Kingmaker: A Simulation Analysis of Strategic Voting

Chance Addis

May 2025

A thesis presented to
The Division of Mathematical and Natural Sciences
Reed College

Submitted in partial fulfillment of the requirements
for the degree Bachelor of Arts

Approved for the Division
(Mathematics - Statistics)

_____
Michael Pearce

Contents

Abstract

In the field of social choice theory, The Gibbard-Satterthwaite Theorem tells us that any all non-trivial, non-dictatorial voting methods do *not* have a dominant strategy for any individual voter. Thus every voting system which follows these axioms (which they should) must be manipulable. Thus any voting method must consider how voters will strategically vote in order to benefit their social welfare.

There have been efforts such as ... that aim to measure how resilient a voting method is to certain kinds of strategic voting, but ...

My thesis expands upon this literature by simulating complex social conditions in order to:

1. Synthesize optimal (in some measurable sense) strategies for some voter block,
2. Use those novel (as well as known) voting strategies to compare the resilience of common voting methods, and
3. ...

Dedication

To my parents, for their ceaseless support.

1. Introduction

2. Background

2.1. A History of Social Choice

Social choice has been applied since humanity has existed. After all there has always been a need to make collective decisions. Its theory, however, is more modern, and tracks changes in our values and methodologies around collective action and voting.

2.1.1. Precursors

Early social choice theory was far more concerned with voting as a procedure than of social welfare as an ideal.

One such precursor, Pliny the Younger—a Roman magistrate—wrote a letter to Titius Aristo on June 24, 105 describing the a criminal trail which he presided over. Under the traditional senate procedures, there would first be a vote of innocence / guilt, and then if guilty, or punishment, to which exile or execution were proposed. Of the three proposals, aquittal, exile, or execution, aquittal held the plurality of supporters, but not the majority, although exile would have won in a direct two-way vote against either aquittal or execution. Pliny expected that guilt would be asserted, and then execution. Being himself in favor or exile, he proposed a novel three-way election. This had the disired effect; supporters of execution withdrew their proposal (since aquittal would win if they split the vote for exile), and the vote defaulted to a two-way election between aquittal and exile, which exile carried.

Here we find one of the earliest recorded instances of voting manipulation. Pliny deliberately changed the voting rules so that execution would have split the vote between exile and execution, leading to acquittal. In response the voters changed strategies and voted for the second best option.

Other notable precursors include Catalan Ramon Lull (1232–1316) and the German Nikolaus von Kues (1401–1464). Ramon Lull produced, among other

things, what is now known as Copeland (or Lull's) method, in which the winner is the candidate with the most pairwise wins against other candidates via ranked ballots. Von Kues is known for proposing what is now known as Borda's method, a method that we will discuss shortly.

2.1.2. The Math of Consensus

Jean-Charles de Borda (1733–1799) and Nicolas de Condorcet (1743–1794) are considered the modern founders of social choice theory. They are distinguished from their precursors by the mathematical nature of their analysis.

Of the two, Borda's contributions are the more limited, having only written few pages. In that space he outlined the Borda count, a Condorcet method—which we'll discuss in just a moment—where the each candidate is assigned a score defined as the aggregate points assigned by their placement on each ballot. (With k options, rank 1 gets $k-1$ points, rank 2 gets $k-2$, etc.) The candidates with the most points wins. In the same article, he also shows that the plurality method is flawed, given that it can select a pairwise loser to all other candidates (i.e. a Condorcet loser).

Condorcet alternatively devotes an entire books—along with several papers —worth of material on voting in his *Essay on the Application of Analysis to the Probability of Majority Decisions* [translated from the native French]. In his work, Condorcet establishes many notable results, including Condorcet's jury theorem and Condorcet's paradox.

Condorcet's jury theorem proves that if each jury member is more likely than not to make a correct verdict, then the probability that the majority verdict is correct increases as the number of jury members increases.

Condorcet's paradox, a fundamental property of majority rule, states that with more than 2 candidates plurality methods can become intransitive (*not* transitive) —it's possible for A to pairwise beat B, B to pairwise beat C, *and* C to pairwise beat A.

