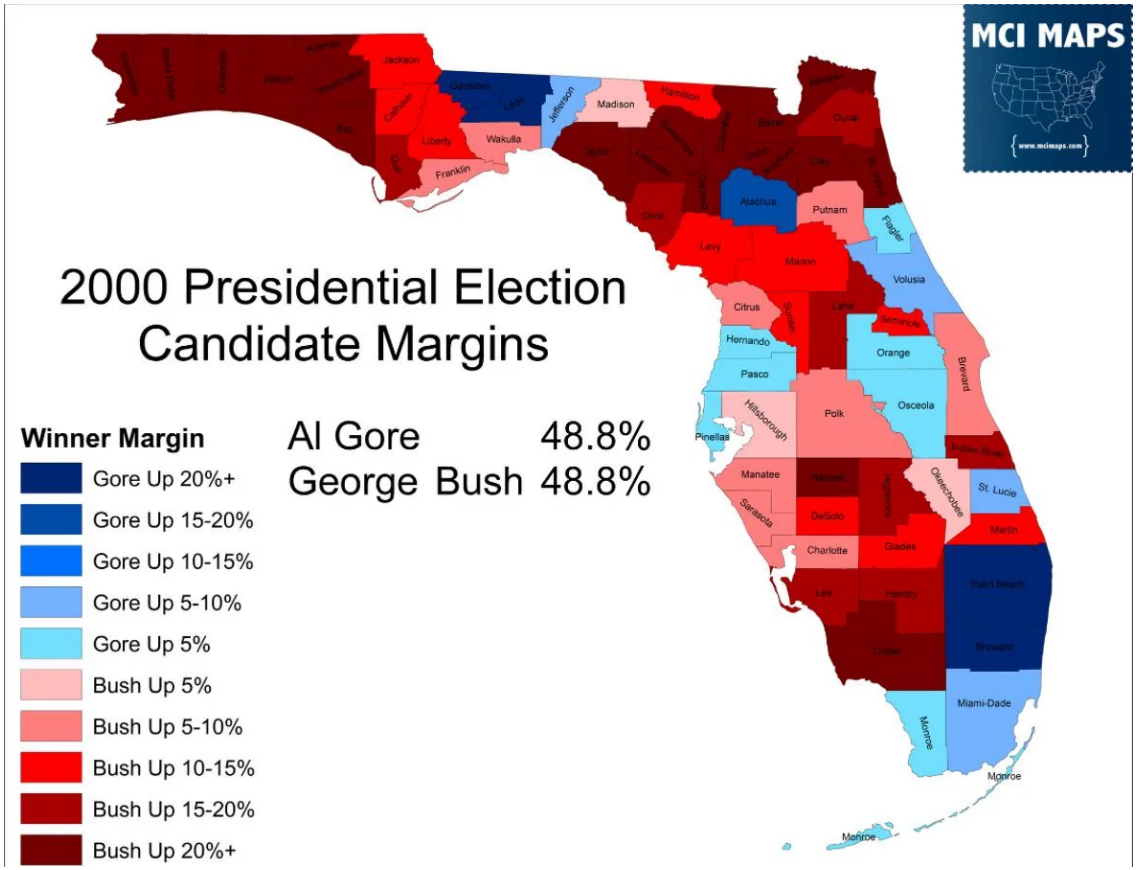
# Why Rigging an Election is Too Hard

## The Surprising Mathematics of Democracy

Arjun Maneesh Agarwal

BSc II, Chennai Mathematical Institute

# November 7, 2000



# Post Game Analysis

- Florida. Bush vs Gore. 537 votes separate them.
- Ralph Nader received 97,421 votes
- Exit polls: Most Nader voters preferred Gore to Bush
- **Question:** Did Nader *spoil* the election?

# Why Should We Care?

*All governments suffer a recurring problem: Power attracts pathological personalities. It is not that power corrupts but that it is magnetic to the corruptible. — Frank Herbert*

**Today's Journey:**

1. Can we design a perfect voting system? (Spoiler: No)
2. Can computational complexity save democracy?
3. What does this mean for real elections?

