Using computational hardness as a barrier against manipulation

Vincent Conitzer
conitzer@cs.duke.edu

Inevitability of manipulability

- Ideally, our mechanisms are strategy-proof, but may be too much to ask for
- Recall Gibbard-Satterthwaite theorem:

  Suppose there are at least 3 alternatives

  There exists no rule that is simultaneously:
  - onto (for every alternative, there are some votes that would make that alternative win),
  - nondictatorial, and
  - strategy-proof
- Typically don't want a rule that is dictatorial or not onto
- With restricted preferences (e.g., single-peaked preferences), we may still be able to get strategy-proofness
- Also if payments are possible and preferences are quasilinear

Computational hardness as a barrier to manipulation

- A (successful) manipulation is a way of misreporting one's preferences that leads to a better result for oneself
- Gibbard-Satterthwaite only tells us that for some instances, successful manipulations exist
- It does not say that these manipulations are always easy to find
- Do voting rules exist for which manipulations are computationally hard to find?

A formal computational problem

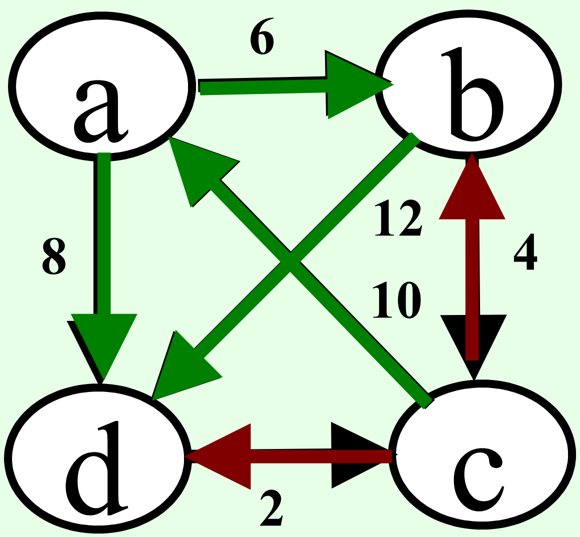
- The simplest version of the manipulation problem:
- CONSTRUCTIVE-MANIPULATION:
  - We are given a voting rule *r*, the (unweighted) votes of the other voters, and an alternative *p*.
  - We are asked if we can cast our (single) vote to make *p* win.
- E.g., for the Borda rule:
  - Voter 1 votes A > B > C
  - Voter 2 votes B > A > C
  - Voter 3 votes C > A > B
- Borda scores are now: A: 4, B: 3, C: 2
- Can we make B win?
- Answer: YES. Vote B > C > A (Borda scores: A: 4, B: 5, C: 3)

Early research

- **Theorem.** CONSTRUCTIVE-MANIPULATION is NP-complete for the second-order Copeland rule. [Bartholdi, Tovey, Trick 1989]
  - Second order Copeland = alternative's score is sum of Copeland scores of alternatives it defeats
- **Theorem.** CONSTRUCTIVE-MANIPULATION is NP-complete for the STV rule. [Bartholdi, Orlin 1991]
- Most other rules are easy to manipulate (in P)

Ranked pairs rule [Tideman 1987]

- Order pairwise elections by decreasing strength of victory
- Successively "lock in" results of pairwise elections unless it causes a cycle



Final ranking:
c>a>b>d

- **Theorem.** CONSTRUCTIVE-MANIPULATION is NP-complete for the ranked pairs rule [Xia et al. IJCAI 2009]

“Tweaking” voting rules

- It would be nice to be able to tweak rules:
  - Change the rule slightly so that
    - Hardness of manipulation is increased (significantly)
    - Many of the original rule’s properties still hold
- It would also be nice to have a single, universal tweak for all (or many) rules
- One such tweak: add a preround [Conitzer & Sandholm IJCAI 03]

Adding a preround

[Conitzer & Sandholm IJCAI-03]

- A preround proceeds as follows:
  - *Pair* the alternatives
  - Each alternative faces its opponent in a *pairwise election*
  - The winners proceed to the original rule
- Makes many rules hard to manipulate

Preround example (with Borda)

STEP 1:
A. Collect votes and
B. Match alternatives
(no order required)

Voter 1: A>B>C>D>E>F
Voter 2: D>E>F>A>B>C
Voter 3: F>D>B>E>C>A

Match A with B
Match C with F
Match D with E

STEP 2:
Determine winners of preround

A vs B: A ranked higher by 1,2
C vs F: F ranked higher by 2,3
D vs E: D ranked higher by all

