Tactical Voting Lab
Group 9

Yimin Yang
i6246099

Sheng Kuang
i6237193

Jingyang Zeng
i6256898

Xing Su
i6207451

ABSTRACT

Strategic voting, as a pivotal mechanism, plays an important role in many aspects of our society. From the perspective of reinforcement learning, strategic voting is considered as an important group decision-making mechanism in multi-agent settings. In this paper, we first implemented a robust Tactical Voting Analyst (TVA) agent to meet the analysis requirement of various voting schemes and strategic voting situations. With the inputs of specific variables (i.e., true preference, designated voting scheme), this agent outputs the non-strategic voting outcome, overall voter happiness level, strategic voting options and overall risk of strategic voting. Furthermore, we extend our agent to analyse several special situations (i.e.,collusion, counter-strategic voting and imperfect information). The results show that the challenge to find a promising strategic voting to increase the individual happiness level increases along with the number of participants joining the collusion. When with counter-strategy, the voters in higher happiness level are harder to maintain their happiness level than those with lower happiness level. It can be concluded that this TVA has an effect on voting analysis, especially for strategic voting.

KEYWORDS

Multi-agent System, Strategic Voting, Voting Scheme, Voter Happiness

ACM Reference Format:
Yimin Yang, Sheng Kuang, Jingyang Zeng, and Xing Su. 2021. Tactical Voting Lab Group 9. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), London, UK, May 3–7, 2021*, IFAAMAS, 4 pages.

1 INTRODUCTION

Strategic voting is a very common strategy in daily life. It plays an important role in both political elections and daily chores.[1] For example, strategic voting has just been used in the world-renowned event of the US presidential election. Strategic voting means that at least one of the involved voters supports an alternative other than her/his sincere preference in order to achieve a voting outcome that is more desirable for this voter than the outcome that would result from non-strategic voting. The four common types of strategic voting are compromising, burying, push-over and bullet voting.[2] The main difference between them is the change of the overall score of at least one alternative.

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, London, UK.

2 RESEARCH QUESTIONS

The main research question of this design is to build a software agent called Tactical Voting Analyst (TVA). When entering a voting situation and a voting scheme, TVA can generate a voting result which contains the following content: The non-strategic voting outcome, the overall voter happiness, possibly empty set of strategic-voting options and overall risk of strategic voting for this voting situation.[3] Through this voting process, we can get different voting results and happiness rankings of different voters.

3 METHODS

Tactical Voting Analyst(TVA) is a program agent for strategic voting analysis. It is constructed by four modules, as voting scheme generator, happiness calculation module, strategic voting generator and TVA main module. TVA defines input as voting scheme V and voting situation Π as well as output outcome O, overall happiness H, strategic-voting options S and overall strategic-voting risk R. It can be represented by:

$$[O, H, S, R] = TVA(V, \Pi) \tag{1}$$

$$O = \left\{\omega_1, \omega_2, ...m\right\}, S = \left\{\left\{\omega_1^*, \omega_2^*, ...m^*\right\}, ...\right\} \tag{2}$$

$$V = \begin{Bmatrix} Plurality, \\ VotingforTwo, \\ Anti-plurality, \\ Borda \end{Bmatrix}, \Pi = \begin{pmatrix} \pi_{11} & \cdots & \pi_{j1} \\ \vdots & \ddots & \vdots \\ \pi_{i1} & \cdots & \pi_{ij} \end{pmatrix} \tag{3}$$

Table 1 shows whether these strategic voting method could be applied to certain voting schemes.

	Compromising	***Burying***	***Bullet***
Plurality	Yes	Yes	No
VoteforTwo	Yes	Yes	Yes
Anti − plurality	Yes	Yes	Yes
Borda	Yes	Yes	Yes

Table 1: The feasibility of strategies applied to voting schemes

3.1 Voting Situation

Voting scheme Π is achieved by generating a $m \times n$ matrix (m: number of alternatives; n: number of voters). This matrix collects the true preferences of all voters and is read from a .xlsx file, where the program file is located. In this .xlsx file, the first column represents the index of the alternatives(i=1,...,m); the first row represents the

index of the voters(j=1,...,n); and each column represents the voting preference of individual voter.

