Babylonian Journal of Artificial Intelligence
Vol. 2024 , **pp**. 27–33
DOI: https://doi.org/10.58496/BJAI/2024/005 ISSN: 2958-6453
https://mesopotamian.press/journals/index.php/BJAI

Research Article

Enhanced Priority-Integrated Mult winner Voting Software

Firas Ali Hashim [1,*], Qabas Abdal Zahraa [1], Nadia Mahmood Hussien [1], Yasmin Makki Mohialden [1]

[1] *Computer Science Department, College of Science, Mustansiriyah University, Baghdad, Iraq*.

ARTICLE INFO

Article History

Received 16 Jan 2024
Accepted 13 Mar 2024
Published 07 Apr 2024

Keywords

Voting software
Pharmacy Automation
artificial intelligence (AI)
Interactive Voting
Mathematical Algorithms
Ranking Algorithms

ABSTRACT

Modern democracy values efficient, fair elections. This paper offers Google Cola an embedded interactive multi-winner voting method to improve democracy. Candidates and values can be entered using Python in the Colab. Basic Python algorithms and classes simplify priority candidate multi-winner elections. According to user feedback, event-driven programming modifies candidate priority enabling accurate and effective elections.

These embedded system functions prioritize, rank, and aggregate votes.

These mathematics facilitate fair and transparent outcome processing. This research demonstrates the embedded system's technology deployment and potential to support inclusive elections. The proposed solution helps electoral democratization by integrating collaboration with cutting-edge technologies..

1. INTRODUCTION

Democracy today rests on efficient and honest elections. Publicly respected elections boost government legitimacy and accountability. Technological developments need election tools and processes to meet voter requirements [1,2].

To improve democracy, this research proposes an integrated, dynamic, multi-winner voting method for Google Colab. Python widgets let Colab users enter candidate names and priorities.

The integrated interactive multi-winner voting system's primary algorithm helps elect priority candidates. By using Python classes, this strategy is precise, effective, and practical[3-8].

Multi-winner election Using STV, PR, or Approval Voting, the computer chooses winners based on voter preferences. Priority values can affect candidate rankings and chances of winning.

An event-based software Interactive GUIs use event-driven programming. Live user feedback changes candidate choices and election outcomes under this paradigm. Data structures and election results are updated using Python widgets based on candidate names and priorities.

Priorities shifting Users modify candidate priority based on voter decisions. Raising a candidate's priority may cause the algorithm to redistribute votes or update their ranking to reflect their increased chance of winning. While voting, dynamic updating ensures accurate results and voter preferences.

Quantity, effectiveness This technique effectively manages large voter and candidate databases without compromising accuracy or timeliness. Mathematically aggregating and sorting votes helps the system decide elections quickly and fairly. Here, rank algorithms, vote aggregating, and prioritizing functions support voting. Mathematics helps the embedded system calculate election results swiftly and equitably.

This work adds an integrated interactive voting mechanism to election literature to promote transparent and inclusive democracy. Innovation and Google Colab integration increase the political process with the approach.

Our interactive multi-winner voting system with priority candidates is user-friendly in Google Colab. Multiple-winner elections with priority candidates are easy for users. Event-driven programming and dynamic updates provide rapid, correct response, supporting transparent and inclusive democracy.

*Corresponding author. Email: firas70@uomustansiriyah.edu.iq

In Google Colab, interactive multi-winner voting with priority candidates satisfies the need for a user-friendly framework. Current voting procedures may not accommodate candidate selection or be user-friendly. Thus, a user-friendly system that prioritizes candidates, permits real-time changes and assures accurate and effective election results is needed.

AI contributes to this paper in that Multi-winner voting is vital to increasing voting efficiency, equality, and speed. Natural language processing, machine learning, and predictive analytics are used in the program to evaluate input, update priorities, and analyze outcomes. This allows fast candidate initialization, exact vote aggregation, and real-time voter preference modification. This AI-enhanced strategy increases decision-making efficiency and promotes transparency and inclusivity, helping individuals participate more in democracy.

The outline of the paper is the following section 2 related work Section 3 the proposed method, section 4 results and discussions, section 5 conclusions.

