Generalizing Effective Approval Voting Strategies to Ranked-Ballot Elections

Joel King and Rob LeGrand, Ph.D.

Abstract

Using computers to simulate many different electorates, and an algorithmic design for strategizing ballot selection, we describe certain voting strategies for *Approval* and *Hare* election methods that, under certain conditions, result in individual voters achieving more pleasing outcomes according to their personal preferences. Prior work in this area developed a strategy (named "**strategy A**") for *Approval* elections. We used this strategy as inspiration for a more generalized multi-stage strategic framework of swappable options that change the behavior of the overall voting strategy. Within this framework we confirmed that **strategy A** is the most effective strategy for *Approval* elections. We also found a different set of options that, when used to strategize in a *Hare* election results in happier individual voters.

Introduction

When voting, whether in a national-level political election or in an HOA meeting, voting for the option most likely to win that you can live with, rather than the option you'd really prefer to win, can be a common experience. This process of weighing options to vote for can involve attempting to estimate who is most likely to win and trying to estimate how to vote to make the outcome one more likely to make you happy, weighed against the risk of voting in a way, possibly one more honest/sincere about your true wishes, that doesn't impact the outcome of the election and thus risks a less pleasing outcome.

Plurality voting is surely the best known single-winner voting system: you are given a list of choices in a category, you vote for only one, and the choice with the most votes wins. Other systems, such as *approval* voting (Brams and Fishburn, 1983), can provide you with a list of choices, and you are permitted to vote for more than one option in that category. The winner is decided in the same way as *Plurality* voting: whoever has the most votes wins. If given six choices in either system, *Plurality* provides you with six valid options for how you might cast your ballot (ignoring not voting at all), whereas *approval* provides you with 62 different valid variations on how you might vote (ignoring not voting for anyone, and voting for everyone, neither of which would impact the outcome of the vote).

A common criticism of *Plurality* voting is the tendency for it to 'stabilize' to a point where only two choices are ever viable options to choose from (Duverger, 1964); votes for any choice beyond these two have been criticized as votes 'thrown away' due to their unlikely chance of influencing the outcome of the election.

Approval voting (which is one option some governments have been switching to[1] in the past few years) offers an alternative. For example, imagine you could vote for one of the two people viewed most likely to win in a *Plurality* election, but also cast votes for your two most preferred candidates, and those two may or may not be the candidate you picked out of the two favored to win in *Plurality*. In this manner, if the election still comes down to the same candidates it would have under *Plurality* voting you still can influence the outcome, but if enough other people approve of your alternative choices you stand a chance of being happier with the outcome of the election itself. Such a system of choosing who to vote for might be called a *strategy*.

In "Declared-Strategy Voting: An Instrument For Group Decision-Making" by Lorrie Cranor, an idea (called Declared Strategy Voting) is proposed in which you instead tell a system what your preferences are, design (or, perhaps more realistically, pick from known options of) an algorithm that tries to optimize the outcome of your vote for your preferences, and let the system itself decide what your ballot will be.

In "Computational Aspects of Approval Voting and Declared-Strategy Voting", a few proposals for such strategies in the context of *approval* voting are explored. One such strategy, termed **strategy A**, suggests a strategy that uses a comparison involving the numeric preferences of the voter and a numeric representation[2] of the likelihood of each of those candidates winning through a mathematical formula and numeric comparisons to generate a ballot, and describes how under a certain evaluation of the outcome of elections voted in with this strategy, this strategy is a strong choice for having pleasing outcomes for a given voter. (The above is simplifying the concept for the sake of brevity.)

However, this numeric-based system of strategizing voting, which operates on vote totals, is not directly applicable to certain other voting methods. For example, in a *Hare*[3] election candidates on a ballot are instead ranked by the voter (for example, six candidates are ranked 1 through 6, with 1 in this case representing the 'most preferred' and 6 being 'the least preferred' candidate). Every voter's ballot is then evaluated, and the candidate who has the least number of top most rankings in the entire election is eliminated from consideration. The ballots are then reconsidered under this new, smaller selection of remaining candidates. If a voter's top ranked candidate is eliminated at any one of these *Hare* rounds, the next highest ranked candidate on their ballot is considered to be the highest ranking candidate remaining on their ballot for future rounds. (You might imagine this as candidates on a ballot 'moving up' in ranking on that ballot as candidates above them on that ballot are eliminated from the election as a whole. If they have no chance to win, don't consider them, move down the list to someone who might still win.) Whichever candidate is the last candidate to remain after all other candidates have been eliminated, wins.

---

[1] Including, but not limited to, Fargo, ND in 2018, St. Louis, MO in 2020.

[2] These numeric values could be likened to a percentage of a voter's approval of that candidate, or the percentage chance for that candidate to win the election.

[3] The Hare voting system is also known by other names, including single transferable vote (STV), alternative vote, instant-runoff voting (IRV) and ranked-choice voting (RCV).

Since *Hare* does not involve a simple system of "counting up votes, see who has the most votes", this makes the numeric evaluation of **strategy A** described in "Computational Aspects" (and other numeric evaluations of other strategies) rather difficult to apply to a ranked voting system such as *Hare*. In this paper, we will explore an alternative method of defining these strategies which results in the same outcomes for *approval* and other voting systems[4], but can be applied to a broader range of election methods including ranked-ballot voting systems such as *Hare*, attempt to find strong strategies for *Hare* and other election methods, and compare the outcomes of strong voting strategies for various voting methods and how happy[5] those outcomes make voters. We may see that a particular combination of voting method and strategy results in noticeably happier outcomes over other combinations.

Additional Related Works

In "The Computational Difficulty of Manipulating an Election" (1989), Bartholdi, Tovey, and Trick find that 'second-order *Copeland*' elections are computationally difficult to manipulate. While a winner can be determined in polynomial time, determining whether or not voting insincerely will successfully manipulate the election results towards one candidate is NP-complete.

In "Single transferable vote resists strategic voting" (1991), Bartholdi and Orlin similarly find that Single Transferable Voting systems (*Hare*) have similar properties; it is computationally difficult to manipulate an election because determining whether or not you can cast an insincere vote to elect a specific candidate is NP-complete.

In "Borda rule, *Copeland* method and strategic manipulation" (2001), Favardin, Lepelley, and Serais compare *Copeland* and *Borda* elections (two election methods that involve ranked ballots) and find results that suggest that *Borda* elections may be more commonly manipulable than *Copeland*.

In "Strategic voting and nomination" (2013), Green-Armytage explores various methods of voting insincerely, such as ranking a less preferred candidate over a more preferred candidate, in order to increase the likelihood that a third candidate preferred more than the first two might win ("burying") or might work with other voters to vote for a compromise candidate they can all agree on rather than voting sincerely. They found that *Hare* was least susceptible to manipulation, with *Borda*, *Coombs*, *Approval*, and *Range* to be most susceptible.

Setup

We use C++ to define an election object with a predefined number of candidates running for a single position. The election method, such as *Plurality*, *Approval*, *Borda*, etc, would also be pre-set for this election object. Inside this object we can also track a collection of ballots cast in

[4] We are specifically limiting ourselves to single-winner systems/elections only.
[5] For a specifically defined version of happiness based on voter preferences for the winning candidate.

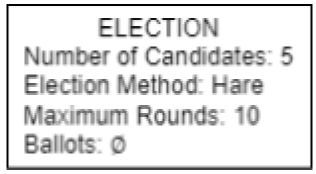

Figure 1-1.

its election, and these ballots can be queried by a group of voters so that they can see the current state of the election in a manner of interpretation of their choosing[6].

In addition to the election object being configured with the election method, the number of candidates within that election, and a storage space for ballots, a maximum number of rounds per election is chosen. In order to give earlier voters and later voters roughly equal power to affect the outcome of the election, and also to provide that first voter with the opportunity to vote strategically based on candidate performance within the election, elections would call for multiple rounds of voting from a group of voters (very much like DSV's ballot-by-ballot mode in Cranor, 1996). In the first round, voters cast their ballots based on their selected strategy, and those ballots are stored within the election object. In subsequent rounds, every voter is guaranteed a chance to change their existing ballot, based on their selected strategy's interpretation of the state of the election, based on currently existing ballots.

If the quality of ballots[7] within an election do not change from one round to the next, the election is considered 'stable', and we can avoid running subsequent rounds within that election, as the voters have stopped changing their ballots based on their preferences and selected strategies. The need for a maximum number of rounds in an election is due to the ability for certain combinations of voter preferences, voting methods, and voting strategies to result in ballots constantly changing from round to round, as updated ballots result in other voters updating their ballots in a constant, unending struggle between voters trying to find a 'better' ballot based on the new election conditions. Without a maximum limit of rounds, the voters might continue changing ballots (and possibly the resulting winner of the election) forever.

In addition to election objects, voter objects can be generated. Each voter is assigned a randomized value (in our code, we generated numbers between 0 and 100, inclusive) per candidate to represent that voter's opinion of that candidate. The higher the value, the higher the approval of that candidate by that voter. In addition, each voter is assigned a strategy which

---

[6] Relevant in Stage 1 of Voting Strategies, described later

[7] For example, in an *Approval* election, while the internal representation of a ballot might change from candidates C, A, and B being approved of (in that order) to B, A, and C being approved of (in that order), the overall ballot remains the same: A, B, and C are the only candidates approved of by that voter from round to round. This is one situation in which the ballot *itself* may change (the order in which candidates are approved of) but the *quality* does not (the same candidates are approved each round). In a ranked voting election (such as *Hare*), the order is important: if order changes, the quality of the ballot has changed, and the rounds are not yet stable.

defines how they decide to vote within any election.

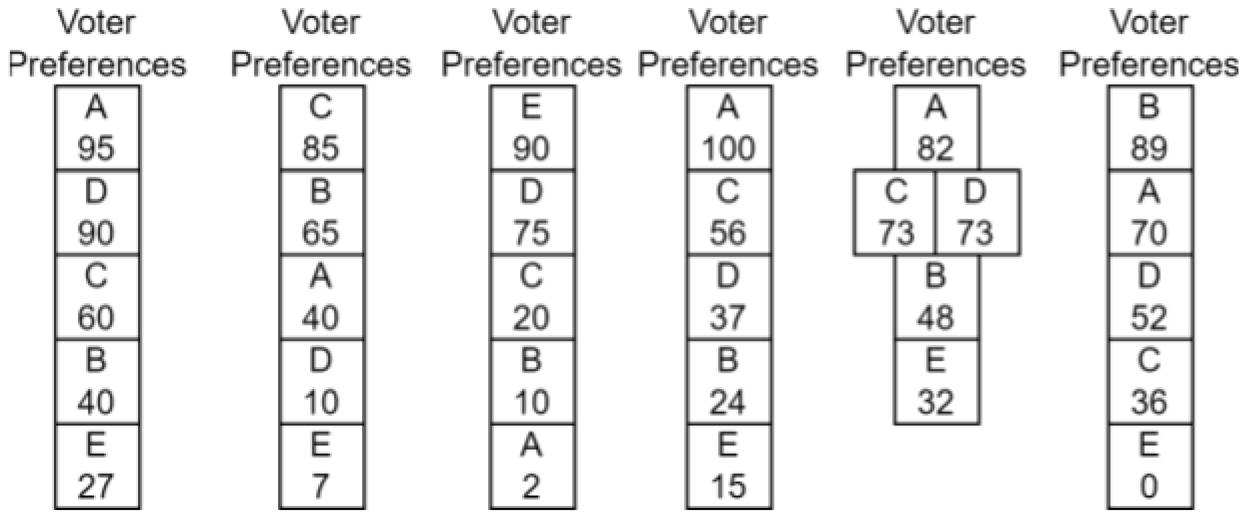

Figure 1-2. An example of six voters' preferences.