STEP 3:
Infer votes on remaining alternatives

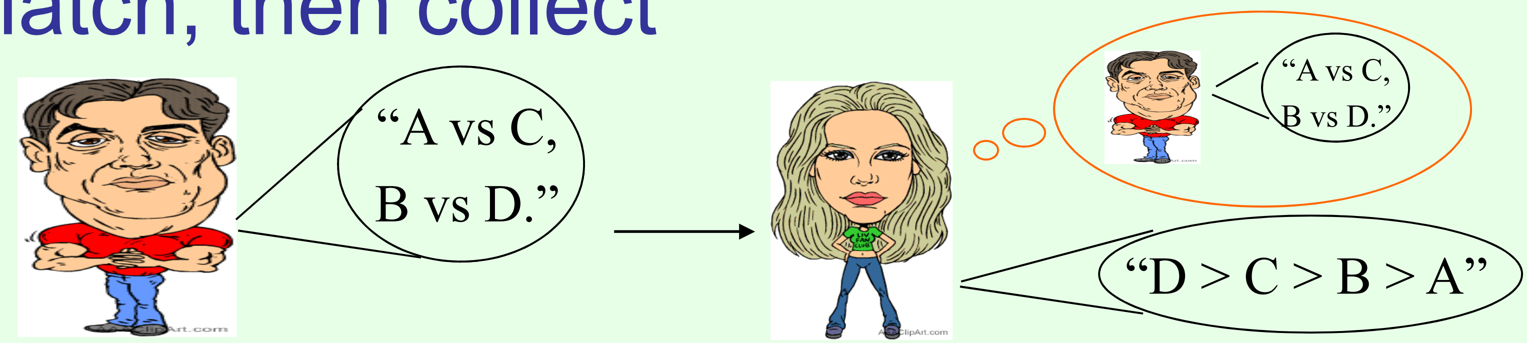
Voter 1: A>D>F
Voter 2: D>F>A
Voter 3: F>D>A

STEP 4:
Execute original rule
(Borda)

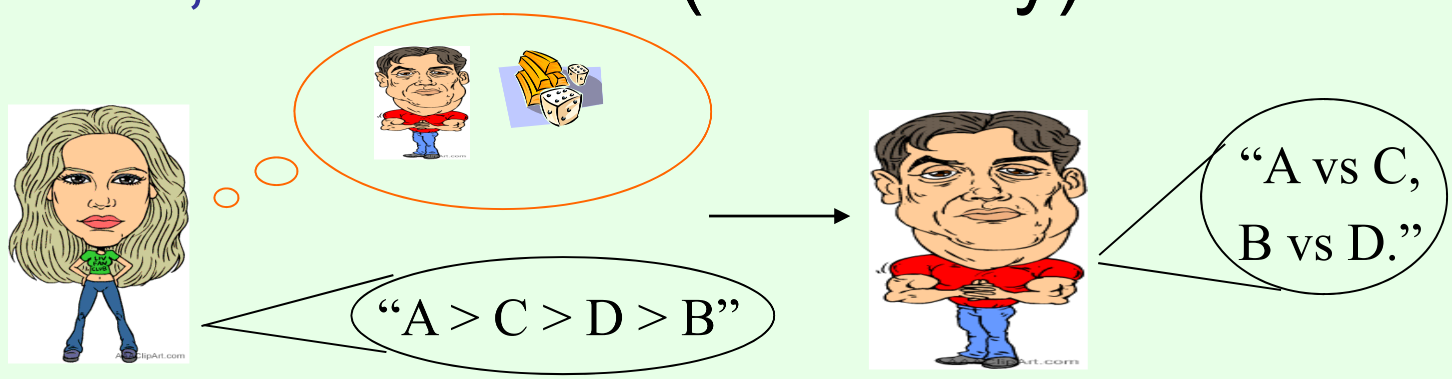
A gets 2 points
F gets 3 points
D gets 4 points and wins!

Matching first, or vote collection first?