Note Charles Dodgson (Lewis Carroll) here

2.1.3. Welfare Economics

Thus far, we've seen discussion about particular voting systems and their properties, such as *Condorcet* and *Borda*, but we have yet to introduce a general approach

to the study of preference aggregation and social choice. This is where *Kenneth Arrow* comes into the picture. It's Kenneth Arrow's nobel prize-worthy contributions that first formalized such a approach. Specifically, to studies a sub-class of preference aggregation methods that he called *social welfare functions*. He then proceeded to use this framework to prove the famous *Arrow's impossibility theorem* [1] (1963), which states that for more than 2 voters voting between 3 or more possible alternatives, there does not exist a social welfare function $\mathcal{W}$ that satisfies a set of reasonable axioms, discussed below.

This theorem sparked much debate in social choice. As the *Stanford Encyclopedia of Philosophy* notes, "William Riker (1920–1993), who inspired the Rochester school in political science, interpreted it as a mathematical proof of the impossibility of populist democracy (e.g., Riker 1982). Others, most prominently Amartya Sen (born 1933), who won the 1998 Nobel Memorial Prize, took it to show that ordinal preferences are insufficient for making satisfactory social choices and that social decisions require a richer informational basis. Commentators also questioned whether Arrow's desiderata on an aggregation method are as innocuous as claimed or whether they should be relaxed."

But while Arrow's impossibility theorem might be *the* social choice theorem, it's a different but no less important social choice theorem that will be the focus of our research today. That theorem being the *Gibbard-Satterthwaite theorem* [2]. The Gibbard-Satterthwaite theorem was first conjectured by philosopher Michael Dummett and the mathematician Robin Farquharson in 1961 and then proved independently by philosopher Allan Gibbard in 1973 and economist Mark Satterthwaite in 1975. It states, broadly, that for any non-dictatorial, non-trivial voting system, then either (1)

2.2. A Formalization of Social Choice

At its core, social choice theory is concerned with the analysis of *preference aggregation*, understood to be the aggregation of individual preferences, each of which compares two or more social alternatives, into a single collective preference (or choice). The basic framework, which is still standard, was introduced by Kenneth Arrow in 1951.

2.2.1. Basic Framework

Let $N = \{1, 2, ...\}$ be a set of n individuals ($n \geq 2$), and $A = \{\pi_1, \pi_2, ...\}$ be a set of m social alternatives, such as candidates, policies, goods, etc. Each individual v_i has a *preference ordering* P_i over these alternatives. A *preference ordering* is defined by a complete, total order on X known as a *weak preference*. It is written with the symbol $\preceq, \succeq$, where $\pi_1 \preceq \pi_2$ is defined as π_1 is preferred or indifferent to π_2. There are also shorthands for *strict preference* ($\pi_1 \prec \pi_2 := \pi_1 \preceq \pi_2 \wedge \pi_2 \npreceq \pi_1$) and *strict indifference* ($\pi_1 \sim \pi_2 := \pi_1 \preceq \pi_2 \wedge \pi_2 \preceq \pi_1$)
[1]

A collection of preference orderings across a set of individuals $\{P_1, P_2, ..., P_n\rangle\}$, is called a *profile*. A *social welfare function* is a function $W : P \to A$ ...

2.2.2. Classical Social Choice Rules

...

2.3. A Statistics Of Social Choice

2.3.1. Preference Realization

So far, the only conception of voting has been with *preferences*. But do voters actually submit their ballots deterministically? No. It's unrealistic to assume that voters always vote rationally in a predefined way. There is an element of randomness in the voting process. Say that a voter has the following opinions: $\pi_1 = 65\%$ approval and $\pi_2 = 35\%$ approval. The voter may submit a ballot with $\pi_1 \prec \pi_2$ or $\pi_2 \prec \pi_1$ depending on how they feel on the day of the election. Voting is a *stochastic process*.

This conceptualization facilitates the need to disambiguate a *preference* from a *ballot*.[2] A preference is redefined as a distribution over preference orderings, and a ballot is a realization of that distribution. Think of a preference like a superposition, and when the election is held, the preference collapses into a ballot.

---

[1]Here I forgo the more general formalism of *preference aggregation rule* in favor of the more specific case of *social welfare functions*. The term *social welfare function* is a specific type of preference aggregation rule that *always* produces a complete social ranking of alternatives. For the scope of this thesis, social welfare functions are more suitable.

[2]Here I redefine preference to a new definition and define ballot in its place.

Stochastic voting is a concept hardwired into the framework of this thesis, as all the methods for generating ballots are stochastic. This framing allows us to conceptualize these *synthesizers* as voters who non-deterministically submit their ballots via some set of rules. Granted those rules might be "randomly select an ordering of candidates", but it's still of the same process as real voting.

