|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Contract Name** | **Purpose/Application domain** | **Administration** | **Vote input** | **Winning Strategy** | **Winning variable** |
| Ballot.sol | Used to vote for multiple proposals | chairperson | uint[] **weights**,  bytes32[] **voterNames**,  uint[] **proposalIds** | Proposal with the most votes win | winningProposal |
| DemaxBallotFactory.sol | Used to vote for 2 proposals | Governor | uint **proposal** | Proposal with the most votes win | YES/NO |
| Gatekeeper.sol | Used in Panvala network  Used to vote for multiple slates | Gatekeeper | address **voter**,  bytes32 **commitHash**,  uint **numTokens** | Slate with the most votes win | contest.winner |
| ibaVoter.sol | Used to vote for multiple proposals | Chainperson | address **chainperson**, uint256 **ballot**,  uint256 **proposalNum** | Proposal with the most votes win | winner |
| localElection | Originally planned for 2019 Hong Kong local elections.  Used to vote for council. | Owner | uint256 **voterID**,  uint256 **hashedEmail**,  string memory **council**, string memory **singleVote** | Council with the most valid votes win | councilVoterNumber[council] |
| PLCRVotingCheckpoint.sol | Used in Polymath Network where the issuer or board members can propose an agenda to get the opinion of the investors over it.  Used to vote for multiple proposals | Admin | uint256 **\_ballotId**,  bytes32 **\_secretVote** | Proposal with the most votes win | winningProposal |
| Redenom.sol | Used in Redenom DAO to vote for start-up projects to be funded by Redenom.  Used to vote for multiple projects | onlyAdmin | uint **\_id** | Project with the most votes and is active win | \_winningProject |
| TACVoting.sol | Used in Taekwondo Cooperative to vote in taekwondo matches.  Used to vote for multiple matches | - | uint64 **electionId**,  uint64 **matchId** | Match with the most votes win | Elections[electionId].winningMatch |
| TomiQuery2.sol | Used to vote for 2 proposals | Governor | address **user**,  uint256 **proposal** | Proposal with the most votes win | YES/NO |
| WeightedVoteCheckpoint.sol | Used in Polymath Network.  Used to vote for multiple proposals | Admin | uint256 \_**ballotId**, uint256 \_**proposalId** | Proposal with the most votes win | winningProposal |

# Voting Criteria

**Majority Criterion**

If a candidate wins by a majority, then that candidate should win the election by another method.

Evaluation: If the majority winner does not win by another method, then this criterion is violated. If majority winner does win by another method, then it is not violated.

**Condorcet** **Criterion**

If a candidate wins by head-to-head (pairwise) over every other candidate, then that candidate should win the election.

Evaluation: If one candidate wins over all others when paired up and does not win by another method, then the criterion is violated. If by another method, the head-to-head winner still wins, then the criterion is not violated.

**Irrelevant Alternatives Criterion**

A candidate wins. In a recount the only changes are that one or more of the candidates are removed. Original winner should win.

Evaluation: A candidate drops out. Look at the 1st place votes. If another candidate wins, then it is violated. If original winner still wins, then the criterion is not violated.

**Monotonicity Criterion**

A candidate wins an election. Voter preference changes before the re-election only favour the winner; the winner should still win the election.

Evaluation: Voter preference changes favouring the winner and re-election winner different, it is violated. Voter preference changes favouring the winner and re-election winner the same, it is not violated.

# Voting Methods

**Plurality Method**

The choice with the most first-preference votes is declared the winner. Ties are possible, and would have to be settled through some sort of run-off vote.

Problem with Plurality Method:

If there are 3 or more choices, it is possible that a choice could lose but when compared to one-to-one comparison could be preferred over the plurality winner. This violates the Condorcet Criterion.

**Borda Method**

Points are assigned to candidates based on their ranking, for example: 1 point for last, 2 points for second-to-last. The point values for all ballots are totalled and the candidate with the largest point total is the winner.

Problem with Borda Method:

A candidate could receive majority of the first-choice votes and still lose the election. This violates the Majority Criterion.

**Pairwise Comparison**

For each pair of candidates (there are C(N,2) of them), we calculate how many voters prefer each. The candidate of the pair whom most voters prefer is awarded one point, and the loser get 0 points. If there is a tie, each candidate gets half a point. At the end, the candidate with the most points wins.

This satisfies the Condorcet Criterion, Majority Criterion, and Monotonicity Criterion.

Problem with Pairwise Comparison:

Violates the Independence of Irrelevant Alternatives Criterion.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Majority Criterion | Monotonicity Criterion | Condorcet Criterion | Irrelevant Alternatives Criterion |
| Plurality Method | Satisfies | Satisfies | Violates | Violates |
| Borda Method | Violates | Satisfies | Violates | Violates |
| Pairwise Comparison | Satisfies | Satisfies | Satisfies | Violates |