# What is Computational Social Choice?

Social Science is the study of societies and the relationships among members within those societies. We often have anthropology, archaeology, economics, geography, history, linguistics, management, communication studies, psychology, culturology, and **political science.**

Decision theory or Choice Theory, at the heart of it, is the math of modeling real-life scenarios and optimizing decision-making. It deals with financial markets and elections, fairness and moral hazard, auctions and corruption; everywhere a decision is to be made, decision theory enters.

Computer Science is, at the heart of it, concerned with which problems can we solve and how fast can we solve them.


The image under this function is referred to as **"Computational Social Science"** or **"Computational Social Choice"** aka **COMSOC**.

# What is Voting Theory?

- Treating voting rules as an algorithm
- While this may seem like an obvious idea, it was quite a revolutionary idea 30 years back.
- Today, it's still young enough that breakthroughs happen *weekly*.

# Act I

The Impossibility of Perfection

# What We Want From Voting

**The Basic Problem:** $n$ candidates, $v$ voters with preferences. Choose a winner fairly.

**What seems reasonable?**

- **Anonymous**: All voters count equally (swapping voters doesn't change outcome)
- **Neutral**: All candidates treated equally (swapping all preferences swaps outcome)
- **Monotone**: Getting more support can't hurt you
- **Unanimous**: If everyone prefers A to B, then B doesn't win

These seem *obviously* desirable, right?

# One More Reasonable Property

## Independence of Irrelevant Alternatives (IIA)

The choice between candidates A and B should depend **only** on voters' preferences between A and B.

*Translation:* Adding or removing candidate C shouldn't change whether A or B wins.

This rules out the "Nader spoiler" effect!

Surely we can design a system with all these properties...

# Arrow's Impossibility Theorem

**Theorem (Kenneth Arrow, 1951)**

The **only** voting rule satisfying Unanimity and IIA is **Dictatorship**.

**Perfect voting is mathematically impossible.**

Let that sink in. We can't have all the nice properties. Something has to give.

*(Arrow won the Nobel Prize for proving this in 1972)*

# Example: Why IIA Fails

**Scenario:** 3 voters, 3 candidates (A, B, C)

**Round 1:**

- Voter 1: $A > B > C$
- Voter 2: $B > C > A$
- Voter 3: $C > A > B$

**Plurality winner:** Tie! Say A wins.

**Round 2:** C drops out

- Voter 1: $A > B$
- Voter 2: $B > A$
- Voter 3: $A > B$

**Plurality winner:** A wins 2-1

**Wait...** Removing C (the "irrelevant" candidate) changed the result from tie to clear winner!
IIA is **hard** to satisfy.

# Can't We Manipulate Our Way Out?

## Simple Manipulation Example

3 voters, 3 candidates (a, b, c). Plurality voting, alphabetical tiebreaking.

- Voter 1 ranks: $a$ first
- Voter 2 ranks: $b$ first
- Voter 3's true preference: $c \succ b \succ a$

If Voter 3 votes **honestly**: Three-way tie $\rightarrow$ $a$ wins (worst for Voter 3!)
If Voter 3 votes **strategically** ($b$ first): $b$ wins (better for Voter 3!)

Voter 3 benefits from **lying** about their preferences. This is **manipulation**.

# Gibbard-Satterthwaite: The Second Blow

**Theorem (Gibbard-Satterthwaite, 1973-1975)**

For 3+ candidates with no restrictions, every voting rule must be:

- **Dictatorial** (one voter decides everything), OR
- **Imposing** (some candidate can never win), OR
- **Manipulable** (voters benefit from lying)

**Every fair voting system can be manipulated.**

*Democracy seems doomed by mathematics...*

# Can We At Least Compute Results Quickly?

How long does it take to determine a winner? ($n$ candidates, $v$ voters)

- **Plurality:** $O(n + v)$ — super fast!
- **Borda/Scoring rules:** $O(nv)$ — reasonable
- **Single Transferable Vote:** $O(n^2 + nv)$ — still okay

But some "fair" rules are much worse...

# Kemeny's Rule: The Fairest of Them All

## Kemeny's Rule

Find the ranking that **minimizes total disagreement** with all voters.

Measures disagreement by counting: for each pair of candidates, how many voters rank them opposite to our proposed order?