- Match, then collect

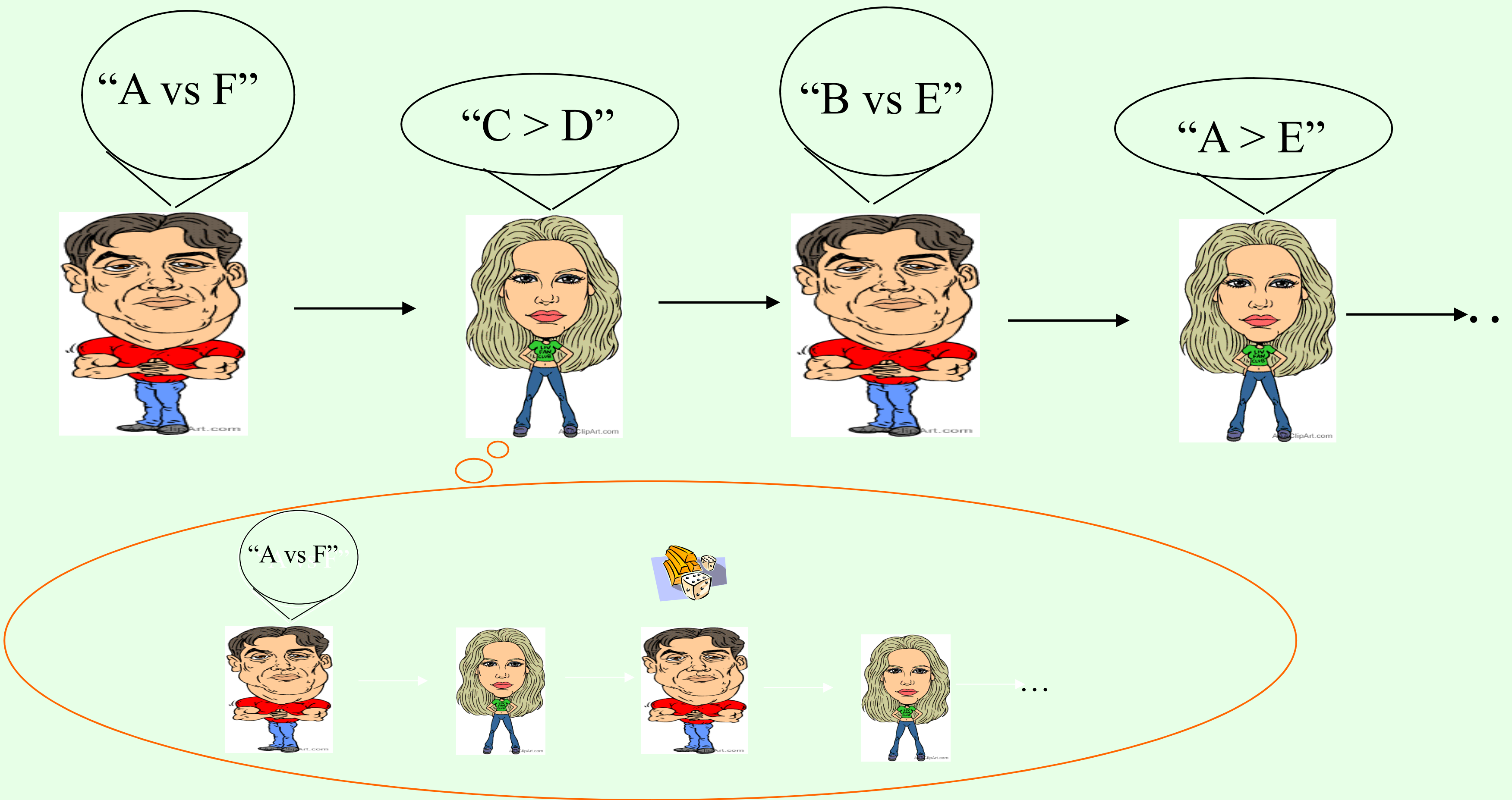


- Collect, then match (randomly)



Could also interleave…

- Elicitor alternates between:
  - (Randomly) announcing part of the matching
  - Eliciting part of each voter’s vote



How hard is manipulation when a preround is added?

- Manipulation hardness differs depending on the order/interleaving of preround matching and vote collection:
- **Theorem.** NP-hard if preround matching is done first
- **Theorem.** #P-hard if vote collection is done first
- **Theorem.** PSPACE-hard if the two are interleaved (for a complicated interleaving protocol)
- In each case, the tweak introduces the hardness for *any* rule satisfying certain sufficient conditions
  – All of Plurality, Borda, Maximin, STV satisfy the conditions in all cases, so they are hard to manipulate with the preround

What if there are few alternatives? [Conitzer et al. JACM 2007]

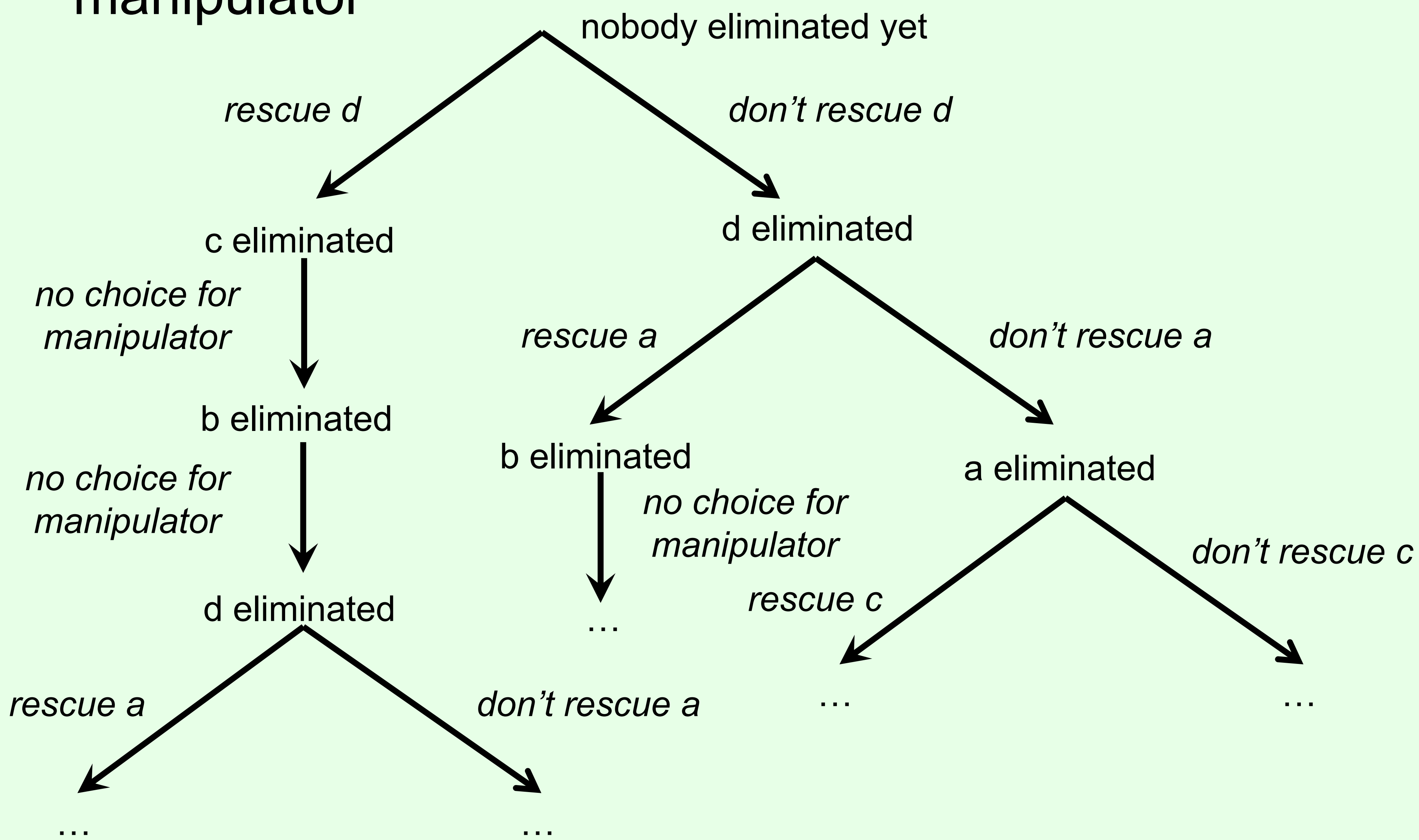
- The previous results rely on the number of alternatives (*m*) being unbounded
- There is a recursive algorithm for manipulating STV with $O(1.62^m)$ calls (and usually much fewer)
- E.g., 20 alternatives: $1.62^{20} = 15500$