3.2 Voting Scheme Generator

In this assignment, we consider four voting schemes: plurality voting, voting for two, anti-plurality voting and Borda voting. For the purpose of generating voting schemes, we define four Python functions(i,e.,getPluralOutcome(), getVotingForTwoOutcome(), getAntiPluralOutcome(), getBordaOutcome()).All functions have a necessary input parameter Matrix Π. To apply bullet voting conditionally, two extra parameters(i.e., bullet_tactic, bullet_voter) are introduced to the last three voting scheme functions. The output of functions is the voting outcome listing all alternatives in a descent sequence.[4]

3.3 Happiness Calculation Module

Overall voter happiness level H is a sum of happiness levels of all individual voters. The procedure steps are as following: first, calculate the distance d_j for each voter; second, calculate the individual happiness level; third, sum them together. Equations are listed below:

$$d_j = \sum_{i=1}^{m} w_i \left(O - order\left(\pi_j\right)\right), \tag{4}$$

$$H_j = \frac{1}{1 + \left|d_j\right|} \tag{5}$$

$$H = \sum_{j=1}^{n} H_j \tag{6}$$

In our code, we separate happiness level calculation into two Python functions:getVoterHappiness(), getHappiness(). The function getVoterHappiness() gets outcome O , true preference π_j and voter index voter_index as input, individual happiness level H_j as output. The function getHappiness() gets outcome O, true preference Π as input, overall happiness level H as output.

3.4 Strategic Voting Generator

There are three strategic voting types to be considered: compromising, burying and bullet voting. In particular, compromising and burying can be adopt to manipulate the vote simultaneously.

3.4.1 ***Compromising.*** The voter who wants to compromise makes an insincere vote by ranking an alternative higher in order to increase the voter happiness level.The pseudocode of compromising is presented in Algorithm 1.

3.4.2 ***Burying.*** Burying is also an insincere vote in which a voter ranks an alternatives lower to beat it.[5] Meanwhile , this solution increases the voter individual happiness level. The pseudocode of burying is presented in Algorithm 2.

3.4.3 ***BulletVoting.*** Bullet voting is quite a different strategy from the other two. The rationale is that a voter votes for only one alternative, despite having the option to vote for more than one according to current voting scheme. The pseudocode of bullet voting is presented in Algorithm 3.

Algorithm 1: Compromising

Input: v_index: The strategic voter index; Π: True preference matrix;
Output: π^*: The strategic voting preference for the voter; O^*: Compromised outcome; H^*: Overall happiness level; h^*: The voter's happiness level;
initialize π^*, O^*, H^*, h^*;
for $alternative - i \in \{1, ..., n\}$ **do**
  for $alternative - j \in \{1, ..., n\}$ & $i \neq j$ **do**
    π^*_{ij} = swapOrderForward(i, j);
    O^*_{ij} = getOutcome(v_index, Π);
    h^*_{ij} = getVoterHappiness(v_index, π^*_{ij});
    if $h^*_{ij} > h^*$ **then**
      $\pi^* = \pi^*_{ij}$;$O^* = O^*_{ij}$;
      H^* = getHappiness(Π^*);
    end
  end
end

Algorithm 2: Burying

Input: v_index: The strategic voter index; Π: True preference matrix;
Output: π^*: The strategic voting preference for the voter; O^*: Compromised outcome; H^*: Overall happiness level; h^*: The voter's happiness level;
initialize π^*, O^*, H^*, h^*;
for $alternative - i \in \{1, ..., n\}$ **do**
  for $alternative - j \in \{1, ..., n\}$ & $i \neq j$ **do**
    π^*_{ij} = swapOrderBackward(i, j);
    O^*_{ij} = getOutcome(v_index, Π);
    h^*_{ij} = getVoterHappiness(v_index, π^*_{ij});
    if $h^*_{ij} > h^*$ **then**
      $\pi^* = \pi^*_{ij}$;$O^* = O^*_{ij}$;
      H^* = getHappiness(Π^*);
    end
  end
end

3.5 TVA main module

TVA main module integrates all modules together to achieve two goals:

(1) Generating the optimal strategic vote for each voter;
(2) Data pre-process and interaction mode.

When running the main program, system first checks the validity of data, then asks user to choose one voting scheme to continue.After enter the index from 1 to 4 of voting schemes, the system prints all the expected outputs.