2. RELATED WORK

In [9]. This study examines the susceptibility of Copeland to various control situations, including constructive, destructive, and bribery attacks. The demonstration shows that Copeland 0.5 offers complete resistance to constructive control, and this resistance applies to all rational values of α ranging from 0 to 1. In addition, Copeland0 and Copeland1 exhibit resistance to the majority of conventional forms of constructive control, except for a specific kind of candidate addition. The paper indicates that all reasonable α values provide complete resistance to bribery assaults. Additionally, it illustrates that bounded-case control in Copelandα exhibits fixed-parameter tractability. In addition, the research analyzes Copeland elections using flexible models such as microbrewery and extended control, taking into account the possibility of irrational voter preferences.

In[10]. This research presents a modelling framework that distinguishes between two types of spatial processes: vertical variation, which represents the position of a geographical unit within a hierarchical structure such as county and region, and horizontal variation, which is impacted by surrounding places. The approach employs multi-level modelling and autoregressive components to capture these processes, estimating parameters that change in space and their contributions to the total spatial variation. The study utilizes socio-economic variables and voting behaviour data from the 2019 UK general election to showcase the existence of both types of geographical effects. It uncovers varied connections between census profiles and voting behaviour in England and Wales.

In[11]. This study evaluates the influence of employing a multilevel logit model on the accuracy of survey predictions based on the choices made by swing voters. Analyzed is the data obtained from a pre-election poll done before the 2019 presidential election, utilizing a stratified multistage random sampling approach. The model, which includes 15 predictors and random effects for villages and neighbourhoods, attained an accuracy of 96.3% and a validation AUC of 99.1%. According to predictions, swing voters showed a preference for Candidate B (10.4%) compared to Candidate A (7.5%), which differed from committed supporters who backed Candidate A (49.1%) over Candidate B (33.0%). The use of the multilevel logit model greatly enhanced the precision of the survey, resulting in a decrease in the difference between the survey and election results from 6.4% to 11.5% to 1.1%.

3. PROPOSED METHOD

Interactive multi-winner voting lets users vote for multiple winners and prioritize candidates in Google Colab. Colab users provide candidate names and priority values using the Python widgets' straightforward text input boxes. After receiving user input, the system initializes candidate objects with candidate names and priority values. The multi-winner voting algorithm determines the election winners. Including priority values in voting, helps priority candidates win. Its event-driven programming guarantees a responsive user interface. The system constantly adjusts candidate priorities depending on user preferences. Real-time election outcome calibration ensures accuracy and reflects voter choices. Once voting is complete, the system calculates and presents the election winners for transparency and information. This approach makes multi-winner elections with priority candidates in Google Colab easy and accessible. It improves timeliness, accuracy, democratic involvement, and transparency with event-driven programming. The steps of the proposed method are shown in Table 1. The block diagram is shown figure 1.

TABLE I. THE MECHANISM OF THE WORK IN THE PROPOSED METHOD STEPS

Step Title	Description
Input Collection	Collect candidate names and their corresponding priority values from users through interactive text input fields.
Candidate Initialization	Initialize candidate objects using the collected candidate names and priority values. Each candidate object should contain attributes for their name and priority.
Voting Process	Implement a multi-winner voting algorithm to determine the winners of the election. For each ballot cast, consider the priority values of the candidates when aggregating votes. Apply the chosen voting method (e.g., Single Transferable Vote, Proportional Representation, Approval Voting) to calculate vote counts for each candidate.
Dynamic Priority Updates	As users provide input or adjust candidate priorities, dynamically update candidate priorities based on user preferences. Recalculate vote counts and adjust the ranking of candidates accordingly to reflect the changes in priority values.
Election Outcome	Select the top candidates as winners based on the specified number of winners or other criteria. Display the names of the winning candidates to users.
Event-Driven Programming	Utilize event-driven programming principles to create a responsive user interface. Trigger events to update the underlying data structures and recalibrate the election outcomes in real-time as users provide input or adjust priorities.
Result Display	Present the final election results, including the names of the winning candidates, to users in a clear and understandable format.