2.3.2. Statistical Social Choice Rules

2.4. Strategic Voting

...

3. Methods

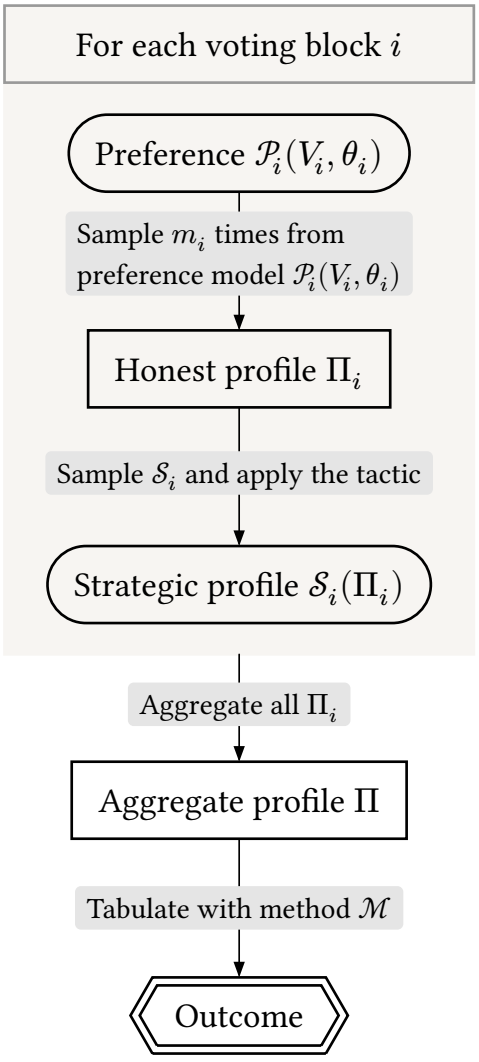

Fig 1: Overview of the election pipeline

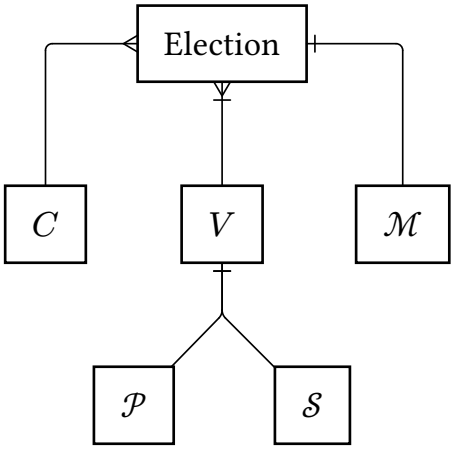

Fig 2: Structure of an election configuration

In kingmaker, ballot generation, aggregation, and tabulation has a well-defined behavior, which we will explore in-depth shortly. The above diagrams give a high-level overview of the pipeline and state. As you can see, ballots begin as

preferences, are realized into ballots, are converted *independently* into strategic ballots, and then aggregated into a profile. This profile is then tabulated via the voting method, and a winner is determined. Its important to understand that this process is *not* necessarily realistic and has limitations that will be discussed in detail.

3.1. Candidates

Before the election can begin, candidates must be registered. This is done by creating a `Candidate` struct. A candidate is as follows:

```
pub struct Candidate {
    id: Id,
    name: String,
    party: Option<String>,
    positions: Option<Vec<NotNan<f32>>>,
}
```

You might be confused about the addition of the `positions` field to the `Candidate` struct. This is ignored in most cases, but we'll talk about when and where it will be used later on.

3.2. Preferences & Realization

Preferences within kingmaker correspond roughly with preferences as described in theory, but mean something distinct in practice. Preferences are defined in the following manner:

```
pub trait Preference<B: Ballot>: Send + Sync + Debug {
    fn draw(&self, candidate_pool: &CandidatePool, rng: &mut
StdRng) -> B;
    fn sample(
        &self,
        candidate_pool: &[Candidate],
        sample_size: usize,
        rng: &mut StdRng,
```

```
    ) → Vec<B> {
        (0..sample_size)
            .map(|_| self.draw(candidate_pool, rng))
            .collect()
    }
}
```

On its surface this is precisely equal to its theoretical counterpart: bound by the ballot type and set of candidates, preference is some distribution that can be drawn from to produce a ballot. However note the `sample` method. This hints to the larger distinction.