4 EXPERIMENT&DISCUSSION

We begin basic experiment by demonstrating TVA's performance using five test cases in Section 4.1. A brief summary of test cases is presented in Table 2. After analyzing the basic experiments, we

Algorithm 3: Bullet Voting

Input: v_index: The strategic voter index; Π: True preference matrix;

Output: π^*: The strategic voting preference for the voter; O^*: Compromised outcome; H^*: Overall happiness level; h^*: The voter's happiness level;

initialize π^*, O^*, H^*, h^*;

for $alternative - i \in \{1, ..., n\}$ **do**

  π_i^* = popAlternatives(i);

  O_i^* = getOutcome(v_index, Π);

  h_i^* = getVoterHappiness(v_index, π_i^*);

  if $h_i^* > h^*$ **then**

    $\pi^* = \pi_i^*$;$O^* = O_i^*$;

    H^* = getHappiness(Π^*);

  end

end

realized that strategic voting in real life is much more complicated than our simplified experiments. Therefore, we set up several extension experiments in order to explore the more complicated voting situation in Section 4.2-4.5.

Test case	1	2	3	4	5
Number of preferences	4	5	5	10	10
Number of voters	5	4	10	5	10

Table 2: A brief summery of five test cases we use in the basic experiment

4.1 Basic experiment

This assignment will first test plurality voting as well as other three voting schemes and reveal whether there is no voting scheme that is always better than others.

As shown in table 2, we designed five examples with different numbers of preference and voters, each example is generated randomly. To analyse the association between voting scheme and voter's preference, we calculated the risk of each scheme(see table 3).As shown in the table, there is no voting scheme that is always better than the other. In most cases, borda produce higher risk, it make sense because Borda takes whole preference order into account.[6] As for other plurality,vote-for-two and anti-plurality voting schemes, we could hardly indicate any intuitive connection between voting schemes and voting preference matrices.

The voters and candidates matrix of different structures are the major factor that affects the outcome and risk. Besides, in this assignment we found that the initialization of the voter's preference matrix may also have an impact on the risk. We change the preference matrix of the second and fourth voter in data example 5, by randomly selecting candidates A and C, and B and D, swapping the preferences of A and C as well as the preferences of B and D. Thus ,a lot of new data samples were generated, for example, the risks change from 0.4, 0.5, 0.5 and 0.9 to 0.33, 0.67, 0.67 and 1 after changing the initial preference of voter 2. From adequate experimentation, we can see that different initializations certainly influence the possible tactical voters and risks of each scheme. However, As our initialization method is simple and lack larger preference matrix, it is insufficient to conclude a accurate relationship between the matrix initialization and each voting scheme.

Test case	Plurality	Vote-for-two	Anti-plurality	Borda
1	0	0	0.67	0
2	0	0	0.6	0.6
3	0.1	0.2	0.6	0.8
4	0.33	0.5	0.67	1.0
5	0.4	0.5	0.5	0.9

Table 3: Overall risk of strategic voting among five test cases using four voting schemes

Another aspect of our experiment is to investigate whether the type of tactical voting will affect global happiness. However, after sufficient experimentation of our data samples, we found that for any voting scheme it is easily constructing an example where the overall happiness increase or decreases after tactical voting, that is, no matter what the scheme is, there will be always easily to find some specific examples where global happiness increases or decreases. Therefore, we can conclude that is no voting scheme that is not susceptible to tactical voting and also that decreases global happiness after tactical voting.

4.2 Extension1: Collusion

Voter collusion is one of the complicated voting situations. Here, we assume a collusion principle that voters collaborate only if they can receive higher happiness. Basically, after analyzing the original true happiness for every voter, we search the combination among the voters, and this combination should increase the happiness of every member in this combination because if a voter can not get any benefit from collusion, he will not collaborate.

According to the search, we found that the amount of the small combination is higher than the amount of the larger combination. For example, see Figure 1. This result is what we expect because when more people join the collusion, it's more difficult to find a voting strategy that can increase the happiness of every voter in the collusion.

4.3 Extension2: Counter-strategic Voting

Voting strategies are accessible for every voter. Thus, it's reasonable that if a voter uses counter-strategy voting to maintain or reach a high happiness level when other voters use strategy voting. Furthermore, other voters also would use a new counter-strategy to counter the counter-strategy which is applied before. [7] Therefore, we set up experiments to analyze to counter-strategy situation. In the experiments, the information of all voters' preference list is updated in each episode, and the voter who has the lowest happiness would use strategy or counter-strategy to try to get a better happiness level in each episode.