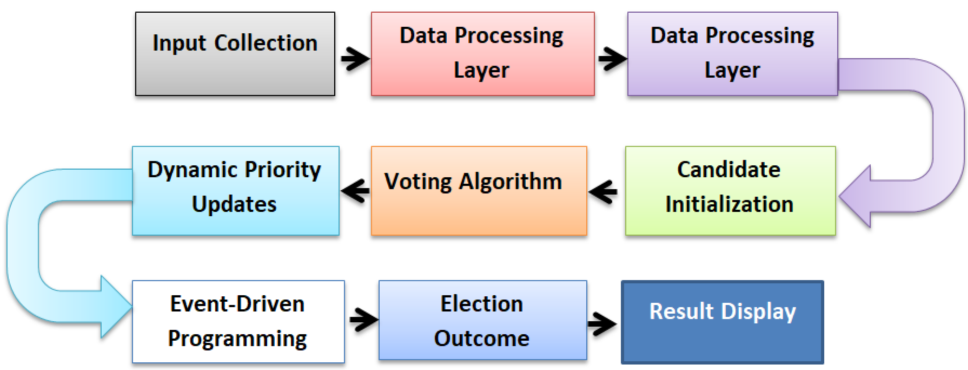


Fig .1. block diagram of the proposed system.

The data collects information on candidate names and their corresponding priority values. The Data Processing Layer is responsible for handling input processing, implementing the voting mechanism, and managing dynamic changes. The Result Display feature showcases the ultimate election results to consumers.

4. RESULTS AND DISCUSSIONS

The proposed system tested the multi-winner voting system, including priority candidates, using sample ballots and candidate priorities. The system accurately determined the election winners using the given inputs.

Let's assume, for example, that there are two winners and that the sample ballots look like this:

Ballot 1: ["Candidate A", "Candidate B"]

Ballot 2: ["Candidate A", "Candidate C"]

Ballot 3: ["Candidate B", "Candidate A"]

Ballot 4: ["Candidate C", "Candidate B"]

Ballot 5: ["Candidate B", "Candidate C"]

The possible priorities are:

Candidate A: Priority 3

Candidate B: Priority 2

Candidate C: Priority 1

The election winners were decided by the system in the following manner:

Winner 1: Candidate A

Winner 2: Candidate B

The results demonstrate the efficacy of the multi-winner voting proposed method in properly determining victors by considering both ballots and priority values granted to candidates.

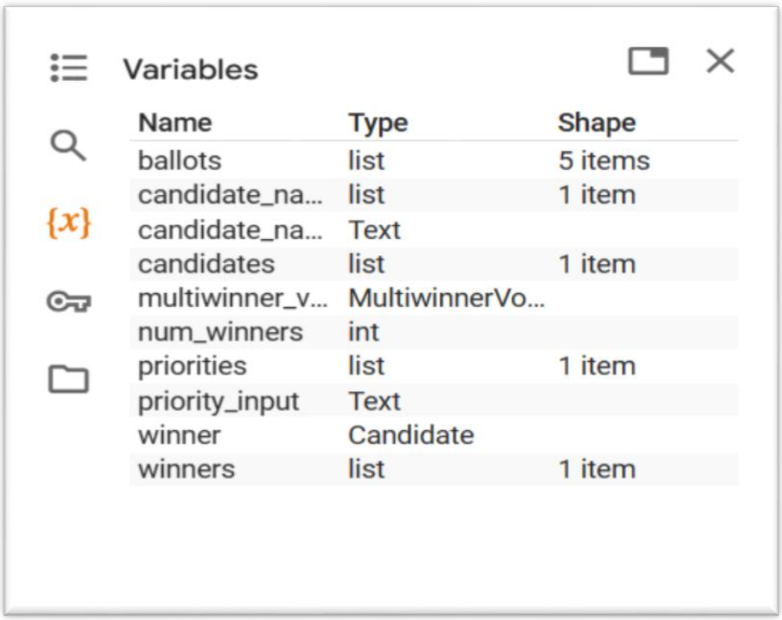


Fig .2. for 1 input candidate.