The difference is that preferences are not defined on a voter by voter basis, but instead are defined for a block of voters (e.g. Democrats, Independents, etc). This means that when different voters draw from the preference, they are drawing from the same underlying distribution. Thus we should model preferences as aggregate preferences of all voters in the voting block, and that each draw is that voters individual preferences being defined. Although this notion is not entirely accurate, since grouping all voter preferences together loses information. However it is sufficient for our purposes.

We are primarily interested in well-known and well-understood preferences, so `kingmaker` implements a number of well-known preferences. These include:

- `Impartial`: Select a preference uniformly at random.
- `Manual`: Select a preference based on user input.
- `PlackettLuce`: Select a preference based on a Plackett-Luce distribution.
- `Mallows`: Select a preference based on a Mallows distribution.

3.3. Tactics & Strategies

Tactics in kingmaker correspond exactly with tactics as defines by theory, as deterministic processes that voters can engage with to increase their social welfare. Tactics are defined in the following manner:

```rust
pub trait Tactic<B: Ballot>: Send + Sync + Debug {
    fn apply(&self, ballot: B) -> B;
}
```

Strategies, on the other hand, are a statistical extension that allows for stochastic strategic thinking. Strategies are defined as distributions over tactics, and are defined:

```rust
pub struct Strategy<B: Ballot> {
    tactics: Vec<(Arc<dyn Tactic<B>>, f32)>,
}
```

where `Arc<dyn Tactic<B>>` is a `Tactic` and `f32` is the likelihood of the voter choosing that tactic.

Again, `kingmaker` implements a number of well-known tactics. These include:

- `Identity`: Default to the voter's original preference.
- `Random`: Select a re-ordering uniformly at random.
- `Compromise`: Place candidates with a higher likelihood of being elected first, over more preferred candidates.
- `Burial`: Place candidates with a higher likelihood of being elected last, over less preferred candidates.
- `Pushover`: Place pushover candidates higher in the ranking, not to increase the likelihood of winning, but to eliminate stronger more preferred candidates early in tabulation rounds to then be defeated later by more preferred candidates.

3.4. Voters & Voting Blocks

A voting block is defined:

```rust
pub struct VotingBlock<B: Ballot> {
    members: usize,
    preference: Arc<dyn Preference<B>>,
    strategy: Strategy<B>,
}
```

which is to say, a group of voters who share a common preference and strategy for voting.

Voting blocks are centrally important to understanding the trade-offs and limitations of `kingmaker`. They allow us to model the behavior of voters and their preferences, but only in certain ways. Like we mentioned earlier, preference aggregation and the difference between group and individual preferences exist because of voting blocks and constructed to have one aggregate preference. The same aggregation applies to strategy, where we aggregate the strategies of each voter in the block to determine the overall strategy of the block.

3.5. Methods & Outcomes

With the profile of ballots in hand, the final step is to tabulate the results and determine the outcome. For this we use whichever `Method` we defined when configuring the election. An election is defined:

```
pub trait Method: Send + Sync {
    type Ballot: Ballot;
    type Winner: Outcome;
    fn outcome(
        &self,
        candidate_pool: &[Candidate],
        profile: &Profile<Self::Ballot>,
    ) → Self::Winner;
}
```

and functions precisely as the theory would dictate. It takes a profile and candidate pool and determines the winner(s). Note that the outcomes could be a `SingleWinner` or a `MultiWinner` depending on whether the office is single-member or multi-member.

`kingmaker` implements a number of well-known methods, as well as a few novel ones. These include:

- `RandomDictator`: Select the winner by randomly selecting a voter and then selecting their top choice.
- `Plurality` / `First Past the Post`: Select a winner by selecting the candidate with the most first-place votes.
- `Approval`: Select a winner by selecting the candidate with the most approval votes.

- `Borda`: For a given ballot, the first place candidate receives `n` points, the second place candidate receives `n-1` points, and so on, where `n` is the number of candidates. The candidate with the highest total score wins.
- `InstantRunoff`: Select a winner by eliminating the candidate with the fewest first-place votes until a candidate has a majority of first-place votes.
- `SingleTransferableVote`: Select n candidates by eliminating the candidate with the fewest first-place, transferring votes from candidates who get elected, and redistributing votes until n candidates have a majority of first-place votes.
- `Star`: ...

