Search versus Search for Collapsing Electoral Control Types

B. Carleton[1], **Michael C. Chavrimootoo[2],**
L. Hemaspaandra[3], D. Narváez[4], C. Taliancich[5], H. Welles[3]

- EUMAS 2024

[1]:Cornell university, [2]:Denison University, [3]:University of Rochester, [4]:Virginia Tech, [5]:Property Matrix

Example of an Election

- **Context:** The chair of the CS department is ***hiring one of three faculty candidates.*** They've gathered the preferences of the CS faculty and will conduct an ***approval election****** to select a candidate.
- Candidates: ✪ (star researcher), 🕮 (exceptional teacher), $\bowtie$ (well-rounded applicant).
- Each voter (of 12) states the candidates that they approve of.

1. ✪	5. 🕮 $\bowtie$	9. $\bowtie$✪
2. ✪	6. 🕮 $\bowtie$	10. $\bowtie$ 🕮
3. ✪	7. 🕮	11. $\bowtie$✪ 🕮
4. ✪ $\bowtie$	8. $\bowtie$✪	12. $\bowtie$ 🕮

$\bowtie$ wins!

*: Each voter approves a subset of the candidates. A candidate receives one point for each vote that approves them. A winner is a candidate with maximal score.

Election System (aka Voting Rule)

An **election system** is a function that maps a set of candidates C and a set of votes V to a subset (aka winner set) of C.

- (C, V) is called an **election**.
- Informally, it describes how to select from C given some preferences (votes).
- E.g., approval voting, plurality, majority, ranked choice, etc.

There are **different types of votes**. Common ones are approval ballots (i.e., sets of "approved" candidates) and linear orders.

- E.g., An approval ballot over {Harris, Trump, Kennedy} is {Harris, Kennedy}.
- E.g., A linear order over {Harris, Trump, Kennedy}, is Harris $>$ Kennedy $>$ Trump.

Control by Partitioning of Voters

The Chair sees that ⋈ wins, but wants to hire ✯, so they partition the voters.

First Round: Voters are separated, and each subcommittee runs a subelection with all three candidates.

- Subcommittee 1:

1. ✯
2. ✯
3. ✯
4. ✯ ⋈
8. ⋈✯
9. ⋈✯
10. ⋈ 🕮

✯ wins!

- Subcommittee 2:

5. 🕮 ⋈
6. 🕮 ⋈
7. 🕮
11. ⋈✯🕮
12. ⋈🕮

🕮 wins!

Second/Final Round: First-round winners* compete in final round.

- Candidates (2): ✯ and 🕮
- Voters (12) with preferences over the remaining candidates:

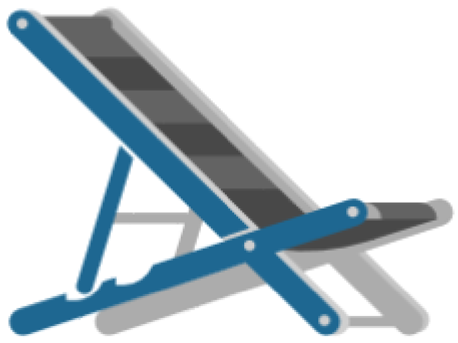
1. ✯
2. ✯
3. ✯
4. ✯
5. 🕮
6. 🕮
7. 🕮
8. ✯
9. ✯
10. 🕮
11. ✯🕮
12. 🕮

✯ wins!

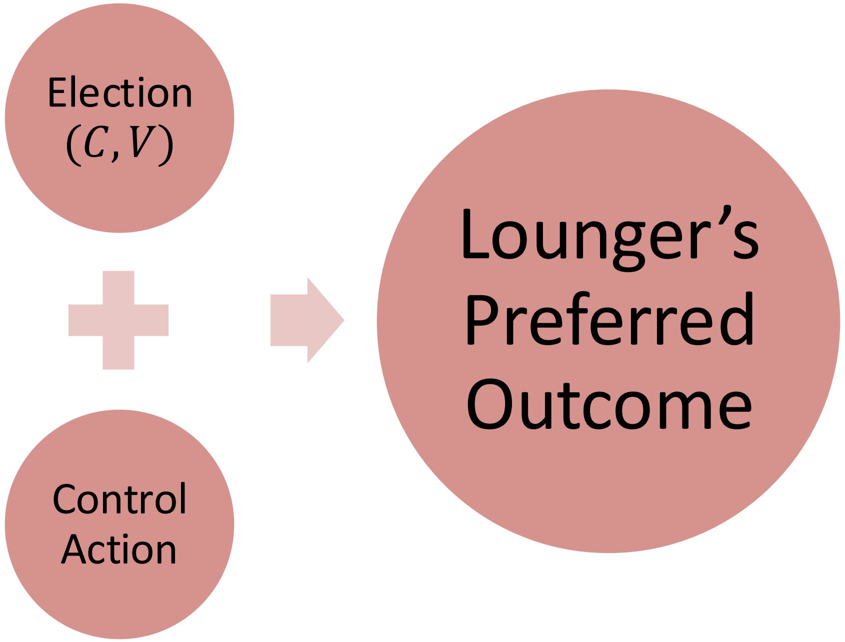
Electoral Control (Informally)