Voters, or even groups of voters, are not tied to a specific election. Only defined candidate counts need to match between voters and elections. This means that any group of voters with a set of (for example) six candidate preferences could participate in any election that is set up with six candidates. This means that a specific group of voters can participate in a *Plurality* election, a *Borda* election, an *Approval* election, and other types of elections, all with the same candidate preferences (reflecting how a person's opinions on topics or of people typically won't change simply because the election method changes). Strategies for each voter can be altered or adjusted as desired in order to test how effective certain combinations of strategy stages might work within different election methods. For example, a strategy that performs well within an *Approval* election might perform poorly (and result in a less favored candidate winning) in a *Plurality* election.

This flexibility means that different voting strategies within the same election method can be compared against each other, and strong strategies discovered this way for one election method can be compared to strong strategies for another election method.

Voting Strategies Overview

Each voter's assigned voting strategy is defined in three parts, and each part defines how a sequential stage of their decision making operates. Each stage is described in more detail below, but we will briefly summarize them here:

- The first stage defines how they would interpret the current state of the ballots currently cast in an election, typically in the context of *an* election method[8] (though not necessarily tied to

[8] such as, but not limited to, *Plurality*, *Borda*, the speed candidates are eliminated in *Hare*, or even an option to decide to ignore the candidate's current performance altogether (which results in a voter voting their preferences with no additional strategizing

the election method of the election currently being evaluated). This attempts to reflect how a voter might be aware of current polls by news organizations, or a general feel for the opinions within their community.

- The second stage organizes the order in which they would make pairwise comparisons between all the candidates in an election, based on the performance information provided to them by the first stage.
- The third stage defines whether that particular voter would explicitly build a ballot from the top, from the bottom, or both. These choices are made in the order defined by the second stage. Once a voter makes a decision about where a candidate belongs on a ballot, that decision does not change during that specific ballot's construction.

Stage 1: Checking the Polls

The first stage of any voter's strategy requests the current election's 'state' in some form, typically in the context of an election method. For simple counting-style election methods such as *Plurality* or *Approval*, the voter receives a set of values representing the count of votes per candidate. If Candidate A has three votes, Candidate B has one vote, and Candidate C has zero votes in an *approval* election, the values returned are a simple {3, 1, 0}.

For election systems that are not based on simple counting to determine an outcome, the voter requests this outcome information to be presented in one of two ways:

- how well individual candidates do when running solely against the winner (as calculated with currently cast ballots), hereafter referred to as a pairwise comparison. A candidate with more people voting for them over the current winner results in a higher score.
- how quickly the candidates are eliminated from a ranked-ballot election process (again, with currently cast ballots), which we refer to as their 'elimination speed'. Longer lasting candidates are assigned higher scores.

In either option, the highest score possible for that option is always assigned to the winner of the election.

In every election, the very first voter in the very first round of voting receives no data about the likelihood of any candidate winning, since no votes have yet been cast. In these cases, only the third stage has an impact on how the voter votes due to the design of the second and third stages, and their cast ballot is a wholly sincere reflection of that voter's actual preferences, which is ideal strategy when the voter has zero information. So long as more than one round of voting occurs within an election object, the first voter still has an opportunity to utilize a voting strategy in an informed manner in rounds following the first (*i.e.*, they could utilize the first-stage strategy above based on other voter's ballots rather than blindly voting sincerely like the first voter does in the first round).

We had several defined settings for this stage. Some options are defined below, either to aid in explaining this stage, or due to their relevance in other parts of this paper:

- **allZero** simply returns empty polling data. As a result, stage 2 provides no sorting or gating, and the voter's preferences are the only determining factor in stage 3. This was the equivalent of us telling a voter to simply vote their own preferences without any regard to the state of the election, opinion polls, etc.

- **approval** returns polling data where above average rankings on a ballot are counted as a vote, to simulate *Approval* elections.

- **bordaEval** sums up ballots as if they were cast in a *Borda* election (Saari, 1995).

- **copeland** returns a candidate-by-candidate evaluation where a candidate scores higher if they win more one-on-one matchups against each of the other candidates.

- **simpson** returns a candidate-by-candidate evaluation where a candidate is scored by the smallest number of votes it received in any pairwise matchup against all the other candidates.

- **hareEliminationSpeed** returns an evaluation of the candidates where the election's ballots are processed as if the election is a *Hare* election, and candidates are ranked by how quickly they are eliminated in the *Hare* process, with the longest lasting candidate receiving the 'best' place.

- **pairwise** calculates the winner of the election based on the election's assigned type. This winner would be rated the highest within the returned poll. All other candidates are ranked below this winner in an order determined by how many people voted for that candidate above the winner of the election as it currently stands. This comparison method was intended for ranked-ballot elections, such as *Hare*.

- **victoryPairwise**, which calculates identically to **pairwise** for both the current winner and anyone who can beat the winner in a one-on-one matchup, but sets anyone who can't beat the current winner in a one-on-one **pairwise** matchup to all be tied at last place.

- **winOrLosePairwise** is another variant of **pairwise**. The winning candidate of the election type was still ranked higher than all other candidates. Instead of strictly ordering the non-winning candidates, it would tie all candidates who could beat the winner if those were the only two candidates to choose from at a level below the winner, and all candidates who could not beat the winning candidate at another level below that. This permitted voters to see who was projected to win, who might win if given a chance, who was less likely to win, while the ties at the lower levels allowed the voter to express their preferences within those groups at stage 3.

Stage 2: Choosing Among Pairs of Candidates

The second stage of the voter's strategy works through the list of candidates provided by the first stage, ordered by their performance in that stage, and is paired up with each other in every possible combination, in a sorted manner chosen by this stage. In each pairing, we considered three options:

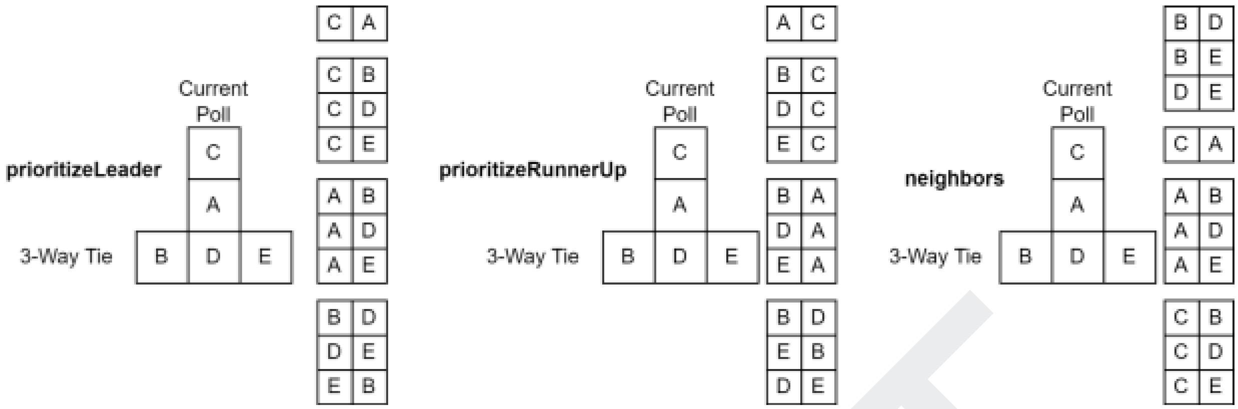

Figure 2-1. Note that horizontal order within a pair has no impact on strategy. Only vertical position within chosen pairs, and the grouping of pairs impacts stage 3 evaluation.

- **prioritizeLeader**, where the stronger of the two candidates being considered is the principal determiner of evaluation order, with the weaker candidate breaking ties,
- **prioritizeRunnerUp**, where the weaker candidate determines the evaluation order, with the stronger candidate breaking ties,
- **neighbors**, where pairs are grouped according to how closely they performed with respect to each other. In those pairs the overall performance (within the election so far) of the stronger candidate within that pairing breaks any distance ties between pairs. This option was inspired by the moving-average strategy suggested by Smith (2000).

These pairs are grouped together if they performed equally well with each other in the election, and stage 3 evaluates these groupings in order from highest to lowest as a set before moving on to the next grouping.

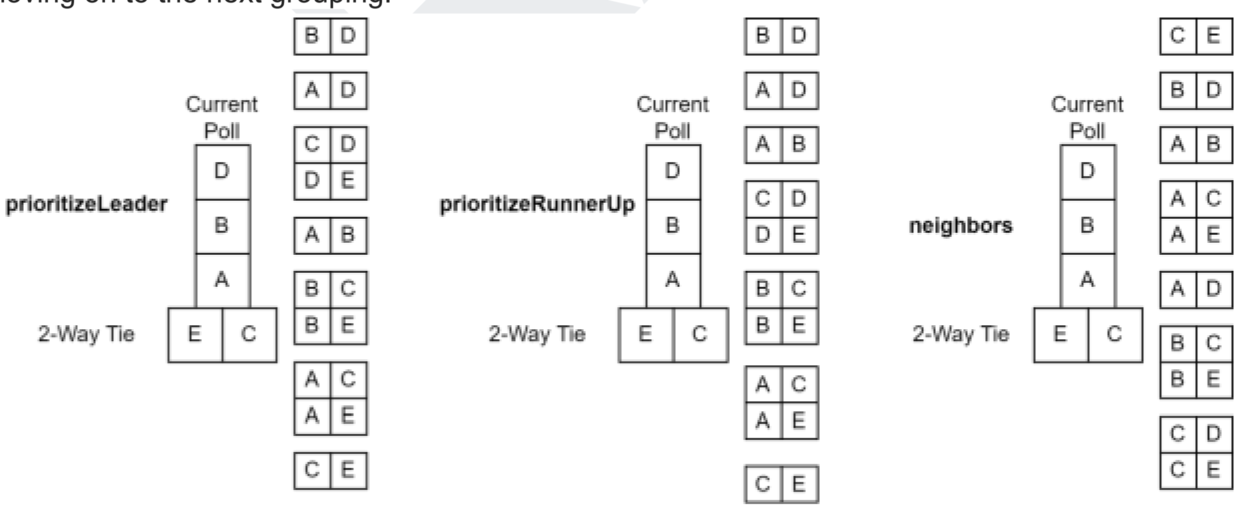

Figure 2-2. The result of prioritizeLeader here will result in stage 3 evalutating B and D before moving on to evaluate A and D, followed by evaluating C, D, and E.