- Sometimes the alternative space is much larger
  - Voting over allocations of goods/tasks
  - California governor elections

- But what if it is not?
  - A typical election for a representative will only have a few

STV manipulation algorithm

- Idea: simulate election under various actions for the manipulator



Analysis of algorithm

- Let *T(m)* be the maximum number of recursive calls to the algorithm (nodes in the tree) for *m* alternatives
- Let *T'(m)* be the maximum number of recursive calls to the algorithm (nodes in the tree) for *m* alternatives given that the manipulator's vote is currently committed
- $T(m) \leq 1 + T(m-1) + T'(m-1)$
- $T'(m) \leq 1 + T(m-1)$
- Combining the two: $T(m) \leq 2 + T(m-1) + T(m-2)$
- The solution is $O(((1+\sqrt{5})/2)^m)$
- Note this is only worst-case; in practice manipulator probably won't make a difference in most rounds

Manipulation complexity *with few alternatives*

- Ideally, would like hardness results for *constant* number of alternatives
- But then manipulator can simply evaluate each possible vote
  - assuming the others' votes are known & executing rule is in P
- Even for coalitions of manipulators, there are only polynomially many *effectively different* vote profiles (if rule is anonymous)
- However, if we place *weights* on votes, complexity may return…

Unbounded #alternatives

	Unweighted voters	Weighted voters
Individual manipulation	Can be hard	Can be hard
Coalitional manipulation	Can be hard	Can be hard

Constant #alternatives

	Unweighted voters	Weighted voters
Individual manipulation	easy	easy
Coalitional manipulation	easy	Potentially hard

Constructive manipulation now becomes:

- We are given the weighted votes of the others (with the weights)
- And we are given the weights of members of our coalition
- Can we make our preferred alternative *p* win?
- E.g., another Borda example:
- Voter 1 (weight 4): A>B>C, voter 2 (weight 7): B>A>C
- Manipulators: one with weight 4, one with weight 9
- Can we make C win?
- Yes! Solution: weight 4 voter votes C>B>A, weight 9 voter votes C>A>B
  - Borda scores: A: 24, B: 22, C: 26

A simple example of hardness

- We want: given the other voters' votes…
- … it is NP-hard to find votes for the manipulators to achieve their objective
- Simple example: veto rule, constructive manipulation, 3 alternatives
- Suppose, from the given votes, *p* has received 2K-1 more vetoes than *a*, and 2K-1 more than *b*
- The manipulators' combined weight is 4K
  - every manipulator has a weight that is a multiple of 2
- The only way for *p* to win is if the manipulators veto *a* with 2K weight, and *b* with 2K weight
- But this is doing PARTITION => NP-hard!

What does it mean for a rule to be *easy* to manipulate?

- Given the other voters' votes…
- …there is a polynomial-time algorithm to find votes for the manipulators to achieve their objective
- If the rule is computationally easy to run, then it is easy to check whether a given vector of votes for the manipulators is successful
- **Lemma:** Suppose the rule satisfies (for some number of alternatives):
  - If there is a successful manipulation…
  - … then there is a successful manipulation where all manipulators vote identically.
- Then the rule is easy to manipulate (for that number of alternatives)
  - Simply check all possible orderings of the alternatives (constant)

Example: Maximin with 3 alternatives is easy to manipulate constructively

- Recall: alternative's Maximin score = worst score in any pairwise election
- 3 alternatives: *p, a, b*. Manipulators want *p* to win
- Suppose there exists a vote vector for the manipulators that makes *p* win
- WLOG can assume that all manipulators rank p first
  - So, they either vote *p* > *a* > *b* or *p* > *b* > *a*
- Case I: *a*'s worst pairwise is against *b*, *b*'s worst against *a*
  - One of them would have a maximin score of at least half the vote weight, and win (or be tied for first) => cannot happen
- Case II: one of *a* and *b*'s worst pairwise is against *p*
  - Say it is *a*; then can have all the manipulators vote *p* > *a* > *b*
    - Will not affect *p* or *a*'s score, can only decrease *b*'s score