When an election chair can **alter the structure** of an election to yield their preferred outcome (make a candidate win/not win).

- Common actions to alter structure are add/delete/partition candidates/voters.



Lounger, the electoral chair



The Standard* Control Types in COMSOC

Two Main Categories: Nonpartition-Based and Partition-Based

*These are the longstanding models since Bartholdi et al. (1992) and Hemaspaandra et al. (2007).

The 24 Partition-Based Types

Outcome

- **Constructive (CC)**: Make a specified candidate win.
- **Destructive (DC)**: Prevent a specified candidate from winning.

Action

- **Partition of Voters (PV)**: Partition the votes to form two subelections, whose winner(s) compete in a final round.
- **Partition of Candidates (PC)**: Partition the candidates to form one subelection whose winner(s) compete against the remaining candidates.
- **Run-Off Partition of Candidates (RPC):** Like PV, but partition candidates.

Tie-Handling

- **Ties Eliminate (TE)**: Tied winners of a subelection are eliminated.
- **Ties Promote (TP)**: Promote every subelection winner.

Winner Model

- **Unique Winner (UW)**: At most one candidate can be a winner.
- **Nonunique Winner (NUW)**: Multiple candidates can be winners.

The 20 Nonpartition-Based Types

Outcome

- Constructive **(CC):** Make a specified candidate win.
- Destructive **(DC):** Prevent a specified candidate from winning.

Action

- Unlimited Adding Candidates **(UAC)**
- Adding Candidates **(AC)**
- Adding Votes **(AV)**
- Deleting Candidates **(DC)**
- Deleting Votes **(DV)**

Winner Model

- Unique Winner **(UW):** At most one candidate can be a winner.
- Nonunique Winner **(NUW):** Multiple candidates can be winners.

The Typical Model

Decision Model

- Questions with Yes/No answers; Standard in Theoretical Computer Science.

Approval-CC-PV-TE-UW (Informally)

- **Inputs:** Candidate set C, vote set V, focus candidate p.
- **Question:** Is there a partition of V such that p is the *unique winner* of the two-stage election where the surviving winners of the subelections compete in a final round.

Using our toy example

- $C = \{$✯, 🕮, ⋈$\}$, V = {…}
- $(C, V,$ ✯$)$ ∈ Approval-CC-PV-TE-UW.

This is a set!

Collapsing Control Types

Hemaspaandra et al. (2020) and Carleton et al. (2024)

For each election system $\mathcal{E}$,

- Destructive control using the TE model yields the same outcomes for each candidate partitioning and both winner models.
- Destructive control using the TP + NUW models is the same for both types of candidate partitioning.

$\boxed{\binom{4}{2} = 6 \text{ pairs}}$

When looking at specific election systems:

$\boxed{1 \text{ pair}}$

- There is one more collapsing pair under veto.
- There are 14 more collapsing pairs under approval.
- We also know of certain collapses that follow from axiomatic properties.

Same decision complexity (e.g., in P or NP-complete, etc.)

Do Collapsing Types Share *Search* Complexity?

Why Consider Search Complexity?

<u>In practice:</u> One wants algorithms to compute “solutions”, not just determine their existence.

- We can often leverage "self-reducibility", but can we always?

Membership in P <u>does not guarantee</u> a polynomial-time search algorithm!

- True under reasonable assumptions.
- I.e., if two problems have the same *decision* complexity, they may have witness schemes with <u>different *search* complexity.</u>

Insight: A Surprising "Separation" of Search and Decision

Borodin and Demers (1976): There is a set $A \subseteq \mathrm{SAT}$ where $A \in \mathrm{P}$, but no polynomial-time algorithm can always find a satisfying assignment.