Stage 3: The Voter Weighs In

The third stage of a voter's strategy is set to one of three options:

- **top**, which goes through candidates and decide who to approve (give the highest still-available rating on the ballot), if any qualify,
- **bottom**, which goes through candidates and decide who to disapprove (give the lowest still-available rating on the ballot), if any qualify, or
- **bothEnds**, which performs both **top** and **bottom** as appropriate.

First, the voter evaluates all candidates within a grouping set up by stage 2 that remain 'undecided' on their ballot (*i.e.*, no approval/disapproval decision has yet been made for that candidate by this voter). The voter's preference for each candidate considered is compared to the voter's preferences about other candidates matched against the candidate being considered. If the voter's preference for that candidate is higher than the average preference of the candidates it is being compared against *and* it is using a third-stage strategy that approves candidates (as opposed to using one that disapproves of candidates only), the voter marks that candidate as one to approve. If the candidate is less-preferred than the average of the candidates it is being compared against *and* the chosen third-stage strategy permits disapproval, the candidate is marked for disapproval.

If the preference of the candidate being considered exactly matches the voter's average preferences for the other candidates being compared against, no approval or disapproval is

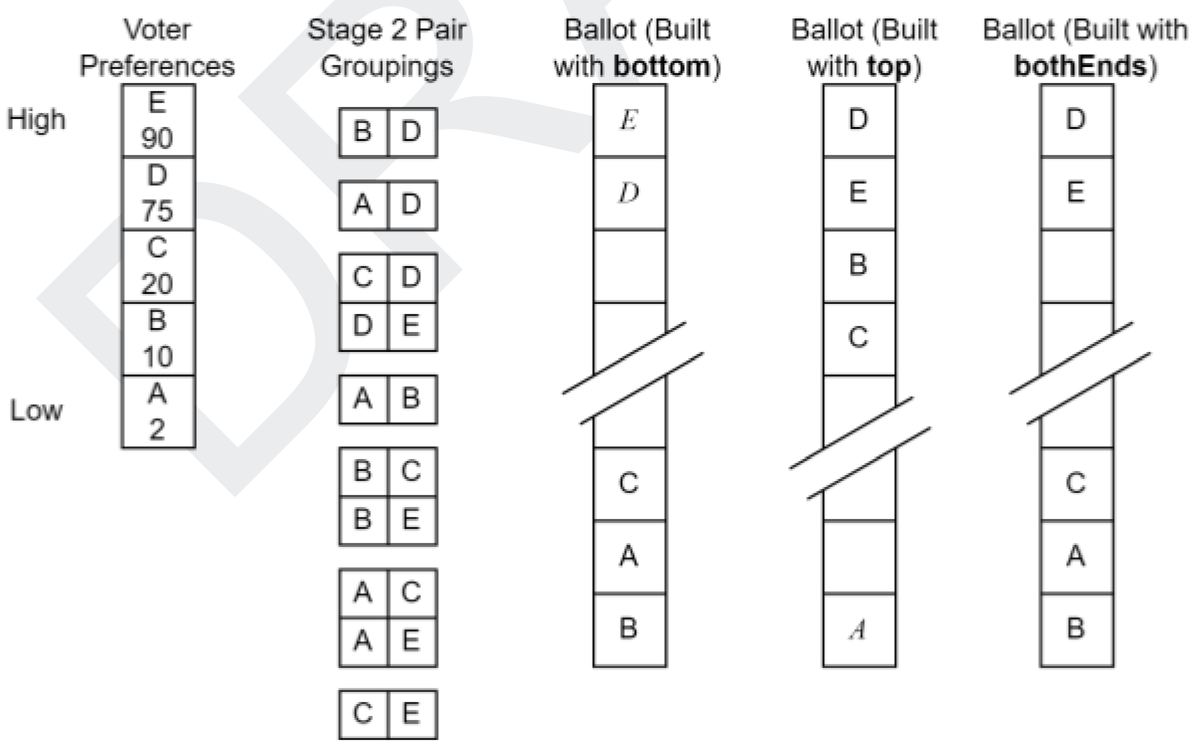

Figure 3-1. Despite preferring E over D, D is placed above E on ballots where strategy happens at the top of the ballot because D is seen before E when going through stage 2 pairings. Note that while C is a 'middle of the road' candidate in the voter's overall opinion, it opts to disapprove of that candidate when allowed to. When this decision is made, it is only evaluated in the context of the grouping it is in, {CD, DE}, not the voter's overall preferences. C is rated worse than the average of C, D and E, earning it disapproval. See Figure 3-2 for an explanation of E, D, and A in **bottom**/**top**.

assigned, leaving the candidate open to being assigned either category in subsequent evaluations within poorer performing candidate pair groups.

Then, candidates that have been marked for approval or disapproval but have not yet been marked in a position on the voter's ballot this round are sorted on to the voter's ballot in an

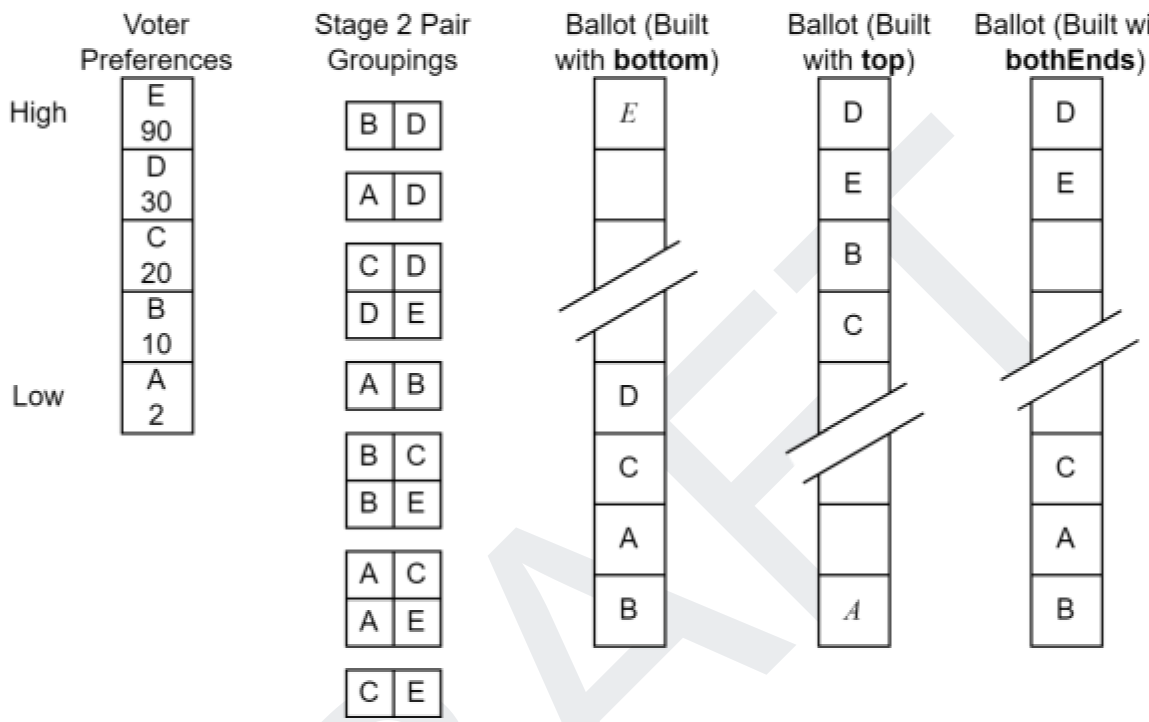

Figure 3-2. Compare to 3-1. Note in this example only the cardinal preference for D has changed, which has changed the outcome of **bottom**. In this instance, both D and C are placed when CD-DE is evaluated. Italicized letters (*E* in **bottom**, *A* in **top**) represent examples of candidates who were not assigned to the ballot after all pairs from Stage 2 had been processed by Stage 3 (as those candidates did not qualify for the **top** or **bottom** of the ballot).

order respecting the voter's preferences.

As mentioned earlier, any candidate placed on the ballot at this point will not be reevaluated when looking at any future pairings, whether within the same group of equally performing pairs (as the outcome would remain the same) or weaker performing pair groupings.

Stage three repeats with the next grouping of candidate pairs who performed more poorly, until all such groups have been evaluated, or all candidates have been placed on the ballot.

With **top** or **bottom**, and in certain rare cases[9] of **bothEnds**, a few high or low ratings are assigned after processing all stage 2 pair groupings, as a few candidates may still be unclassified as approved or disapproved. In the cases where a candidate is not placed on the

[9] For example, a voter's preference for a candidate might exactly match all of the voter's average preferences for all of the other candidates that candidate was compared against in stage 2.

ballot by the time all pairwise match-ups are evaluated, those remaining uncategorized candidates are assigned values based on the third-stage strategy chosen. If the third-stage strategy selected was **bothEnds**, uncategorized candidates are assigned a 'middle position'. If the third-stage strategy the voter utilizes was **top**, remaining candidates would all be marked for disapproval, and when utilizing **bottom**, uncategorized candidates are marked for approval.

One Complete Example

Here is one very simple example featuring three voters voting in an approval election with four candidates. In this example, a maximum of three rounds of voting are allowed.

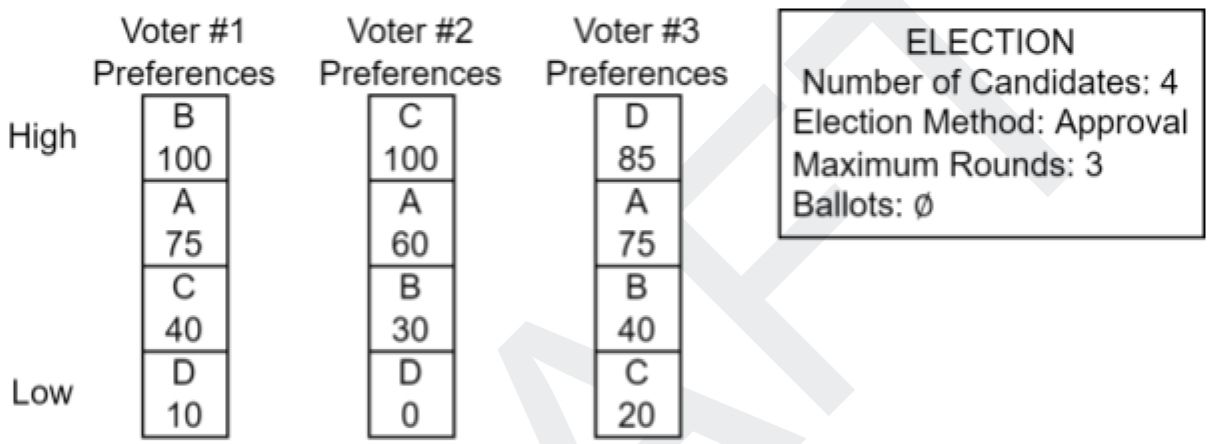

Figure 4-1. A very simplified election example.