Results for *constructive* manipulation

Number of candidates	2	3	4,5,6	≥ 7
Borda	P	NP-c	NP-c	NP-c
veto	P	NP-c*	NP-c*	NP-c*
STV	P	NP-c	NP-c	NP-c
plurality with runoff	P	NP-c*	NP-c*	NP-c*
Copeland	P	P*	NP-c	NP-c
maximin	P	P*	NP-c	NP-c
randomized cup	P	P*	P*	NP-c
regular cup	P	P	P	P
plurality	P	P	P	P

Complexity of CONSTRUCTIVE CW-MANIPULATION

Destructive manipulation

- Exactly the same, except:
- Instead of a preferred alternative
- We now have a hated alternative
- Our goal is to make sure that the hated alternative does not win (whoever else wins)

Results for *destructive* manipulation

Number of candidates	2	≥ 3
STV	P	NP-c*
plurality with runoff	P	NP-c*
randomized cup	P	?
Borda	P	P
veto	P	P*
Copeland	P	P
maximin	P	P
regular cup	P	P
plurality	P	P

Complexity of DESTRUCTIVE CW-MANIPULATION

Hardness is only worst-case…

- Results such as NP-hardness suggest that the runtime of any successful manipulation algorithm is going to grow dramatically on some instances
- But there may be algorithms that solve most instances fast
- Can we make most manipulable instances hard to solve?

Bad news…

- Increasingly many results suggest that **many instances are in fact easy to manipulate**
- Heuristic algorithms [Conitzer & Sandholm AAAI-06, Procaccia & Rosenschein JAIR-07]
- Results showing that whether the manipulators can make a difference depends primarily on their number
  - If n nonmanipulator votes drawn i.i.d., with high probability, $o(\sqrt{n})$ manipulators cannot make a difference, $\omega(\sqrt{n})$ can make any alternative win that the nonmanipulators are not systematically biased against [Procaccia & Rosenschein AAMAS-07, Xia & Conitzer EC-08a]
  - Border case of $\Theta(\sqrt{n})$ has been investigated [Walsh IJCAI-09]
- Quantitative versions of Gibbard-Satterthwaite showing that under certain conditions, for some voter, even a random manipulation on a random instance has significant probability of succeeding [Friedgut, Kalai, Nisan FOCS-08; Xia & Conitzer EC-08b; Dobzinski & Procaccia WINE-08]

Weak monotonicity

voting rule — alternative set — nonmanipulator votes — nonmanipulator weights — manipulator weights

- An instance (R, C, v, k_v, k_w) is weakly monotone if for every pair of alternatives c_1, c_2 in C, one of the following two conditions holds:

- either: c_2 does not win for any manipulator votes w,
- or: if all manipulators rank c_2 first and c_1 last, then c_1 does not win.

A simple manipulation algorithm

[Conitzer & Sandholm AAAI 06]

Find-Two-Winners(R, C, v, k_v, k_w)

- choose arbitrary manipulator votes w_1
- $c_1 \leftarrow R(C, v, k_v, w_1, k_w)$
- for every c_2 in C, $c_2 \neq c_1$
  - choose w_2 in which every manipulator ranks c_2 first and c_1 last
  - $c \leftarrow R(C, v, k_v, w_2, k_w)$
  - if $c \neq c_1$ return $\{(w_1, c_1), (w_2, c)\}$
- return $\{(w_1, c_1)\}$