- Assuming $\mathrm{P} \neq \mathrm{NP} \cap \mathrm{coNP}$, often considered reasonable.

Hemaspaandra et al. (2020): There are decision control problems that are in P and yet no polynomial-time algorithm can compute a successful attack.

- They embed the technique of Borodin and Demers and give more general results, about manipulative actions.

Search versus Search

Search can separate from search

- Using the technique of Borodin and Demers, you can give a certificate scheme for Σ^* that has no polynomial-time algorithm.

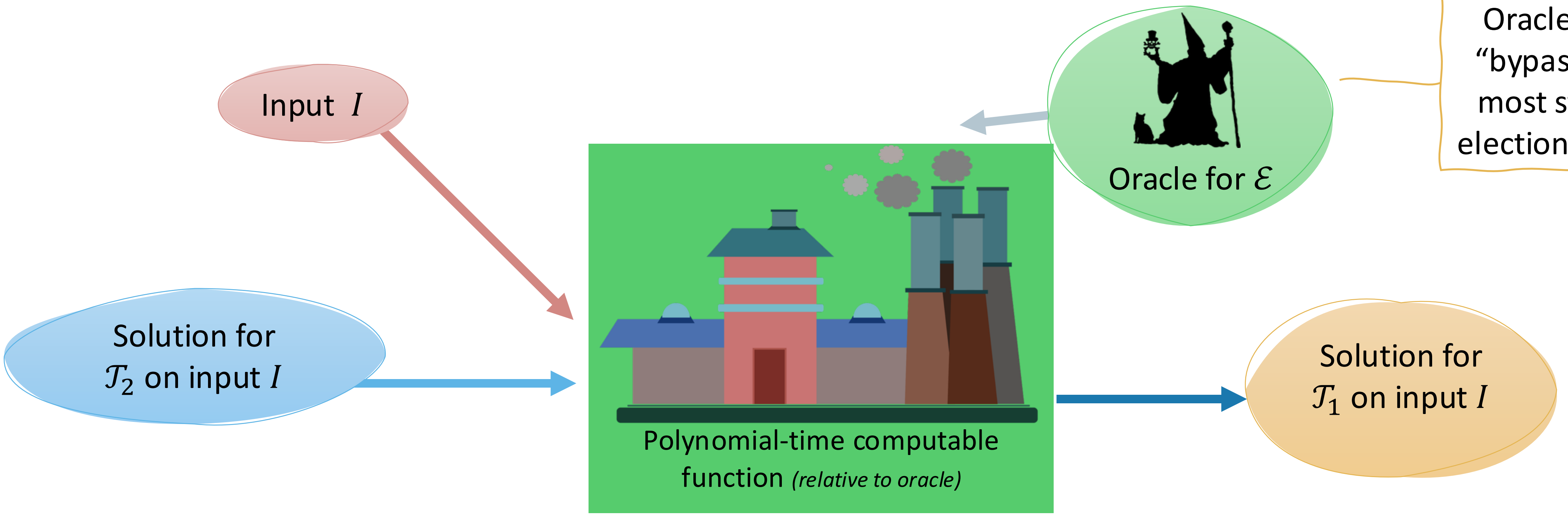
Our Question: Can it be that two control types $\mathcal{T}_1$ and $\mathcal{T}_2$ collapse as decision problems, and yet their search complexities differ?

- For all the known collapses, we prove **NO** by developing a framework.

Our Search Reductions

- $\boxed{\mathcal{T}_1 \leq^{p,\mathcal{E}}_{\text{search}} \mathcal{T}_2}$ ($\mathcal{T}_1$ is polynomially-search reducible to $\mathcal{T}_2$ w.r.t. $\mathcal{E}$):
  - There is an algorithm that runs in polynomial time relative an oracle for $\mathcal{E}$ and on each input (I, S), if S is a solution for I under $\mathcal{T}_2$, then the algorithm outputs a solution for I under $\mathcal{T}_1$.