All voters are set to strategize with **approval** for stage 1, **prioritizeLeader** for stage 2, and **bothEnds** for stage 3. Voters vote in order in a cycle until they stop changing their ballots or the maximum number of rounds is reached.

Round 1: Voter 1

Even though voter 1 is set to **approval** in stage 1, they receive polling data for the state of the election that shows all candidates with zero votes, because that is the reality of the election so far. This means that stage 2 groups all pairs of candidates together, so all candidates are options for the very first step of stage 3 evaluation, so the voter votes in order of their preference.

The way this works is as follows:

1. Candidate B is evaluated, as it is the most preferred candidate the voter has. Candidate B is in this grouping of pairs (as all pairs are in a single group in this initial vote for voter 1), and it rates better (100) in the eyes of the voter than the average of all candidates it is paired against in this group ((75+40+10)/3 = 41.666…), so it is marked for *approval*.
2. Candidate A is evaluated. The voter's preference for candidate A rates better with a preference value of 75 than the average of all candidates it is compared against (B, C, and D, with an average of 50), so it is also marked for *approval*.

3. Candidate C is evaluated. The voter's preference for candidate C (40) rates worse than the average preference values for candidates it is paired against in this group (A, B, and D, with an average of about 61.66), so it is marked for *disapproval*.

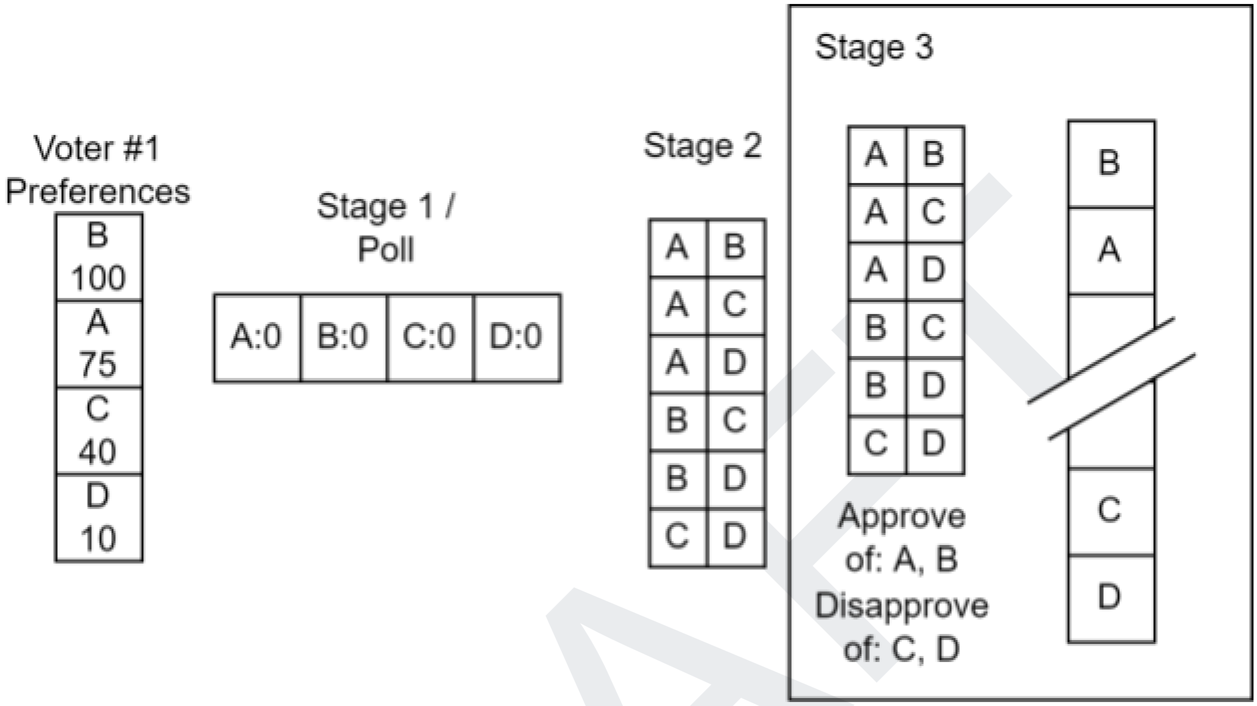

Figure 4-2.

4. Candidate D is evaluated. The voter's preference for candidate D (10) rates worse than the average preference values for candidates it is paired against in this group, so it is marked for *disapproval*.

All candidates marked for approval are placed at the top still-available values of the ballot, in order of the voter's preferences.

All candidates marked for disapproval are placed at the bottom still-available values of the ballot, in order of the voter's preferences.

All candidates have been assigned a position, so voter 3 stops evaluating its strategy, and turns in the ballot as completed. A ballot has now been cast with one vote for A, and one vote for B.

Round 1: Voter 2

Then, voter 2 steps up, and receives the current vote totals for its stage 1. (See Figure 4-3)

Voter 2's stage 2 evaluates that

- A and B are both equally likely to win the election if they were only running against each other as they have one vote apiece.
- A vs C, A vs D, B vs C, and B vs D are all four equally successful pairings, with one candidate having one vote, and one candidate having zero, and
- C vs D is the last and least successful pair (so far) out of the six possible pairs of candidates, with both candidates having zero votes.

Stage 3 runs through the candidates in this voter's preferred order (C, then A, then B, then D) and compares them to the groups of stage 2 pairs in the order presented above. One group of pairs is evaluated before moving on to the next group of pairs.

1. The voter looks at candidate C. As candidate C is not present in the first group (composed of only the AB pair), candidate C is not evaluated at this time. (Note that this is true even though voter 2 values C at the highest possible value, 100, because C is not in this group of pairs from stage 2.)

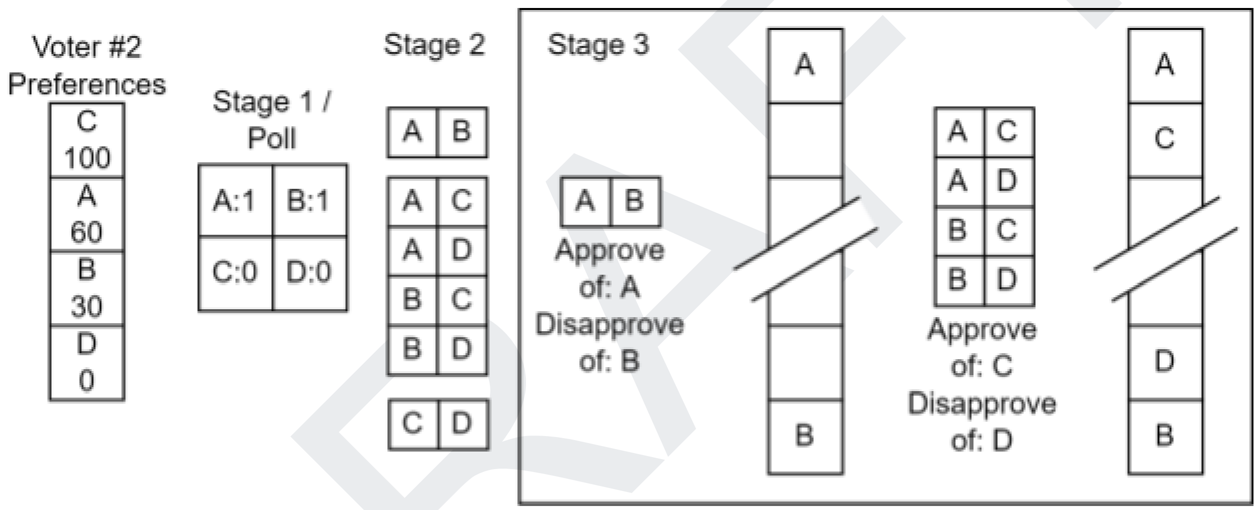

Figure 4-3.

2. The voter then looks at candidate A. Candidate A is in the group (specifically the AB pair). A is rated more highly than the average preference score of candidates it is competing against in this group, so A is marked for *approval*.
3. Candidate B is now considered, as it is third in the list of the voter's preferences. Candidate B is in the AB pair, and it is preferred less than the average of all candidates it is competing against in this group, so it is marked for *disapproval*.
4. Candidate D is then considered. D is not present in this grouping of pairs from stage 2, so candidate D is not assigned a position.

Now that all candidates have been considered against this grouping, candidates marked for approval (candidate A) are assigned the highest slots available on the ballot, in order of voter preference, and candidates marked for disapproval (candidate B) are assigned the lowest available values on the ballot, in order of voter preference.

The next group of pairs (AC, AD, BC, BD) are now considered.

1. Candidate C is considered. Candidate C is in this group of pairs, so the voter's valuation of C (100) is compared to the average of all candidates it competes against within this group (A vs C, B vs C). C is valued higher than the average of these other candidates, so it is marked for *approval*.
2. Candidate A is considered. Candidate A has already been assigned a position on the ballot, so it is left alone.
3. Candidate B is considered. Candidate B has already been assigned a position on the ballot, so it is left alone.
4. Candidate D is considered. Candidate D is in this group of pairs, so the voter's valuation of D is compared to the average of all pairs it exists within in this group (AD, BD). D is valued lower than the average of the candidates it's compared against, so it is marked for *disapproval*.

Now that all candidates have been considered against this grouping, candidates marked for approval (candidate C) are assigned the highest slots available on the ballot, in order of voter preference, and candidates marked for disapproval (candidate D) are assigned the lowest available values on the ballot, in order of voter preference.

All candidates have now been assigned, so voter 2 turns in their ballot to the election. The last pair grouping from stage 2 (containing CD) is unneeded and ignored, as even if all candidates were to be evaluated for this grouping, all candidates have been assigned places on the ballot, so all candidates would be skipped.

Round 1: Voter 3

The vote is now 2 for A, 1 for B, 1 for C, and 0 for D. Voter 3 receives this information as a representation of the total counts of votes in this approval election, and separates the candidate pairs according to its assigned **prioritizeLeader** strategy.