This is the **only** rule satisfying:

- Neutrality (treat candidates equally)
- Consistency (if two groups agree, the combined group agrees)
- Condorcet winner (if someone beats all others head-to-head, they win)

**The catch?**

# The Computational Nightmare

**Theorem (Bartholdi, Tovey, Trick, 1989)**

Finding the Kemeny winner is **NP-hard**.

**What does NP-hard mean?**

- Some problems are easy to **solve**: sorting a list
- Some are easy to **check** but (probably) hard to solve: Sudoku
- NP-hard means: at least as hard as the hardest "check-but-not-solve" problems
- For large elections, this could take **longer than the age of the universe**

**The fairest voting rule is too slow to use!**

# End of Act 1: Despair

**What we've learned:**

- Perfect voting is impossible (Arrow)
- Fair voting can always be manipulated (Gibbard-Satterthwaite)
- The fairest computable rules are too slow (BTT Theorem)

Is democracy mathematically doomed?

**Wait...** What if computational hardness is actually a *feature*, not a bug?

# Act II

When Hardness Helps Democracy

# A Ray of Hope

We just saw: even finding the winner of a fair election can be **hard**.

**Wild idea:** If computing the winner is hard, maybe **finding a manipulation** is hard too!

Think about it:

- In India's 2024 election: 968 million registered voters
- Finding the *optimal* manipulation might require checking billions of voting scenarios
- If that takes centuries to compute, manipulation becomes impractical!

# What Does "Hard" Mean?

**Computer Science classifies problems by difficulty:**

- **Easy (P):** Can solve quickly: sorting a list, finding the shortest path etc
  - Finding the winner in Plurality voting: $\mathcal{O}(n)$ time
- **Hard (NP-complete):** Can *check* a solution quickly, but finding it may take forever
  - Like Sudoku: verifying a solution is easy, finding it is hard
  - The Traveling Salesman Problem
  - Packing your suitcases optimally (Bin packing problem)

**If manipulation is NP-hard, would-be manipulators face an intractable problem!**

# A Nice Result

## Bartholdi-Tovey-Trick Theorem (1989)

For **Single Transferable Vote (STV) and 2-nd Order Copeland** and similar systems:
Finding a successful manipulation is **NP-complete**

**What this means:**

- Manipulation is still *possible* (we can't escape Gibbard-Satterthwaite)
- But finding the manipulation requires solving a problem as hard as Sudoku, SAT, or TSP
- For large elections, this becomes computationally infeasible
- **Complexity theory saves democracy!**

# Manipulation: Easy Cases

**Bad news:** For many scoring rules (Borda, Plurality, etc.), manipulation by a **single voter** is easy.

**Greedy algorithm:** Try placing candidates in your fake vote one by one, picking whatever keeps your favorite ahead.

**But here's the thing...**

In large elections, one voter rarely makes a difference. The real threat is **coalitions**.

# Coalition Manipulation: The Real Threat

## The Con Artist's Problem

*"Just as no con happens alone..."*

What if a **group** of voters coordinates to manipulate the election?

## Good News for Democracy

For Borda rule, coalition manipulation with just **two manipulators** is **NP-complete**!

Even for simple rules, coordinating strategic voting becomes **computationally intractable**.

# Showcase Proof: Borda Coalition Manipulation

**Setup:** We'll show this is as hard as the "Permutation Sum" problem (which is NP-complete).

**Permutation Sum:** Given integers $X_1 \leq X_2 \leq \ldots \leq X_n$ summing to $n(n+1)$, do there exist two permutations $\sigma, \pi$ such that $\sigma(i) + \pi(i) = X_i$?

**Key insight:** We construct an election where:

- Non-manipulators create specific score gaps
- Two manipulators can only fix these gaps if Permutation Sum has a solution
- The manipulators must rank candidates such that their Borda points sum correctly

**Result:** If you can solve Borda coalition manipulation quickly, you can solve Permutation Sum quickly. Since the latter is NP-complete, so is the former!

# The Reduction

We show how to increase the score of a candidate by 1 more than the other candidates except for the last candidate whose score increases by 1 less.