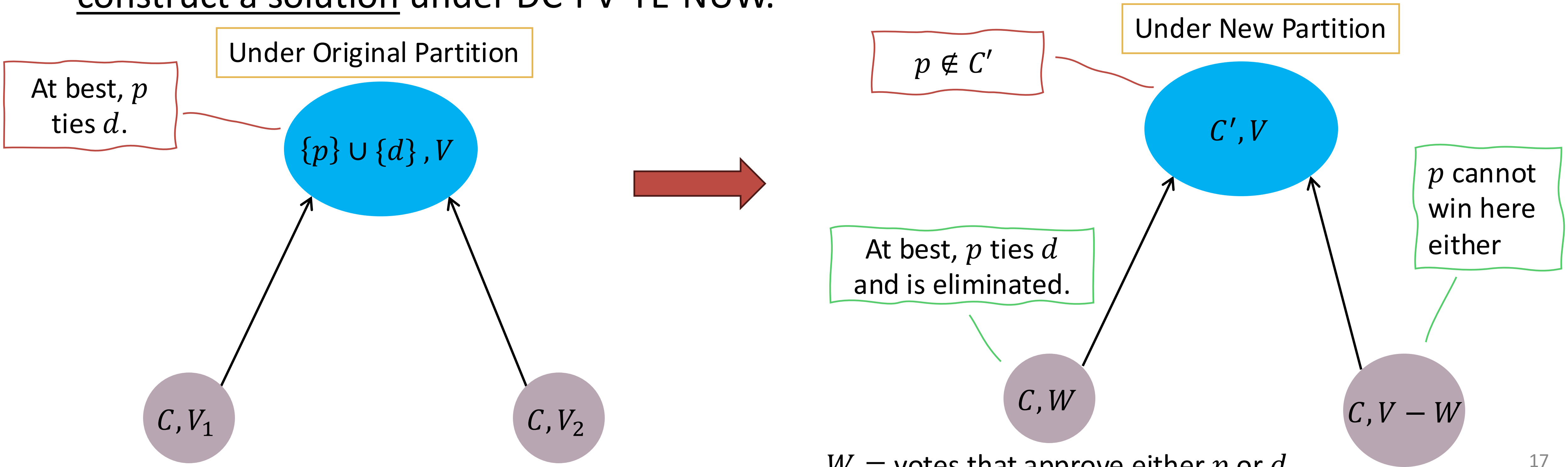
Assumes the types collapse



Oracle can be "bypassed" for most standard election systems!

Sketch: Approval-DC-PV-TE-NUW $\leq^{p}_{\mathbf{search}}$ Approval-DC-PV-TE-UW

- **Assume:** Given input (C, V) and solution (V_1, V_2) such that p is not the unique winner of the two-stage election $\therefore (C, V, p) \in \text{Approval}-\text{DC}-\text{PV}-\text{TE}-\text{UW}$.
  - *Let us look at one case:* if p is in the final election under (V_1, V_2).
- **Prove:** There is a voter partition such that p is not present in the final election, i.e., construct a solution under DC-PV-TE-NUW.



W = votes that approve either p or d.

Our Main Findings I: Search-Relationships

For each **known** <u>collapsing</u> electoral control types $\mathcal{T}_1, \mathcal{T}_2$ that are about $\mathcal{E}$:

- $\boldsymbol{\mathcal{T}_1 \equiv^{p}_{\mathbf{search}} \mathcal{T}_2}$: If $\mathcal{E}$ is *polynomial-time computable* <u>or</u> $\mathcal{E}$ *satisfies Property Unique-*α^*.
  - $\mathcal{E}$ satisfies Unique-α if p being the unique winner of election (C, V) implies that p is the unique winner of every election (C', V), where $p \in C' \subseteq C$.
- $\boldsymbol{\mathcal{T}_1 \equiv^{p,\mathcal{E}}_{\mathbf{search}} \mathcal{T}_2}$: Otherwise.

Our reductions "transfer" "easiness/hardness".

- If two problems are "poly. search-equivalent" (search-reduce to each other) and one is "easy"/"hard", then so is the other.

Our Main Findings II: Concrete Complexities

Give a notion of “SAT-equivalence”.

- Captures the notion of being "as hard" as SAT from a search complexity perspective.

For each known collapsing control problem, we determine polynomial-time computability or “SAT-equivalence”.

- New polynomial-time algorithms, sometimes via immunity arguments.
- Implicit algorithms via search algorithms!
- New “bridge theorem” that connects NP-completeness and SAT-equivalence for

Plurality-DC-PC-TP-NUW (= Plurality-DC-RPC-TP-NUW) is NP-complete.

Future Directions

Provide concrete examples or sufficient conditions to separate search from search.

“Relax” the assumptions in our bridge theorem.

Explore conditions under which SAT-equivalence implies NP-completeness of the respective decision problems.

Provide dichotomy theorems for polynomial-time computability vs. SAT-equivalence.

THANK YOU FOR YOUR ATTENTION! QUESTIONS?

LOCATION: RODRIGUES ISL, MAURITIUS