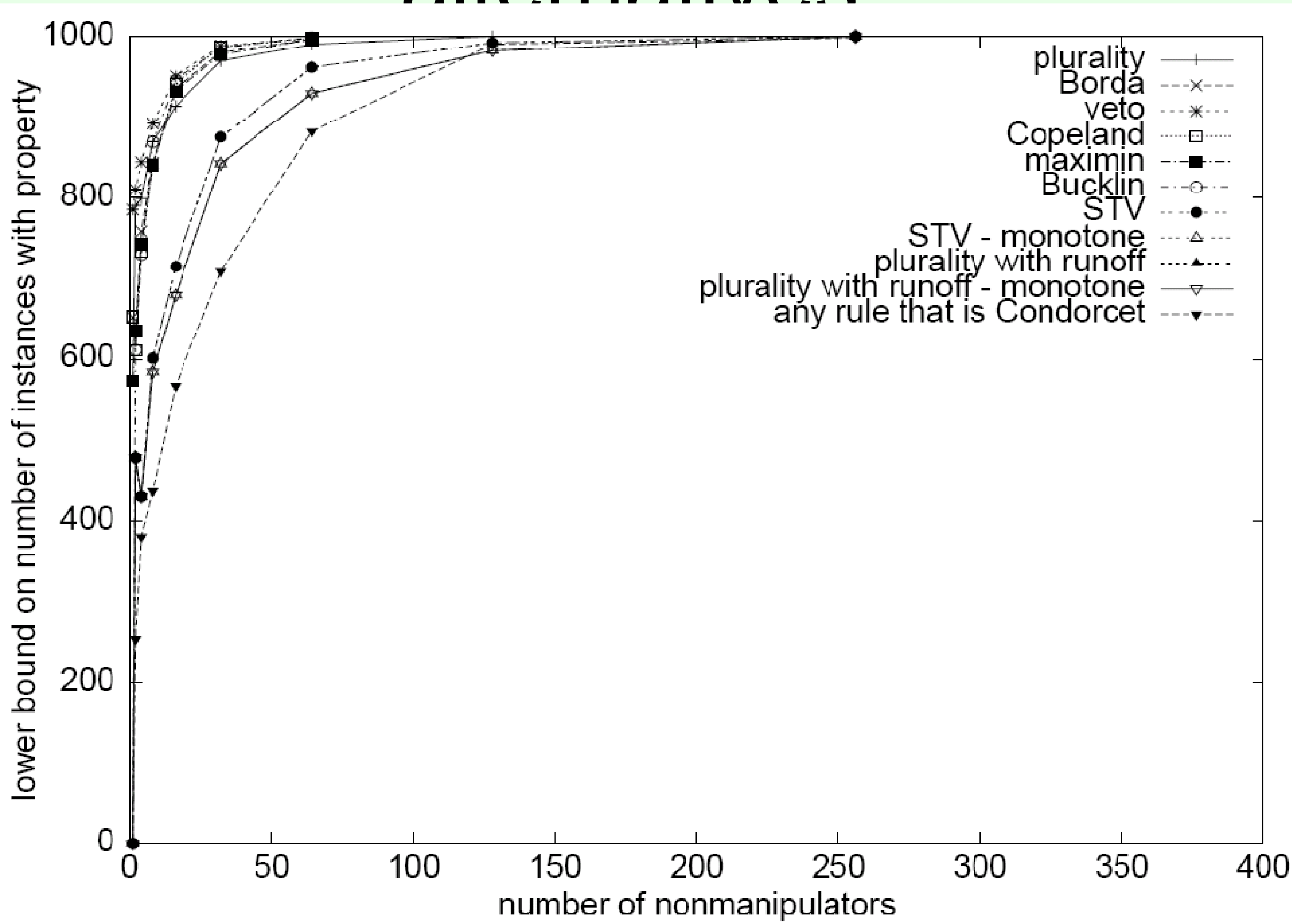
Correctness of the algorithm

- **Theorem.** Find-Two-Winners succeeds on every instance that
  - (a) is weakly monotone, and
  - (b) allows the manipulators to make either of exactly two alternatives win.
- **Proof.**
  - The algorithm is sound (never returns a wrong *(w, c)* pair).
  - By (b), all that remains to show is that it will return a second pair, that is, that it will terminate early.
  - Suppose it reaches the round where c_2 is the other alternative that can win.
  - If $c = c_1$ then by weak monotonicity (a), c_2 can never win (contradiction).
  - So the algorithm must terminate.