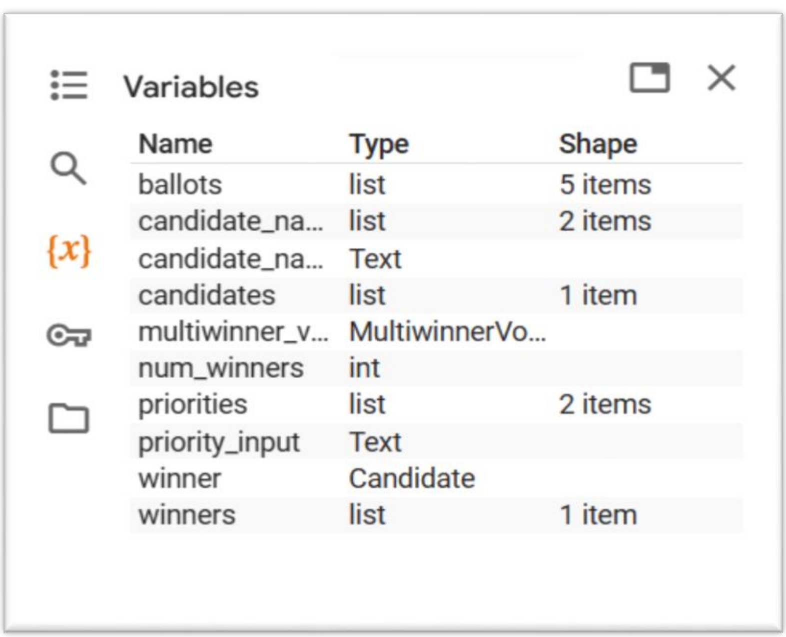
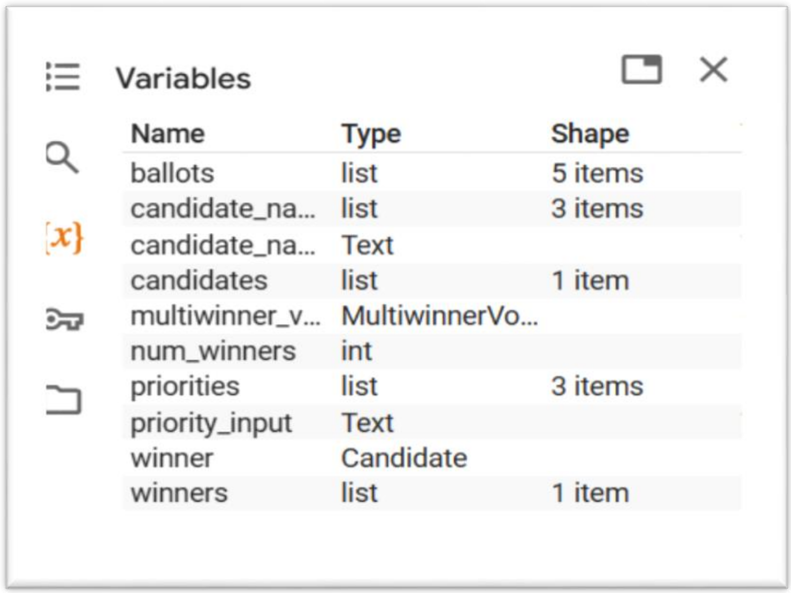


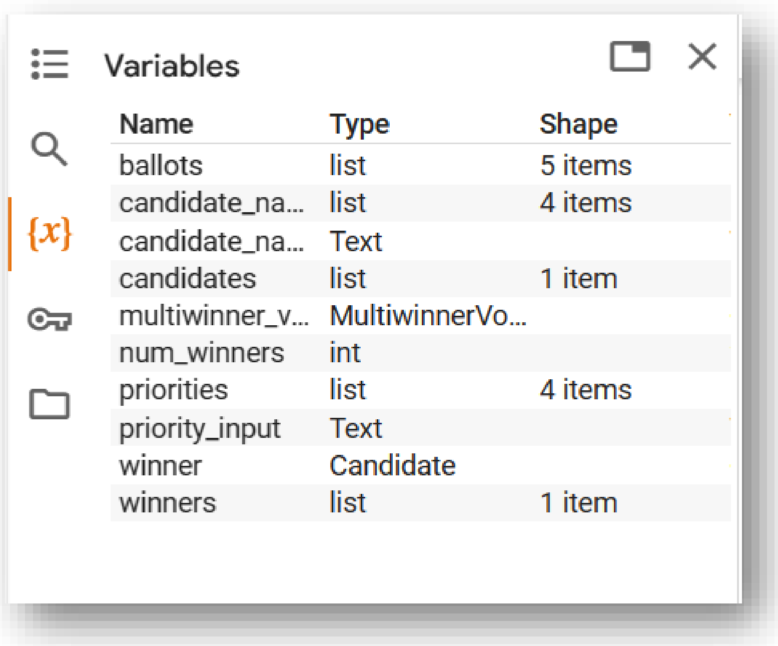
Fig .3. for 2 input candidates.