For instance, suppose we wish to increase the score of candidate 1 by 1 more than candidates 2 to $m$ and by 2 more than candidate $m+1$.
Consider the following pair of votes:

$$1 > m+1 > 2 > \ldots > m-1 > m$$

$$m > m-1 > \ldots > 2 > 1 > m+1$$

The score of candidate 1 increases by $m+1$, of candidates 2 to $m$ by $m$, and of candidate $m+1$ by $m-1$.
By repeated use of this, we can achieve the result we desire. $\square$

# Completing the Reduction

Using the Lemma, we can construct the non-manipulators such that the score vector is:

$$\langle C, 2(n+2) - X_1 + C, \ldots, 2(n+2) - X_n + C, 2(n+2) + C, y \rangle$$

We claim two manipulators can make candidate 1 win if and only if the permutation sum problem has a solution.

($\Rightarrow$) As a permutation sum exists, the manipulators can vote as:

$$\langle n+2, \sigma(1), \ldots, \sigma(n), 0, n+1 \rangle$$

$$\langle n+2, \pi(1), \ldots, \pi(n), 0, n+1 \rangle$$

getting the score to:

$$\langle 2(n+2) + C, 2(n+2) + C, \ldots, 2(n+2) + C, 2(n+1) + y \rangle$$

and thus, getting candidate 1 to win.

# Completing the Reduction (cont.)

($\Leftarrow$) To ensure candidate 1 beats candidate $n+2$, both manipulators must put candidate 1 in first place and the latter in last.

Candidate 1 in future is above the $i+1$-th candidate by $X_i$ votes where $\sum_{i=1}^{n} X_i = n(n+1)$.

This means that if any of them get the score addition of $n+1$, candidate 1 will lose. So, $n+1$ scores will have to go to the last (and least dangerous) candidate.

This makes the manipulated votes of the form:

$$\langle n+2, \sigma(1), \ldots, \sigma(n), 0, n+1 \rangle$$

$$\langle n+2, \pi(1), \ldots, \pi(n), 0, n+1 \rangle$$

where $\sigma$ and $\pi$ are permutations of $1 \ldots n$.

# Completing the Reduction (final)

To ensure candidate 1 beats candidate $i+1$, we must have $\sigma(i) + \pi(i) \leq X_i$.

Since $\sum_{i=1}^{n} \sigma(i) = \frac{n(n+1)}{2}$ and $\sum_{i=1}^{n} \pi(i) = \frac{n(n+1)}{2}$; we must have $\sigma(i) + \pi(i) = X_i$.

This means, we have a solution of the permutation sum problem. $\square$

Similar proofs hold for Copeland etc.

# Real-World Example: NYC 2025

## New York City Mayoral Race

- Uses **Ranked Choice Voting (STV)**
- Eliminates lowest-scoring candidate each round till one candidate has majority.
- Opposition tried to convince Zohran Mamdani's voters to drop out: "Don't waste your vote!"
- But RCV is manipulation-resistant. Ranking him first doesn't "waste" your vote.
- Result: Mamdani won the Democratic primary (contrary to expectations).

**The system mattered.** Under plurality, strategic voting pressure might have worked. Under STV, voters could vote their conscience.

# But What About Those in Power?

Voters trying to manipulate is one thing.

But what about election officials, chairs, or those who control the process?

**Control:** When the people running the election try to rig it by:

- Adding "spoiler" candidates
- Removing candidates from the ballot
- Adding voters (selective enfranchisement)
- Removing voters (selective disenfranchisement)

# Real Examples of Electoral Control

These aren't hypothetical:

- **2000 Bush election:** Nader as spoiler candidate
- **Women's suffrage (Miller 2008):** Showed that post enfranchisement of women, candidates with focus on infant health spending gained votes. This led to some regimes in support of infant and maternal health enfranchising women.
- **Papal conclaves:**
  - Pope Paul VI: Cardinals over 80 can't vote, done to limit influence of Pope John XXIII and Pope Pius XII
  - Pope Francis (2025): Appointed 110 of 135 voting cardinals before his death

**Control is everywhere in real politics.** While in some cases it is for the greater good, it still is a form of manipulation of the election.

# Resistance vs. Vulnerability

**Key concepts:**

- **Immune:** Control is impossible
- **Susceptible:** Control is sometimes possible
- **Resistant:** Control is possible but NP-hard to find
- **Vulnerable:** Control is possible and easy (polynomial time)

Immunity for candidate control is rare. This is due to the study of candidate-voter models.