The first grouping of candidate pairs are AB and AC, as those are the pairs that contain one candidate with two votes, and one candidate with one vote. No other pairing performs better than this, currently.

Voter 3 now considers the candidates in the order of its preferences with the first pair group.

1. Candidate D (the most preferred candidate for Voter 3) is considered. Candidate D is not present in the grouping of pairs (AB, AC), and is ignored for now.
2. Candidate A is considered. Candidate A is present in this grouping of pairs, and is compared against the average of the voter's preferences of the other candidates it is competing with. Candidate A rates higher than the average of B and C, so it is marked for *approval*.
3. Candidate B is considered. Candidate B is present in this grouping of pairs, and is compared against the average of the voter's preferences of the other candidates it is competing with (only candidate A). Candidate B is less preferred than all these candidates (just candidate A in this case) and is marked for *disapproval*.

4. Candidate C is considered. Candidate C is present in this grouping of pairs, and is compared against the average of the voter's preferences of the other candidates it is competing with (only A). Candidate C is less preferred than candidate A, and is marked for *disapproval*.

Note here that one candidate has been marked for approval, and two candidates have been marked for disapproval. These candidates are placed in order of the voter's preferences at their appropriate next available ratings on the ballot (see Figure 4-4).

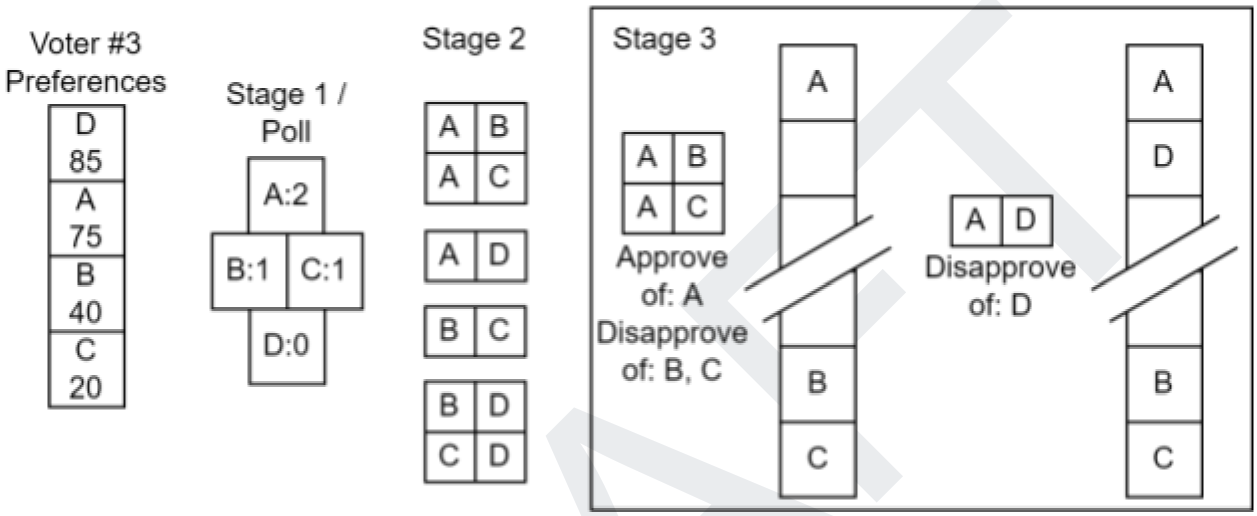

Figure 4-4.

The next grouping of candidate pairs is considered. This grouping only contains one pair: AD.

1. Candidate D is considered. Candidate D is present in this grouping of pairs, and is considered against the average of the voter's preferences of all other competing candidates within this group (just A in this case). D is preferred over A, so candidate D is marked for *approval*.
2. All other candidates are considered in order of the voter's preferences (A, B, C), but have all been assigned values on the ballot, and are thus all skipped.

The remaining candidates marked for approval are placed appropriately on to the ballot, just below prior approved candidates. No candidates are marked for disapproval.

All candidates have been assigned ratings on the ballot, so the ballot is turned in, and this first round of voting is now over.

ELECTION	Ballots:	A	B	C	D
Number of Candidates: 4		1	1	0	0
Election Method: Approval		1	0	1	0
Maximum Rounds: 3		1	0	0	1
Rounds Finished: 1					

Figure 4-5.

Round 2: Voter 1

This time around, voter 1 has ballot information to use for strategizing. The stage 2 pairings **prioritizeLeader**, so the pairs involving A are grouped before the pairs that do not involve A. Candidates are evaluated in order of the voter's preference (B, A, C, D).

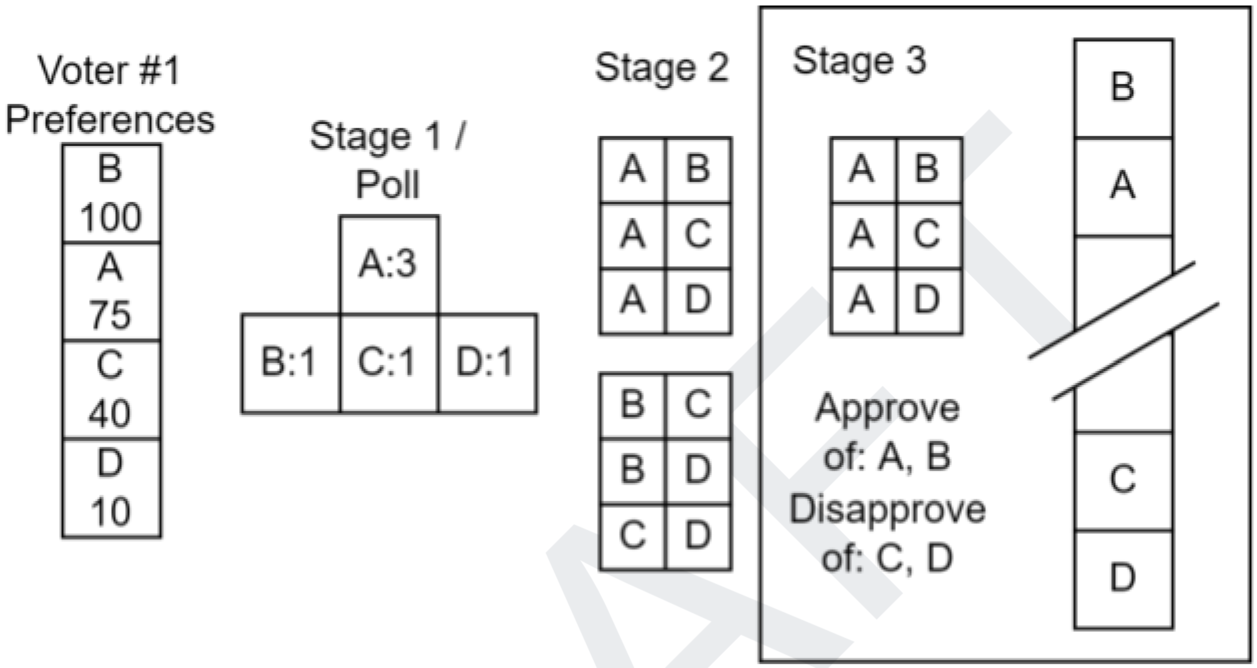

Figure 4-6.

This process results in the same effective ballot in this round.

Round 2: Voter 2

In round 2, voter 2 has the same poll and pair grouping as voter 1 had most recently, for the same reasons, so we refer the reader back to the explanation there for how those stages behave.

It is important to point out, however, that because voter 2 here has different poll and grouping information, the ballot is built in a different *order*.

- Round 1: Voter 2's ballot: A>C>D>B
- Round 2: Voter 2's ballot: C>A>B>D

Despite voter 2's preferences remaining the same, changes within the stage 1 polling data and resultant stage 2 groupings of pairs results in different candidates being able to be assigned an *approval* or *disapproval* in a different order.

In the case of an *Approval* election method, this difference is immaterial: both A and C are approved by voter 2, while D and B are not. However, in our code, the numeric *rating* value for positions on the ballot is information saved inside the ballot, for flexibility between election

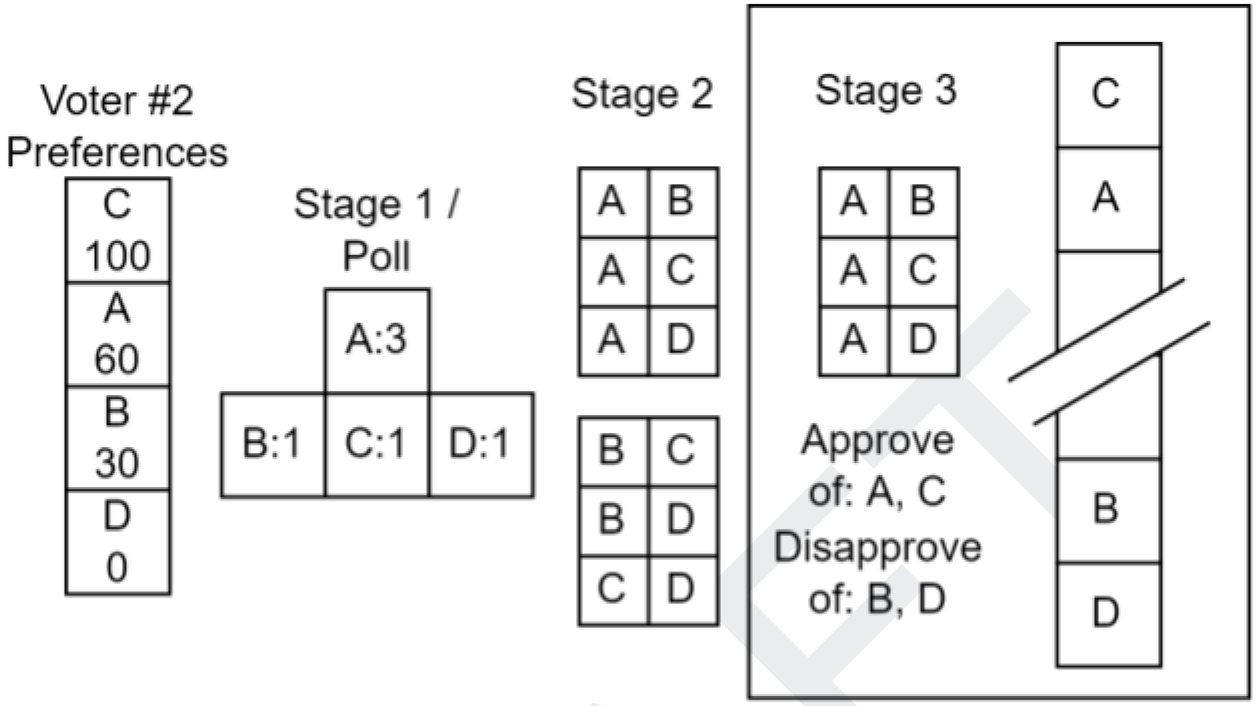

Figure 4-7.

methods. When determining the *stability* of a round, however, the ballot is only evaluated in the context of *that system* the ballot is for. In this case the ballot is for an Approval election, and so it is the "same" ballot as from round 1. If this were a Hare or other ranked-ballot system, the change in the candidate order on the ballot would be considered a change that destabilizes the round (because it's a different ballot, even though it might not change the outcome of the vote) and so it would trigger the requirement of an additional round of voting (so long as the maximum number of rounds had not been reached), to enable other voters to strategize with the new ballot information.