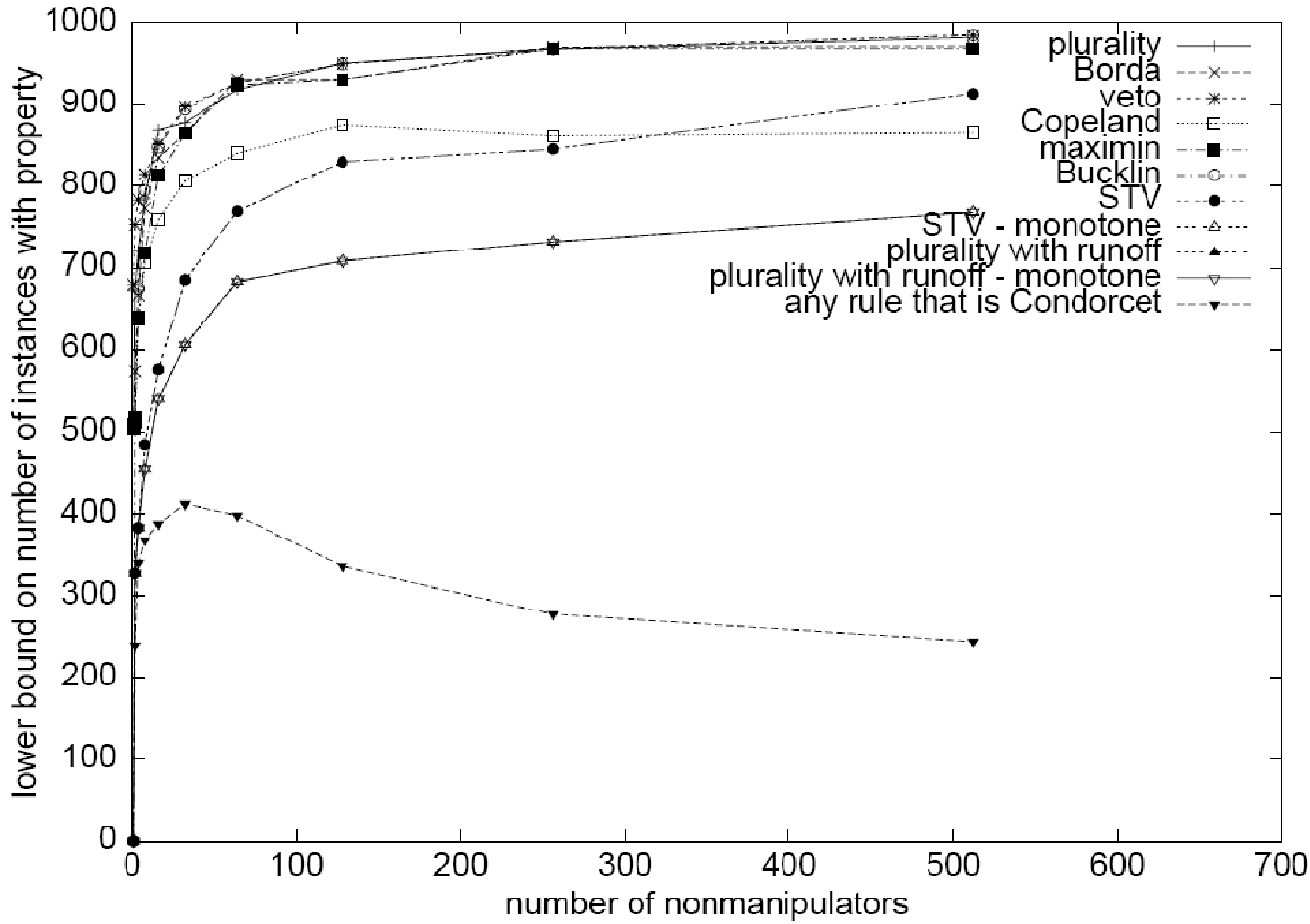
Experimental evaluation

- For what % of manipulable instances do properties (a) and (b) hold?
  - Depends on distribution over instances…
- Use Condorcet's distribution for nonmanipulator votes
  - There exists a correct ranking *t* of the alternatives
  - Roughly: a voter ranks a pair of alternatives correctly with probability *p*, incorrectly with probability *1-p*
    - Independently? This can cause cycles…
  - More precisely: a voter has a given ranking *r* with probability proportional to $p^{a(r, t)}(1-p)^{d(r, t)}$ where $a(r, t)$ = # pairs of alternatives on which *r* and *t* agree, and $d(r, t)$ = # pairs on which they disagree
- Manipulators all have weight *1*
- Nonmanipulable instances are thrown away