We can consider a setting where the candidates have preferences regarding election outcomes, and can strategically choose to join the race or not.

For the case of voter control, immunity is not only rare, but also is utterly undesirable. If we add sufficiently many voters with the same preference order, then their most preferred candidate should becomes the winner.

**What we want:** Resistence to control, not immunity!

# Showcase Proof: Control via Voter Registration

**Approval voting** is resistant to control by adding voters. Here's why:

**Reduction from Exact 3-Cover (X3C):**

- X3C: Given a list of $3k$ objects and sets of these objects of size 3, can you pick exactly $k$ sets that cover everything?
- X3C is a classic NP-complete problem
- We'll show: solving control $\Leftrightarrow$ solving X3C

**The construction:**

- Candidates: Elements to cover + our preferred winner $w$
- Already registered: $k-2$ voters who approve all elements, disapprove $w$
- Unregistered: One voter per set, approves that set + $w$
- Can register $k$ voters

# Why This Works

**The key insight:**

- If we have an exact 3-cover, register those $k$ voters
  - $w$ gets $k$ votes
  - Each element gets $(k-2)+1 = k-1$ votes
  - $w$ wins!
- If we can make $w$ win by registering $k$ voters
  - Each registered voter votes for 3 elements + $w$
  - For $w$ to win, every element must get at most $k-1$ votes
  - This means each element is covered exactly once
  - We've found an exact 3-cover!

**This is the power of reductions:** Transform one hard problem into another to prove hardness!

# Showcase Proof: Control via Voter Disenfranchisement

**Approval voting** is also resistant to control by deleting voters. Here's why:

**Reduction from Exact 3-Cover (X3C), Again!**

- X3C: Given a list of $3k$ objects and sets of these objects of size 3, can you pick exactly $k$ sets that cover everything?
- Let the objects be $B = \{b_1, b_2, \ldots, b_{3k}\}$ Let the sets be $S = \{S_1, S_2, \ldots, S_n\}$.
- We also define $l_i$ to be the number of sets $b_i$ is a part of.
- We'll show: solving control $\Leftrightarrow$ solving X3C

**The construction:**

- Candidates: Elements to cover + our preferred winner $w$
- Start with $2k$ voters,
- $k$ voters where $v_i$ approve elements only in $S_i$.
- $k$ more voters where all approve of $w$ and exactly $k - l_i$ of them approve of $b_i$.
- Can disenfranchise $k$ voters.

# Why This Works

**The key insight:**

- If we have an exact 3-cover, delete those $k$ voters
  - $w$ gets $k$ votes
  - Each element gets $(l_i - 1) + (k - l_i) = k - 1$ votes
  - $w$ wins!
- If we can make $w$ win by deleting $k$ voters
  - WLOG, we may assume that we must delete voters who DON'T approve $w$ (else $w$ loses score).
  - For $w$ to win, every element must get at most $k - 1$ votes
  - This means each element is covered exactly once
  - We've found an exact 3-cover!

**X3C strikes again!**

# What About Bribery

**Control:** Change the election structure
**Bribery:** Change the votes themselves

**The briber's problem:**

1. Who should I bribe?
2. How should I change their votes?
3. Can I afford it within my budget?

This combines control-like decisions (picking targets) with manipulation-like decisions (changing votes).

# Bribery: Easy and Hard Cases

**For plurality voting:**

- **Simple bribery:** Easy (P):greedy works!
  - Keep bribing cheapest voter supporting a winner to support your candidate
- **Weighted bribery:** Still easy (P) with a clever algorithm
  - Try different target scores, bribe heavy voters first
- **Weighted dollar bribery:** NP-complete!
  - Different voters have different costs AND weights
  - Reduces to the Partition Problem

# Weighted Bribery

For example, consider two algorithms that bribe the cheapest voter or heaviest voter till the result is turned. While none of them work alone. (Find the example!)

A combination of these two heuristics does yield a polynomial-time algorithm.

We borrow an idea from Parametrized Algorithms. We make an algorithm which is in P time using some parameter and then find the parameter in P time as well.

Here the parameter is the least amount of points $p$ must end up with after the bribery is done. Let's call it $T$.

Naturally, all the other alternatives have to end up with at most $T$ points.