Round 2: Voter 3

Similarly, voter 3 *approves* and *disapproves* of the same voters, but in a different order. (Round 1: A>D>B>C, Round 2: D>A>B>C). Despite this ordering change, the resultant ballot when evaluated in the context of an Approval election method is the same, and so the ballot is the same as the last ballot this voter cast.

Since all ballots cast in round 2 were *functionally* identical to the ballots cast in round 1 in the context of an *Approval* election, the election is considered to have stabilized, and no more rounds are run, since the voters would simply repeat these identical evaluations and resultant ballots in subsequent rounds.

The winner of the election is A, with three votes to B, C, and D's one vote each. (This is also the Condorcet winner.)

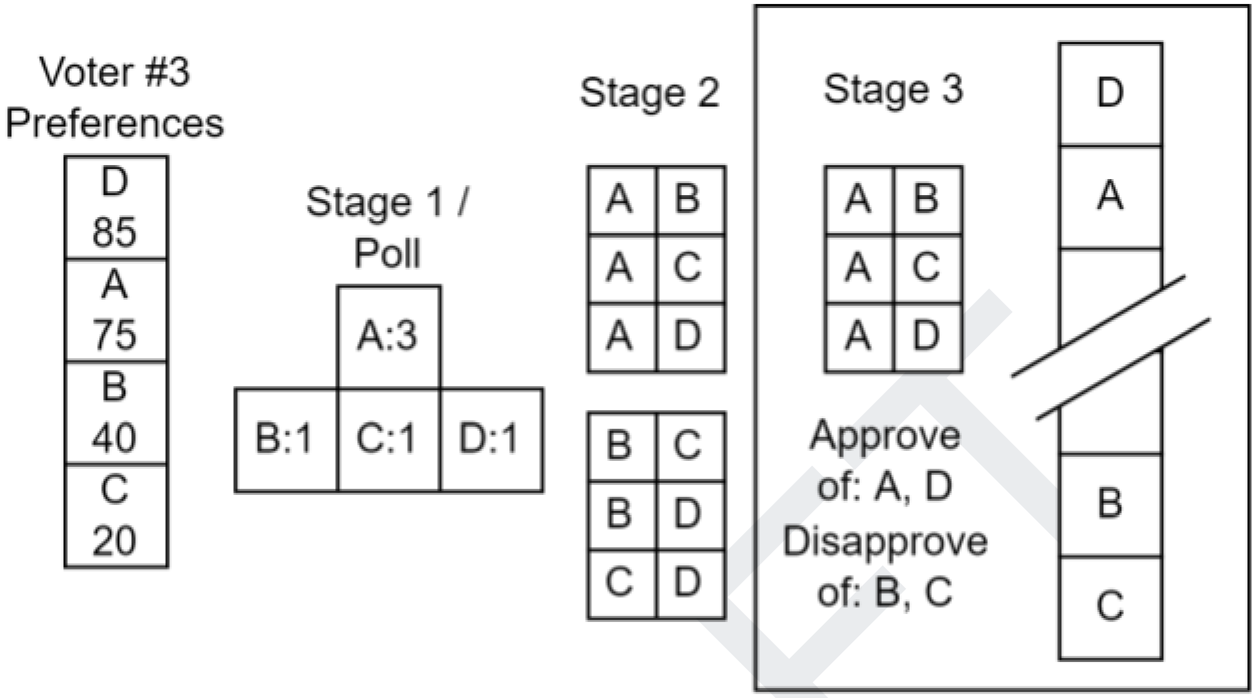

Figure 4-8.

Experiments

For our experiments, we wanted to run a large number of randomly generated groups of voters through many different pairs of strategy combinations and attempt to identify the strategy combination that seemed to result in the happiest overall population. This could be described as similar to a Monte-Carlo simulation. We wanted to test both what would occur if the group was voting with one strategy and what would happen as we gradually changed the strategy of an entire group from one strategy to a second strategy.

We set every voter's strategy to one strategy (X) and then methodically changed the strategy of one voter at a time to a second strategy (Y). To put it another way, if every voter started with X, we would change one voter to Y, measure the outcome, then change another voter to Y before measuring the outcome, repeating the process until every voter had been changed over to Y. For each voter changing strategy, we measured how much happiness that voter gained or lost from the change.

For every pair of strategies we compared, we would compare them in both directions: if we started with X and switched to Y, then we ran the same group of voters with everyone starting with Y and switching them to X.

Initially, we ran 1000 electorates, or groups of voters, with 30 voters each, through both a set of *Approval* elections and *Hare* elections. Each election had six candidates. We ran **allZero** against all other possible strategy combinations. The strategy combinations that had positive outcomes, we then ran against each other in an effort to narrow down a 'best' strategy for *Approval* elections, and a 'best' strategy for *Hare* elections.

We used the results from the above strategy to inform a similar search with a set of 1000 electorates of 30 voters each in elections featuring 6 candidates, this time where the random number generator had been seeded with a constant, preset value, to ensure that all experiments used the exact same set of 30,000 random voters, arranged in exactly the same groups. This search, due to limited time, was more narrowly focused on 'adjacent' strategies to an initial starter strategy in a hill-climbing effort to find the best overall strategy (Russell and Norvig).

With *Approval* elections, we started with **allZero/prioritizeLeader/bothEnds** (equivalent to **strategy A**) as our initial starter strategy, and explored all strategies that involved changing only one element of that three-part strategy. For stage 1, we explored **approval**, **victoryPairwise**, and **winOrLosePairwise**, (as no other stage 1 strategy made sense for an *Approval* election, based on earlier searches) each with the **prioritizeLeader/bothEnds** combo for stages 2 and 3. For stage 2 we explored **neighbor** and **prioritizeRunnerUp**, each with **allZero/bothEnds** for stages 1 and 3. In stage 3, we explored **top** and **bottom**, each with **allZero/prioritizeLeader** for stages 1 and 2.

With *Hare* elections, we started with **allZero/prioritizeLeader/front**. We explored adjacent strategies in the same way as we described above. In stage 1 we explored **bordaEval**, **pairwise**, **hareScore**, **copelandScore**, **simpsonScore**, **victoryPairwise**, and **winOrLosePairwise**. In stage 2 we explored **prioritizeRunnerUp**, and **neighbor**. In stage 3 we explored **bothEnds** and **bottom**.

Results

Figure 5-1 shows the results from one part of our hill-climbing search for an *Approval* election, starting from **allZero/prioritizeLeader/bothEnds** and changing only one stage at a time to explore neighbors.

Here we see that changing stage 2 when voting **allZero** results in exactly the same outcome, and that **approval**, **victoryPairwise**, and **winOrLosePairwise** being used in stage 1 result in happier voters when those voters change that aspect of their strategy.

Figure 5-2 shows the total amount of happiness changed for the individual 30,000 voters when each changes their strategy from our **Strategy A** equivalent strategy of **approval**/**prioritizeLeader**/**bothEnds** to adjacent strategies, where each stage is changed to the listed alternative.

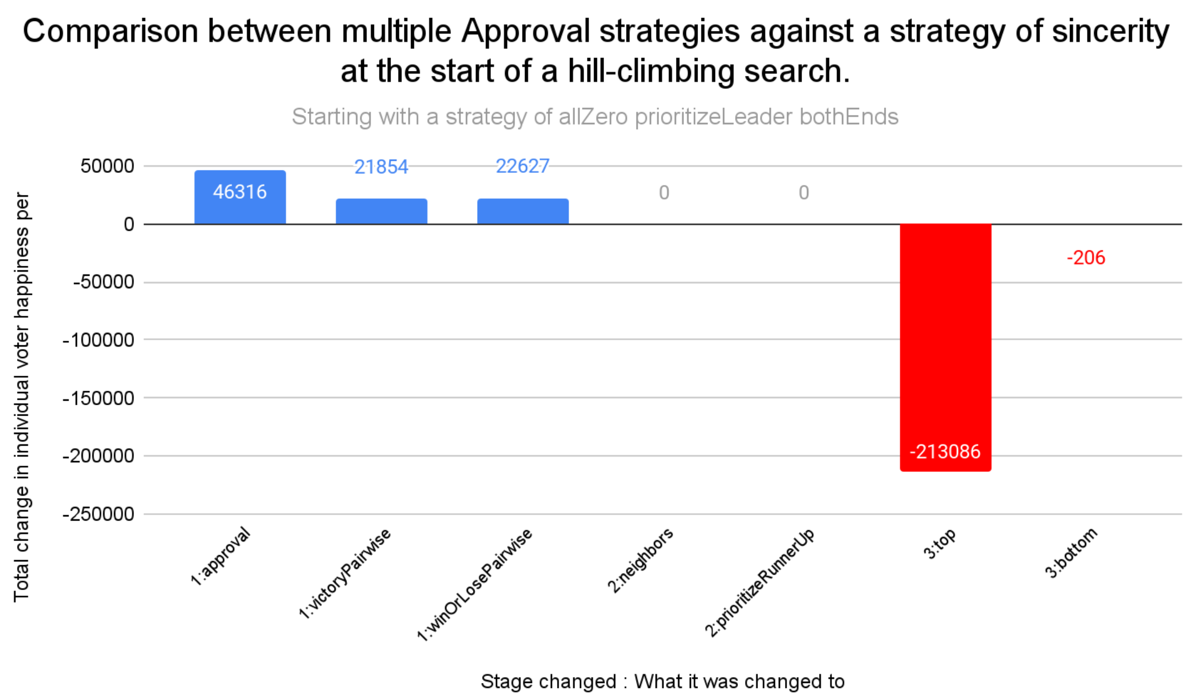

Figure 5-1.