3.6. Elections

```
pub struct Election<B, C, M>
where
    B: Ballot,
    C: Send + Sync,
    M: Method<Ballot = B>,
{
    conditions: C,
    candidate_pool: Vec<Candidate>,
    voter_pool: Vec<VotingBlock<B>>,
    method: M,
}
```

The `Election` simply aggregates the necessary information `conditions`, `candidate_pool`, `voter_pool`, and `method`, and provides a convenient interface for running elections. There are two methods for running elections: `election.run_once(seed: u64)` and `election.run_many(times: usize, seed: u64)`. There are also useful helper functions for displaying election results.

4. Results

5. Discussion

A Symbols and Definitions

Symbol	**Definition**
$C = \{c_i \mid i \in \{1..n\}\}$	A set of candidates
$V = \{v_i \mid i \in \{1..n\}\}$	A set of voters
$\Pi = \{\pi_i \mid i \in \{1..n\}\}$	A set of ballots (a profile)
$\prec, \succ, \not\prec, \not\succ$ $\preceq, \succeq, \not\preceq, \not\succeq$	$\pi_i \prec \pi_j \Rightarrow \pi_i$ is preferred to π_j, $\pi_i \preceq \pi_i \Rightarrow \pi_i$ is preferred or indifferent to π_j
$\mathcal{W}(v_i)$	The social welfare function for voter i
$\mathcal{P}(v_i, \theta)$	The preferences of voter i given hyperparameters θ
$\mathcal{S}(v_i)$	The strategy for voter i
$\mathcal{M}(\Pi)$	The outcome of a method M on a set of ballots Π.

Table 1: Symbols and Definitions

B Code Appendix

```
use kingmaker::prelude::*;

fn main() {
    // configure election(s)
    let candidate_pool = vec![
        Candidate::new(0, "A", Some("DEM"), None),
        Candidate::new(1, "A", Some("REP"), None),
        Candidate::new(2, "C", Some("GREEN"), None),
        Candidate::new(3, "D", None, None),
        Candidate::new(4, "E", None, None),
    ];
    let voter_pool = [
        VotingBlock::builder(
            preferences::Mallows::new(vec![0, 1, 2, 3, 4], 0.2),
            5_000,
        )
        .add_tactic(tactics::Identity, 0.8)
        .add_tactic(tactics::Burial(vec![1]), 0.2)
        .build(),
        VotingBlock::builder(
            preferences::Mallows::new(vec![2, 1, 4, 3, 0], 0.15),
            5_000,
        )
        .add_tactic(tactics::Identity, 0.7)
        .add_tactic(tactics::Burial(vec![1]), 0.3)
        .build(),
    ];
    let election =
        Election::new((), candidate_pool, voter_pool,
methods::Plurality);
    // run election(s)
    let outcomes = election.run_once(0);
```

```
    // display outcome
    election.display([outcomes]);
}
```

And the code is available at *Approximately-Equal/Kingmaker*

References

[1] K. J. Arrow, "A Difficulty in the Concept of Social Welfare," *The Journal of political economy*, vol. 58, no. 4, pp. 328–346, 1950.

[2] A. Gibbard, "Manipulation of Voting Schemes: A General Result," *Econometrica*, vol. 41, no. 4, pp. 587–601, 1973.

[3] R. D. (. D. Luce, "Individual choice behavior; a theoretical analysis.." Wiley, 1959.

[4] R. L. Plackett, "The Analysis of Permutations," *Journal of the Royal Statistical Society. Series C (AppliedStatistics)*, vol. 24, no. 2, pp. 193–202, 1975.

[5] J. G. Kemeny, "Mathematics without Numbers," *Daedalus (Cambridge, Mass.)*, vol. 88, no. 4, pp. 577–591, 1959.

[6] W. G. Ludwin, "Strategic Voting and the Borda Method," *Public Choice*, vol. 33, no. 1, pp. 85–90, 1978.

[7] C. L. Mallows, "Non-Null Ranking Models. I," *Biometrika*, vol. 44, no. 1/2, p. 114–, 1957.

[8] Hammond and Thomas H., "Rank Injustice?: How the Scoring Method for Cross-Country Running Competitions Violates Major Social Choice Principles," *Public choice*, vol. 133, no. 3/4, pp. 359–375, 2007.

[9] Niclas Boehmer *et al.*, "Guide to Numerical Experiments on Elections in Computational Social Choice," 2024, [Online]. Available: https://arxiv.org/abs/2402.11765

[10] C. List, "Social Choice Theory," *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, 2022.

[11] FairVote, "PR Library: Types of Voting Systems."