Thus, for each alternative $a$ that has more than $T$ points, we should keep bribing its heaviest voters until its score decreases to at most $T$.

# Dollar Weighted-Bribery is NP-complete

Dollar Weighted-Bribery is easily shown to be NP-complete thanks to the partition problem.

**Reduction from Partition**

- PART: Given a set of integers summing to an even number, can we partition it into two disjoint sets with equal weight?
- Let the set of integers be $S = \{s_1, s_2, \ldots, s_n\}$
- PART is also a classic NP-complete problem
- We'll show: solving Dollar Weighted-Bribery $\Leftrightarrow$ solving PART

**The construction:**

- Candidates: Consider a two candidate election with $w$ being our preferred winner.
- Voters: We have $n$ voters with the cost and weight of $v_i = s_i$. All vote against $w$.
- Budget : Our budget is $(s_1 + s_2 + \ldots + s_n)/2$

**Hopefully, you can take it from here!**

# Act III

## Why This Matters

# Why Prevent Manipulation At All?

**You might ask:** If everyone manipulates optimally, won't we get the "right" outcome anyway?

**No! Four reasons this fails:**

- **Bad equilibria:** A candidate everyone likes might lose because no one thinks they can win
- **Lack of information:** We never learn true preferences
- **Disenfranchisement of the Unsophisticated Voter:** Sophisticated voters benefit, naive voters suffer
- **Wasted effort:** Energy spent strategizing instead of governing

# Recent Real Example: NYC 2025

**New York City Mayoral Primary (Democratic), 2025**

- We already saw this example back in the manipulation area.
- NYC Uses: **Ranked Choice Voting (IRV/STV)**
- Zohran Mamdani was portrayed as "unlikely to get elected" by opponents
- Strategy: Convince voters to rank him lower to "not waste votes"
- **Result:** Mamdani won anyway!

**Why it mattered:** IRV made the manipulation harder because:

- Voters could rank honestly without "wasting" their vote
- No need to waste time on strategic voting, focus can be on policy
- The system itself resisted manipulation
- Finally, candidates are not adversarial. Candidates can cross endorse each other. Lander and Mamdani did this. Lander asked everyone ranking him first to rank Mamdani second and vice versa with Mamdani and they both performed well. Same with Mamdani and Blake.

# Case Study: Sortition in Student Democracy

## LSE Student Union Experiment (Visinho et al.)

- Avoided power grabs by loud, power-hungry candidates
- Got more thoughtful, well-meaning representatives
- Changed *who runs* for office, not just who wins
- **The voting rule shaped who ran for office!**

**The lesson:** Systems don't just count preferences: they shape who participates and how.

# The Takeaway

**What we've learned:**

1. **Perfect voting is impossible** (Arrow, Gibbard-Satterthwaite, BTT)
2. **BUT:** Computational complexity can protect democracy
   - Making manipulation NP-hard = making it practically impossible
   - Even "imperfect" rules can be resistant to attack
3. **Voting rules matter in practice**
   - They determine who runs for office
   - They shape campaign strategies
   - They affect who gets represented
   - And the obviously, determine what kind of a candidate wins.

# What We Didn't Cover (But You Should Explore!)

- Ranked Pairs, Approval, and dozens of other voting rules
- Parametrized and approximation algorithms
- Voting on restricted domains (modeling partisanship)
- Game-theoretic voting and voting on incomplete information
- More types of Control and Bribery
- Multimodal attacks (control + bribery together)
- Voting on combinatorial domains (matching, fair division)
- and so much more

Want to guess why we didn't see a lot of these?

One, because I have in the two student seminars I got to take, built a reputation for going over time and I want to fix that.

Two, as a lot of this is open.

# Why so many open problems?

Because voting theory is **young** and **active** and **cool**!

- Major results published **two weeks ago** (Bui, Chavrimootoo, Le 2025)
  - New approximation algorithms for manipulation of approval or Condorcet
  - First application of Minimum k-Union to social choice
- Open problems everywhere
- You could contribute as an undergrad! (or so I have been told ... I am yet to find success in this regard : $-<$)

# Thanks for listening!

Questions?

(And yes, we can discuss any of the technical proofs (including those omitted) in detail!)