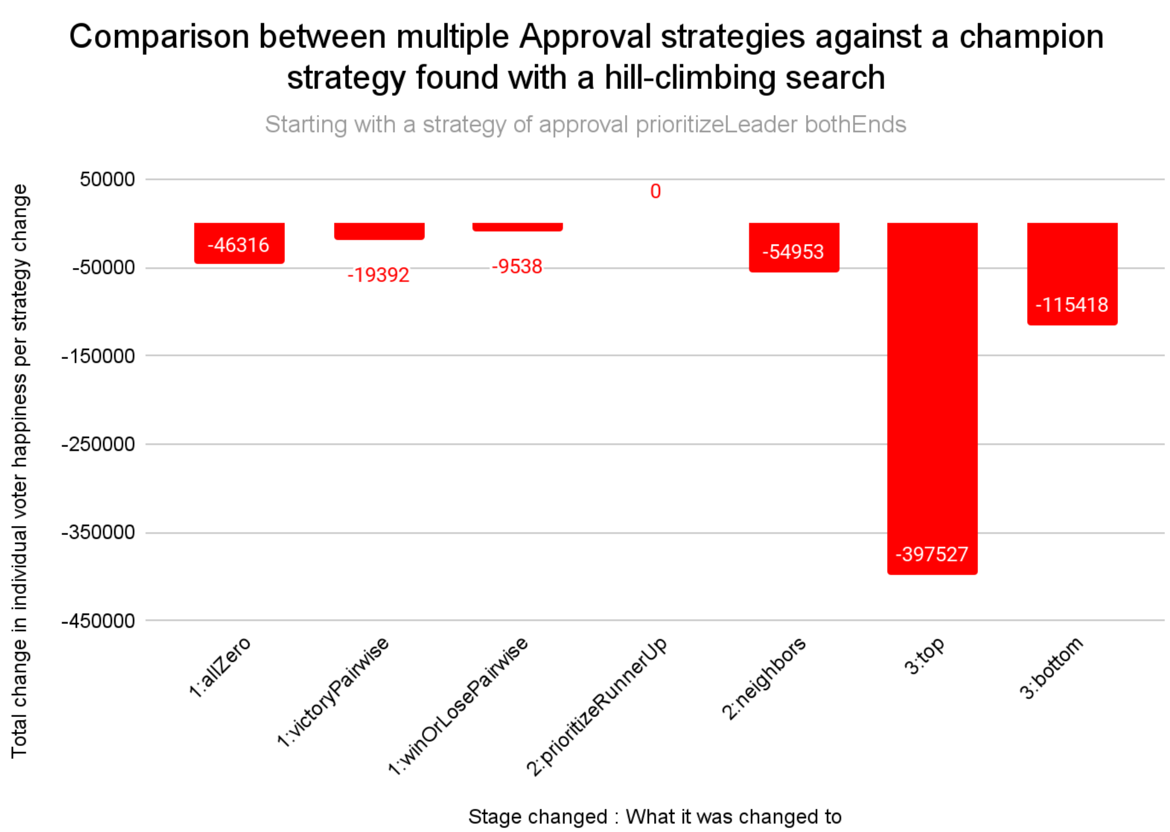

Figure 5-2.

We see that **prioritizeRunnerUp** in stage 2 replacing **prioritizeLeader** results in identical ballots when using **approval** in stage 1 and **bothEnds** in stage 3. All other alternatives result in unhappier strategizing voters on average. We therefore crown **approval**/**prioritizeLeader**/**bothEnds** (equivalent to **strategy A**) the strategy champion for *Approval* elections.

Figure 5-3 shows a *Hare* election starting from **allZero/prioritizeLeader/top**. We did try starting from **allZero/prioritizeLeader/bothEnds**, but no result in that search came up with anything better than an equivalent search, while using **top** as stage 3 was designed specifically to be good for a *Hare* election.

This graph demonstrates that out of all the adjacent strategic options, **prioritizeRunnerUp** and **neighbors** in stage 2 and also **bothEnds** and **bottom** result in identical (sincere) outcomes. Only **victoryPairwise** and **winOrLosePairwise** show an increase in happiness with an election outcome on average when any given voter changes their strategy in stage 1 to one of those two options.

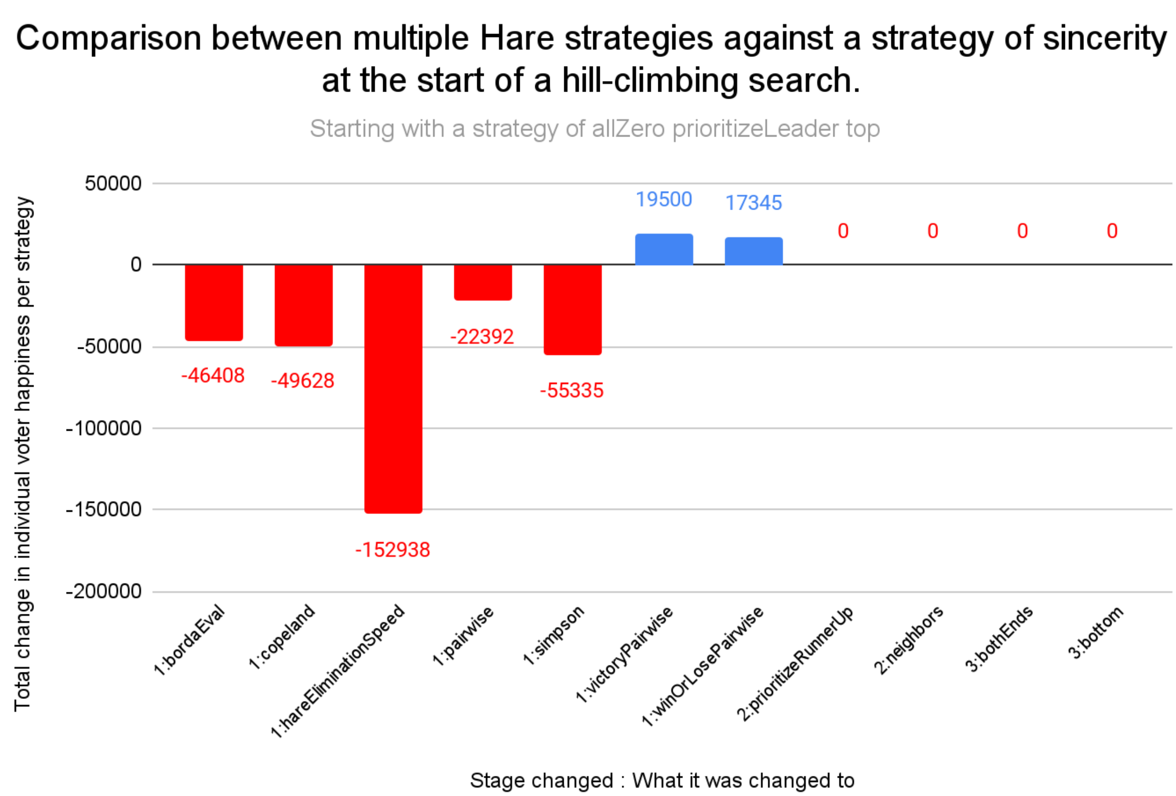

Figure 5-3

Figure 5-4 shows four separate *Hare* elections run, starting from **allZero**/**prioritizeLeader**/**top**, with the resultant changes from the adjacent searches. 'Experiment 1' here is the same data as contained in Figure 5-3, with the other three listed experiments being additional sets of 30 randomized voters in 1000 electorates. We can demonstrate consistent results: **prioritizeRunnerUp** and **neighbors**, **bothEnds** and **bottom** always result in no change in happiness of voters. **victoryPairwise** and **winOrLosePairwise** result in increases in happiness, with no clear 'superior' option between the two (at least in this data), and all other adjacent strategies result in less happy strategizing voters.

Figure 5-5 shows a *Hare* election where voters start from a strategy of **winOrLosePairwise**/**prioritizeLeader**/**top**, and try all adjacent strategies. In this search, we show that, with the exception of **victoryPairwise** in stage 1 and **prioritizeRunnerUp** in stage 2, this combination results in happier strategizing voters on average. The two options of **victoryPairwise** and **prioritizeRunnerUp** only show a small increase in overall happiness.