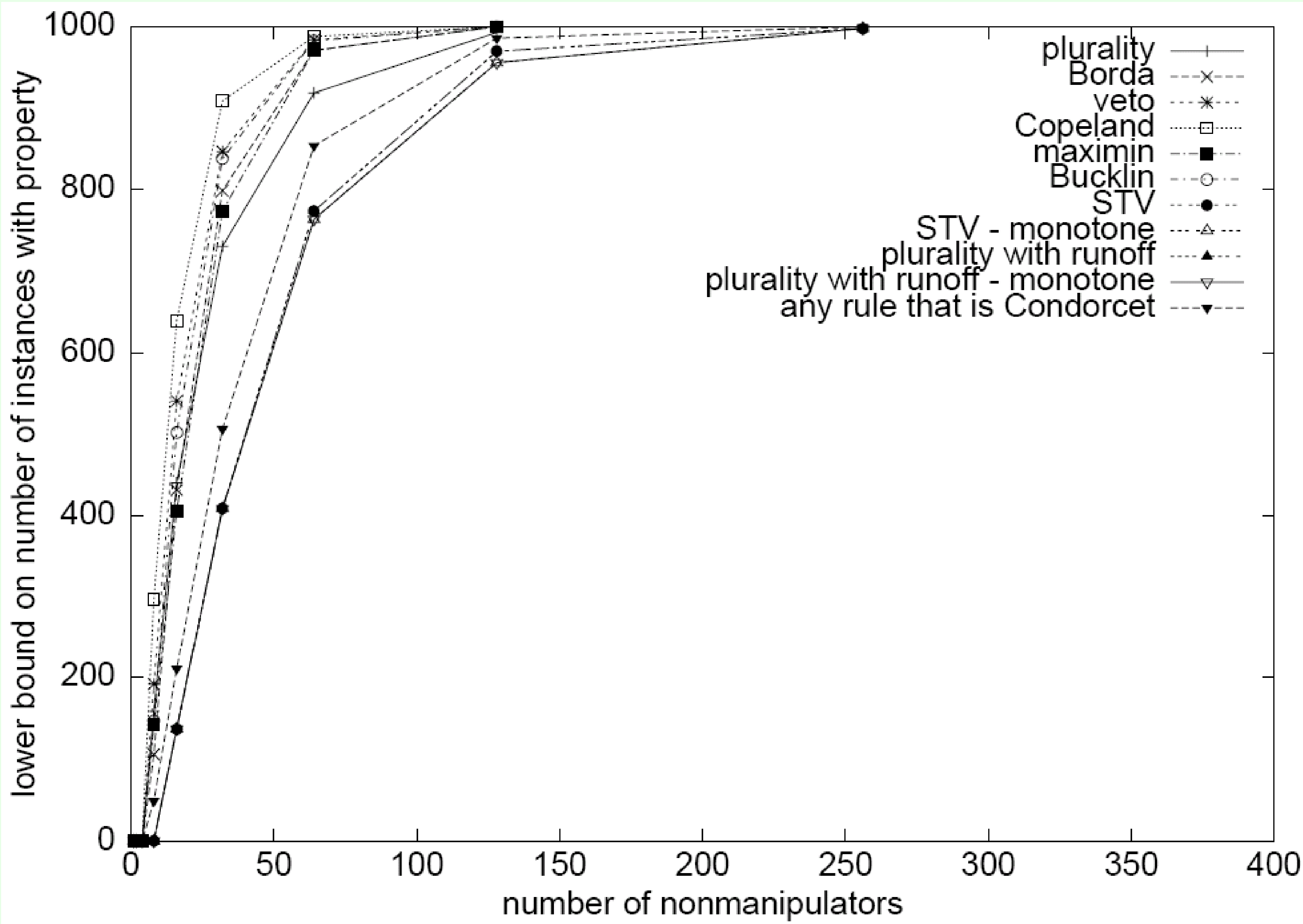
p=.6, one manipulator, 3 alternatives



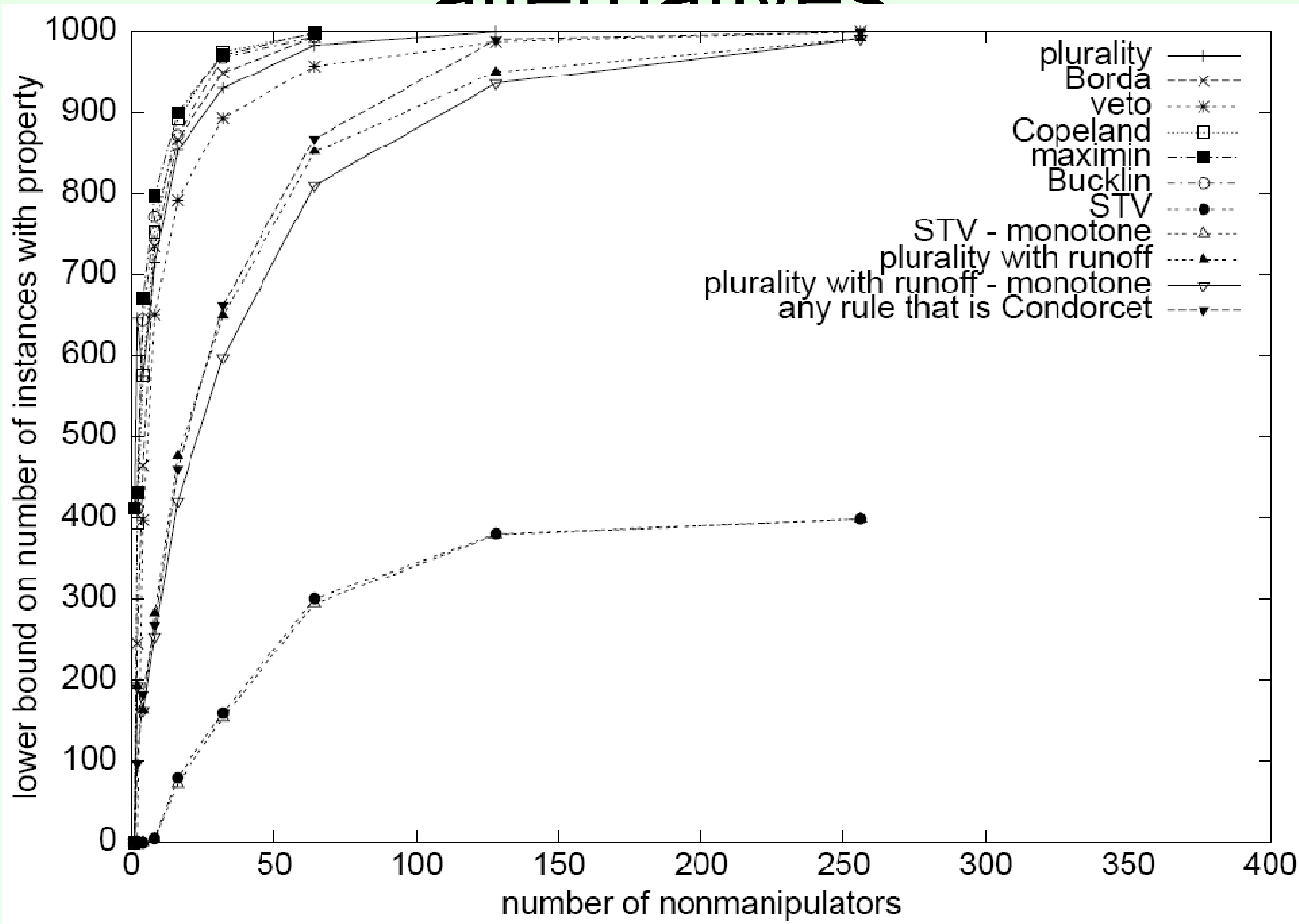
p=.5, one manipulator, 3 alternatives



p=.6, 5 manipulators, 3 alternatives



p=.6, one manipulator, 5 alternatives



Can we circumvent this impossibility result?

- Allow low-ranked alternatives to sometimes win
  - An incentive-compatible randomized rule: choose pair of alternatives at random, winner of pairwise election wins whole election
- Expand definition of voting rules
  - Banish all pivotal voters to a place where they will be unaffected by the election's result (incentive compatible)
  - Can show: half the voters can be pivotal (for any reasonable deterministic rule)
- Use voting rules that are hard to execute
  - But then, hard to use them as well…

Control problems [Bartholdi et al. 1992]

- Imagine that the chairperson of the election controls whether some alternatives participate
- Suppose there are 5 alternatives, a, b, c, d, e
- Chair controls whether c, d, e run (can choose any subset); chair wants b to win
- Rule is plurality; voters' preferences are:
- a > b > c > d > e (11 votes)
- b > a > c > d > e (10 votes)
- c > e > b > a > e (2 votes)
- d > b > a > c > e (2 votes)
- c > a > b > d > e (2 votes)
- e > a > b > c > e (2 votes)
- Can the chair make b win?
- NP-hard

many other types of control, e.g., introducing additional voters