalent to constructive coalitional manipulation. For *weighted* voters, we have seen an NP-hardness proof in the lecture on coping with manipulation (it also holds for unweighted voters).
- Missing alternatives only: polynomial for Borda (not discussed)

  Easy proof: best chance of old alternative $x^\star$ to win if all new alternatives are inserted directly below $x^\star$ in everyone's ballot.

Compilation of Intermediate Election Results

Given a partial ballot profile and a voting rule, how much information do we need to store to be to be able to compute the election winners once the remaining ballot information comes in?

An instance of this general question:

- If some voters have voted already (using complete linear ballots), what is the most compact way of *compiling* the intermediate election result so as to be able to compute the winner(s) once the remaining voters have voted?

The number of bits required to store the relevant information for a given voting rule F is called the *compilation complexity* of F.

Remark: The *number of additional voters* may be known or unknown. We shall discuss the case where it is *unknown*.

Compilation Complexity

Let n be the number of (old) voters and m the number of alternatives.
We need $\lceil \log m \rceil$ bits to represent the name of one alternative.

Some basic observations:

- The compilation complexity of *any voting rule* is at most $n\lceil \log(m!) \rceil$ (just store all ballots)
- The compilation complexity of any *anonymous* voting rule is at most $\min\{n\lceil \log(m!) \rceil, m!\lceil \log(n+1) \rceil\}$ (as we can also store for each ballot the number of voters choosing it, if that's shorter)
- The compilation complexity of any *dictatorial* voting rule is $\lceil \log m \rceil$ (just store the choice of the dictator)
- The compilation complexity of any *constant* voting rule (always electing the same winner) is 0.

Y. Chevaleyre, J. Lang, N. Maudet, and G. Ravailly-Abadie. Compiling the Votes of a Subelectorate. Proc. IJCAI-2009.

Equivalence Classes

Let F be a voting rule.

Call two ballot profiles $\boldsymbol{R}$ and $\boldsymbol{R'}$ *F-equivalent* if for any additional ballot profile $\boldsymbol{R}^\star$, we have $F(\boldsymbol{R} \oplus \boldsymbol{R}^\star) = F(\boldsymbol{R'} \oplus \boldsymbol{R}^\star)$.

The most space-efficient (but not time-efficient!, which is ok) way to compile a ballot profile is to store the index of its equivalence class.

Hence, if $g(F, n, m)$ is the number of equivalence classes for voting rule F, n (old) voters, and m alternatives, then the compilation complexity of F is (exactly) $\lceil \log g(F, n, m) \rceil$.

This suggests a proof technique for proving compilation complexity results: count the equivalence classes!

Compilation Complexity of Plurality

The compilation complexity of *plurality* will be at most $m\lceil\log(n+1)\rceil$ (store the plurality score $\in \{0,\ldots,n\}$ for each alternative).

But we can do better:

Theorem 3 (Chevaleyre et al., 2009) *The compilation complexity of the plurality rule (for an unknown number of additional voters) is exactly* $\lceil\log\binom{m+n-1}{n}\rceil$, *which is in* $\Theta(m\log(1+\frac{n}{m})+n\log(1+\frac{m}{n}))$.

Recall that $f(n)\in\Theta(g(n))$ means that f is asymptotically bounded both from above and from below by g.

The proof for the exact bound is on the next slide. The rest follows from *Stirling's approximation* [$\log(n!)$ is in $\Theta(n\log n)$].

Y. Chevaleyre, J. Lang, N. Maudet, and G. Ravailly-Abadie. Compiling the Votes of a Subelectorate. Proc. IJCAI-2009.

Proof

Profiles $\boldsymbol{R}$ and $\boldsymbol{R'}$ are plurality-equivalent iff for each alternative x the number of times x occurs in the the top position in $\boldsymbol{R}$ is equal to the number of times x occurs in the the top position in $\boldsymbol{R'}$.

There are n (= number of voters) top positions available.

How many ways are there to award top positions to m alternatives, so that the sum of numbers of top positions adds up to n?

That is, how many solutions are there for this equation?

$$x_1 + x_2 + \cdots + x_m \quad = \quad n \quad \text{with } x_i \in \{0, 1, 2, \ldots\}$$

This is equal to the number of ways of picking n balls from an urn with m balls *with replacement*, which in turn is equal to picking n balls from an urn with $m + n - 1$ balls *without replacement*: $\binom{m+n-1}{n}$.

The logarithm of the above is the compilation complexity. ✓

Compilation Complexity of Borda

Two ballots are *Borda-equivalent* if they assign the same Borda scores. But this doesn't tell us how many equivalence classes there are (some combinations of Borda scores will be impossible). So we cannot use this to get the *exact* compilation complexity of Borda.

Theorem 4 (Chevaleyre et al., 2009) *The compilation complexity of Borda (for an unknown num. of additional voters) is* $\Theta(m \log(nm))$.

Proof of the lower bound omitted (counting equivalence classes).

Proof of the upper bound: Borda score of each alternative is at most $n(m-1)$, so need to store at most $O(\log(nm))$ bits per alternative. ✓

Y. Chevaleyre, J. Lang, N. Maudet, and G. Ravailly-Abadie. Compiling the Votes of a Subelectorate. Proc. IJCAI-2009.

Communication Complexity

Yet another interesting question: How much information do voters need to transmit to allow us to compute the winner of an election?

Let n be the number of voters and m the number of alternatives. We need $\lceil \log m \rceil$ bits to communicate the identity of an alternative. Upper bounds are easy to derive:

- Plurality: $O(n \log m)$ — each voter sends one alternative name
- Approval: $O(nm)$ — let each voter communicate a bit-string of length m to encode their approved subset of alternatives
- Borda (any rule with linear orders as ballots): $O(n\, m \log m)$ — each voter sends m alternative names in turn

Conitzer and Sandholm (2005) show that many of these bounds are tight, using tools from *communication complexity*.

V. Conitzer and T. Sandholm. Communication Complexity of Common Voting Rules. Proc. EC-2005.

Summary

We have discussed several situations where only *partial information* regarding the ballots is available. Examples:

- Some voters arrive late.
- Some alternatives enter the election late.
- Elicitation proceeds in stages (ballots are partial orders).

This gives rise to a number of questions:

- *Possible winners:* which alternatives have a chance of winning?
- *Necessary winners:* will certain alternatives surely win?
- *Compilation:* how can we efficiently represent the relevant information elicited so far?

What next? Iterative voting. Then voting in combinatorial domains.