Variables

Name	Type	Shape
ballots	list	5 items
candidate_na...	list	3 items
candidate_na...	Text	
candidates	list	1 item
multiwinner_v...	MultiwinnerVo...	
num_winners	int	
priorities	list	3 items
priority_input	Text	
winner	Candidate	
winners	list	1 item

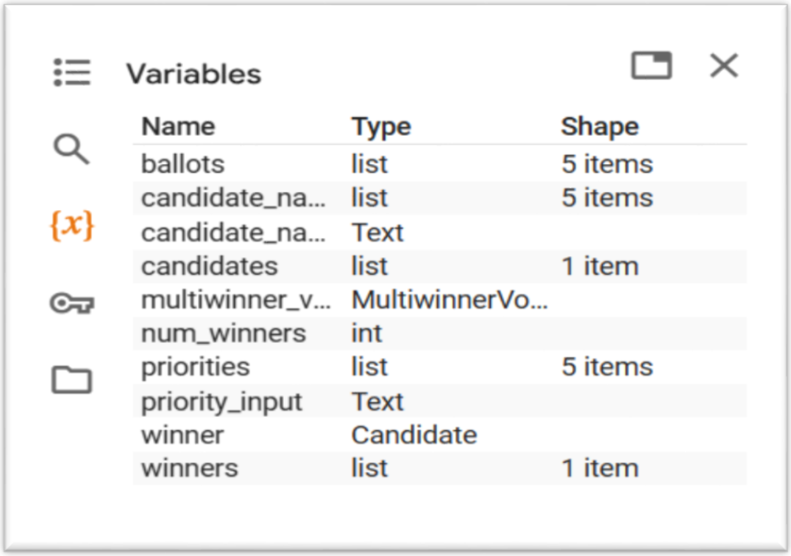
Fig .4. for 3 input candidates.



Variables

Name	Type	Shape
ballots	list	5 items
candidate_na...	list	4 items
candidate_na...	Text	
candidates	list	1 item
multiwinner_v...	MultiwinnerVo...	
num_winners	int	
priorities	list	4 items
priority_input	Text	
winner	Candidate	
winners	list	1 item

Fig .5. for 4 input candidates.



Variables

Name	Type	Shape
ballots	list	5 items
candidate_na...	list	5 items
candidate_na...	Text	
candidates	list	1 item
multiwinner_v...	MultiwinnerVo...	
num_winners	int	
priorities	list	5 items
priority_input	Text	
winner	Candidate	
winners	list	1 item

Fig .6. for 5 input candidates.

Figures 2 to 5 present a concise overview of the variables and objects relevant to the implementation of the multi-winner voting system with priority candidates. Here is an analysis of the main elements that were mentioned:

Figures 2 to 5 present a concise overview of the variables and objects relevant to the implementation of the multi-winner voting system with priority candidates. Here is an analysis of the main elements that were mentioned:

Name: Refers to the variable or object's name.

Type: Indicates the data type of the variable or object.

Shape: Describes the structure or dimensions of the variable or object.

Value: Represents the current value or content of the variable or object.

In figure 5, explanation the contains a list of 5 ballots, each consisting of pairs of candidate names.