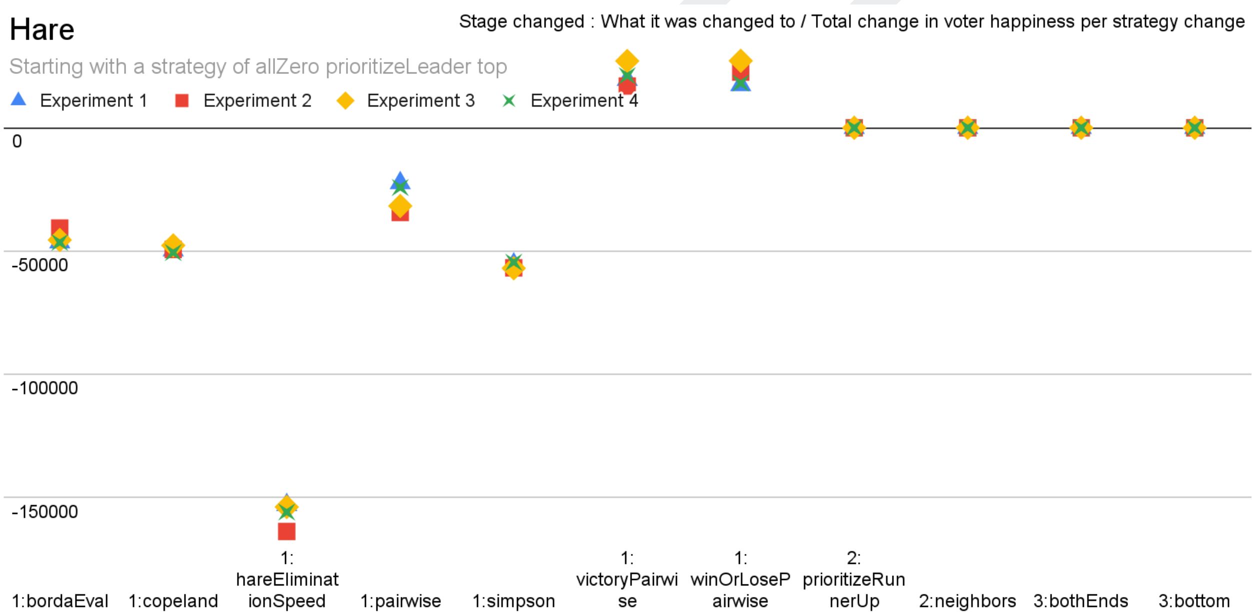

Figure 5-4

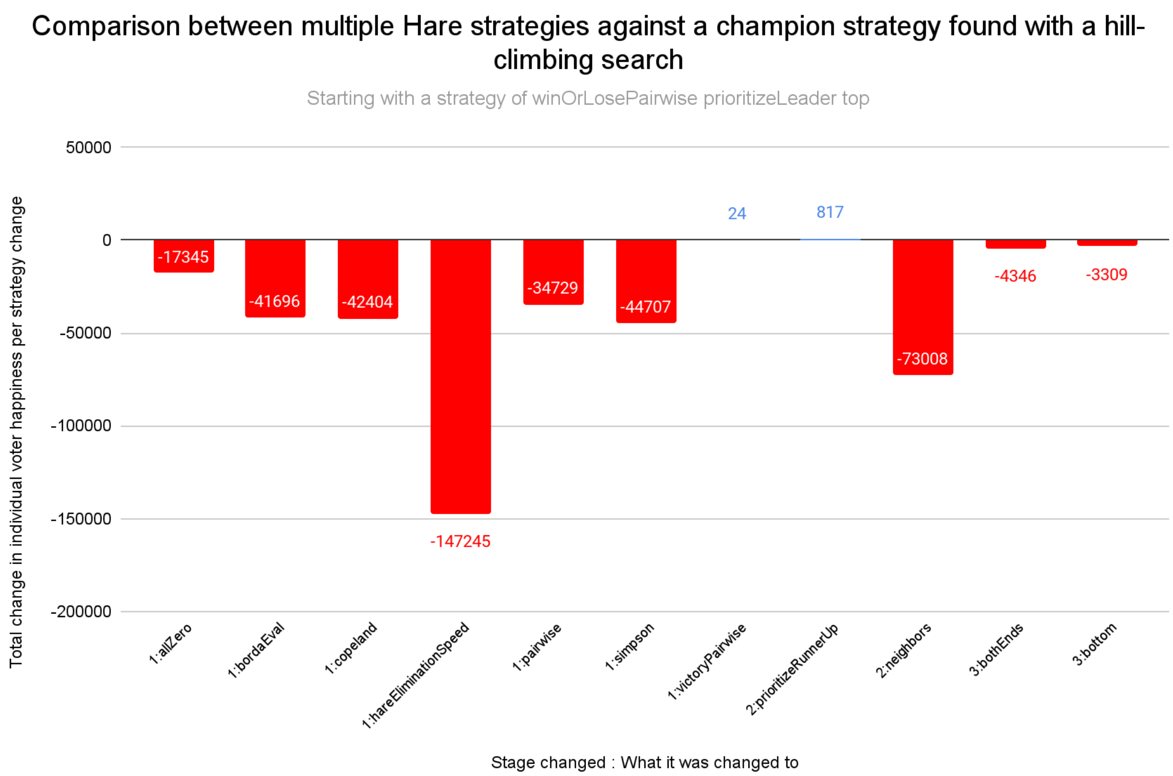

Figure 5-5

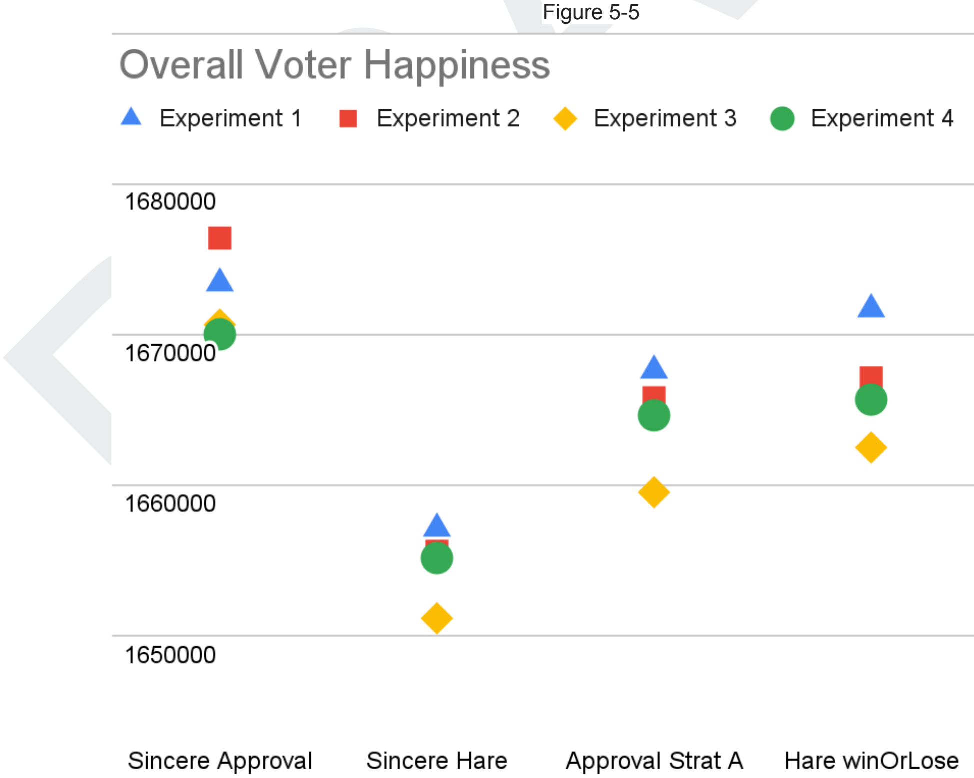

Figure 5-6

Figure 5-6 shows a measure of the overall happiness or rating of the winning candidate in a given election where every voter is using the selected strategy.

Conclusions

For Approval elections, we saw strong indications that our emulation of **Strategy A** did make individual voters that switch to it happier, similarly to the behavior seen in the prior work it was based on (LeGrand, 2008).

For *Hare*, both **victoryPairwise** and **winOrLosePairwise** when used with **top** showed themselves to be an election manipulation strategy that resulted in similarly improved opinions of outcome from the individual voter who switched to that strategy. This is one combination that results in a ballot sometimes voting for a candidate you prefer less over a candidate you prefer more.

We do show that a strategizing crowd of voters in an *Approval* election end up less pleased with the outcome overall compared to voters who only vote sincerely, which matches a result discussed by Smith (2000).

Interestingly, however, we seem to have found a strategy for a *Hare* method of election that seems to result in happier overall voters when compared to voters voting sincerely in *Hare*. This is an interesting result, as it runs counter to results in *Range Voting* by Smith. To speculate why, this may be a result of the strategy used in that prior work strategizing something approximately resembling a **hareEliminationSpeed** and **bothEnds** strategy, while our approach avoids strategizing at the **bottom**, and permits a large range of free expression of opinions within the voter's choices of where to place candidates on the ballot, due to the large number of ties in evaluations **winOrLosePairwise** applies to current poll results.

When all voters are maximally strategic, *Approval* and *Hare* result in roughly equally satisfied voters on average, but, when some voters are sincere or less than maximally strategic, *Approval* outcomes tend to make the average voter happier than *Hare* outcomes. In addition, it turns out that the most effective *Hare* strategies we found sometimes recommend that a voter vote for a less-preferred candidate above a favorite, whereas the most effective *Approval* strategy never will.

Future Work

We limited ourselves to electorates of 30 voters apiece, and six candidates in each election, no more, no less. In addition, we limited ourselves to 10 rounds per election. Exploring more or fewer in each category could be worthwhile.

Further experiments into other election methods, such as *Borda*, *Copeland*, and others using these same sets of strategy options would be fairly straightforward to accomplish.

We attempted to explore how individual voters changing their strategy might impact overall outcomes, in an attempt to see what happened when one person changed their strategy in the face of everyone else using a different strategy. We saw some initial information that suggested in certain cases one person changing their strategy from X to Y when everyone else was using X could be a great decision for them, and even indications where the conditions being reversed resulted in similar outcomes, but we ran out of time to explore this idea further. It would be interesting to see if there was a balance point, where perhaps a certain percentage using X and a certain percentage using Y was stable enough that a single voter changing their own strategy would typically result in negative outcomes.

Explorations on how quickly election systems stabilize based on chosen strategies, or whether they go the full maximum number of rounds without ever stabilizing, might also yield interesting information. Loops of outcomes, where A might win one round, followed by C, followed by D, then looping back to A to repeat showed to be possible using these strategies, and how often those loops occur could be valuable information when evaluating voting systems.

Analysis could be performed on whether or not more reasonable 'real-world-like' distributions of voter opinions result in different outcomes or perhaps more or less manipulable systems. Multi-dimensional opinions, where voters vote based on where candidates stand on multiple topics, might also be an interesting area of research (Boehmer et al., 2021).

The 'risk' of a chosen insincere strategy could be evaluated. For a few examples: Is a given strategy shown to provide small benefits frequently, offset by less frequent but equally small worse outcomes? Or perhaps frequent smaller poor outcomes are being masked by infrequent, massive gains? Or the inverse, where frequent small improvements to outcome are offset by the risk of infrequent, much more undesired outcomes.

There were some indications that either being early or late in voting might impact how effective a strategic vote might be. Testing whether or not that is the case might be possible.

Additional options might be developed for each of the stages.

Analyzing how *different* a strategic ballot is from a sincere ballot could be an interesting comparison. For example, how often a voter places a less preferred candidate over a more preferred candidate.

Bibliography

Bartholdi, J. J., Tovey, C. A., & Trick, M. A. (1989). The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6(3), 227–241. https://www.jstor.org/stable/41105918

Bartholdi, J. J., & Orlin, J. B. (1991). Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8(4), 341–354. https://www.jstor.org/stable/41105995

Boehmer, N., Bredereck, R., Faliszewski, P., Niedermeier, R., & Szufa, S. (2021). Putting a compass on the map of elections. *In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence* (IJCAI '21), pages 59–65. https://www.ijcai.org/proceedings/2021/0009.pdf

Cranor, L. F. (1996). Declared-strategy voting: An instrument for group decision-making. Washington University in St. Louis. https://lorrie.cranor.org/pubs/diss/

Donald G. Saari. Basic Geometry of Voting. Springer-Verlag, Berlin, Germany, 1995.

Duverger, M. (1959). Political parties: Their organization and activity in the modern state. Metheun & Co. Ltd. http://digilib.fisipol.ugm.ac.id/handle/15717717/12090.

Favardin, P., Lepelley, D., & Serais, J. (2002). Borda rule, Copeland method and strategic manipulation. *Review of Economic Design*, 7(2), 213. https://doi-org.easydb.angelo.edu/10.1007/s100580200073

Green-Armytage, J. (2014). Strategic voting and nomination. *Social Choice and Welfare*, 42(1), 111–138. https://doi-org.easydb.angelo.edu/10.1007/s00355-013-0725-3

LeGrand III, R. H. (2008). Computational aspects of approval voting and declared-strategy voting. https://www.cs.angelo.edu/~rlegrand/research/dissertation.pdf

Steven J. Brams and Peter C. Fishburn. *Approval Voting*. Birkhäuser, Boston, Massachusetts, 1983.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4th edition. Pearson, 2021. ISBN: 978-0-13-461099-3.

Smith, W. D. (2000). Range voting. https://www.rangevoting.org/WarrenSmithPages/homepage/rangevote.pdf.