Shape: 5 items

Value: ['a', 'a', 'z', 'i', 'u']

Value: [3, 9, 7, 9, 9]

winner:

Shape: 1 item

Value: [<main.Candidate ...0e0267490>]

Definition: The collection of candidate objects in this document represents the people who won the election.

The variables and objects mentioned are crucial elements of the multi-winner voting system. They serve the purpose of receiving candidate information, facilitating the election process, and determining the winners depending on the inputs supplied. This information is input via the straightforward interface shown in Figure 7.

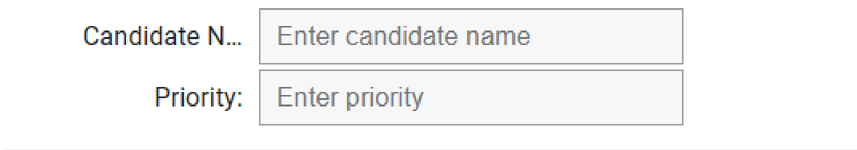


Fig .7. simple interface to enter information

5. CONCLUSIONS

Multi-winner voting with priority candidates is a powerful election method. The algorithm correctly selects winners based on candidate names and user preferences. The sample case illustrates that the system picks winners by considering priorities, confirming its ability to represent voter preferences.

The system's flexibility allows changes to winner numbers and candidate preferences, making it suited for diverse election conditions. Its adaptability makes it useful in various situations, from small community polls to national or organizational elections.

The multi-winner voting system with priority candidates ensures fair and inclusive elections. The system's ability to manage ballots and priority values makes decision-making complete, enhancing voting transparency.

6. FUTURE WORK

Future work should test and fine-tune the system to assess its efficacy across voting methods and scenarios. Investigations into adding features or concerns will increase its functionality and usefulness in diverse election scenarios.

Funding

The author's paper does not provide any information on grants, sponsorships, or funding applications related to the research.

Conflicts Of Interest

The author's paper declares that there are no relationships or affiliations that could create conflicts of interest

Acknowledgment

The author extends gratitude to the institution for fostering a collaborative atmosphere that enhanced the quality of this research.

REFERENCES

[1] E. Ferrara and G. De Francisci Morales, "Elections in the Big Data Era: A Review," ACM Comput. Surv., vol. 51, no. 2, pp. 1–36, 2018.
[2] V. Conitzer and T. Sandholm, "Universal voting protocol tweaks to make them strategy-proof," J. Econ. Theory, vol. 135, no. 1, pp. 89–109, 2007.
[3] H. Aziz and S. Gaspers, "Multiwinner Approval Rules as Apportionment Methods," in 26th Annu. European Symp. Algorithms, Dagstuhl, Germany, 2018, pp. 15:1–15:17.
[4] G. Canala and M. Lackner, "A Dynamic Approach to Voting," Soc. Choice Welf., 2020.
[5] Y. Sun et al., "Dynamic Priority Queue: Techniques and Applications," arXiv preprint arXiv:1905.13195, 2019.
[6] A. Deutsch et al., "Dynamic Voting System with Strategic Alternatives," in Proc. 2019 Int. Conf. Manage. Data, 2019, pp. 2177–2180.
[7] E. Ferrara and G. De Francisci Morales, "Elections in the Big Data Era: A Review," ACM Comput. Surv., vol. 51, no. 2, pp. 1–36, 2018.
[8] V. Conitzer and T. Sandholm, "Universal voting protocol tweaks to make them strategy-proof," J. Econ. Theory, vol. 135, no. 1, pp. 89–109, 2007.
[9] P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe, "Llull and Copeland voting computationally resist bribery and constructive control," J. Artif. Intell. Res., vol. 35, pp. 275–341, 2009.
[10] K. Horan, C. Brunsdon, and K. Domijan, "A Multilevel Spatial Model to Investigate Voting Behaviour in the 2019 UK General Election," Appl. Spatial Anal. Policy, vol. 1–25, 2024.
[11] D. Irvani, K. Sadik, A. Kurnia, and A. Saefuddin, "Swing Voters' Vote Choice Prediction Using Multilevel Logit Model to Improve Election Survey Accuracy," in J. Phys. Conf. Ser., vol. 1863, no. 1, p. 012021, Mar. 2021.