According to the experiments(Figure 2), we found that the voter who has the highest happiness level is hard to maintain his high happiness by counter-strategy. In contrast, the middle happiness

A. True preference

	Voter 0	Voter 1	Voter 2	Voter 3	Voter 4	Voter 5
Preference	C>A>D>B	B>D>C>A	C>D>A>B	B>D>C>A	B>C>D>A	A>B>C>D
Happiness	0.125	0.5	0.143	0.5	1	0.143

B. Collusion between 0 and 5

	Voter 0	Voter 1	Voter 2	Voter 3	Voter 4	Voter 5
Preference	**A>C>B>D**	B>D>C>A	C>D>A>B	B>D>C>A	B>C>D>A	**A>B>C>D**
Happiness	**0.143↑**	0.25↓	0.125↓	0.25↓	0.5↓	**0.25↑**

C. Collusion between 2 and 5

	Voter 0	Voter 1	Voter 2	Voter 3	Voter 4	Voter 5
Preference	C>A>D>B	B>D>C>A	**A>C>B>D**	B>D>C>A	B>C>D>A	**A>C>D>B**
Happiness	0.25↑	0.167↓	**0.2↑**	0.167↓	0.333↓	**0.2↑**

D. Collusion among 0, 2, 5

	Voter 0	Voter 1	Voter 2	Voter 3	Voter 4	Voter 5
Preference	**C>A>D>B**	B>D>C>A	**A>C>B>D**	B>D>C>A	B>C>D>A	**A>C>D>B**
Happiness	**0.25↑**	0.167↓	**0.2↑**	0.167↓	0.333↓	**0.2↑**

Figure 1: Individual happiness levels fluctuation when collisions happen using test case 1

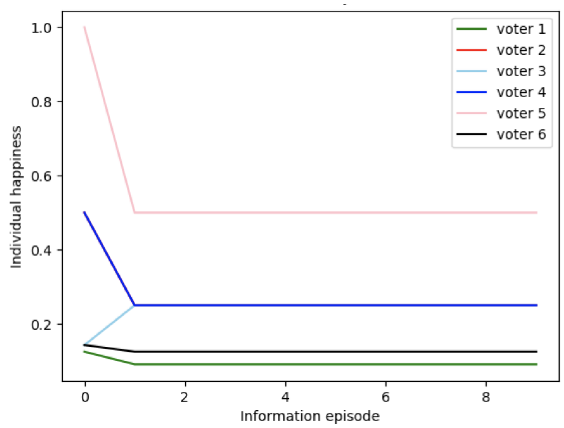

Figure 2: Information evolution of counter-strategic voting

voters have more space to maintain happiness even reach higher happiness. Moreover, the low happiness voters may still keep low happiness no matter what strategy they apply.

4.4 Extension3: Collusion & Counter-strategic Voting

We also consider the situation where voters use complex strategies, such as collusion with voting strategy. Then, we experiment that all voters can apply all voting strategies and collusion with perfect information. As a consequence, every voter should only choose the policy which provides the highest happiness level.

However, in our experiments, we found there is no collusion during experiments. Each voter chooses a voting strategy alone rather than using a voting strategy with collusion. In other words, if every voter only considers optimal happiness, no collusion will be conducted because there always has a voter that finds a better happiness level beyond the collusion.

4.5 Extension4:Imperfect Information

In the basic experiment, we assume every voter has perfect information for voting. However, this situation can not happen in real life, so we also test the situation where the information about other voters is imperfect. In our experiment, we add new information about a voter in each episode, then we apply the strategy basing the known information, at the end we calculate the happiness by the voting result. we expected the happiness would converge after some episode, but in fact, the happiness of a voter does not converge when more information is added. Therefore, in our experiments, we conclude that if the information is imperfect, the voting strategies might be useless in such a situation.

5 CONCLUSION

The proposed research is a study mainly on strategic voting and the development of a model called Tactical Voting Analyst(TVA) which can analyze the risk of strategic voting. When enter different types of voting scheme and voting situation, the model will generate different outcome including a non-strategic voting outcome, overall voter happiness level, possibly empty set of strategic-voting options and overall risk of strategic voting for different voting situations. In the assignment, we investigate the relationship between tactical voting types, happiness and voting scheme. Furthermore, a number of more complex scenarios such as voter collusion, counter-strategic voting and presence of imperfect information were discussed. Future work would be to extend the model to different situation and more complex domains, such as reality democratic elections

REFERENCES

[1] Bruce E Cain. Strategic voting in britain. *American Journal of Political Science*, pages 639–655, 1978.

[2] John J. Bartholdi III James B. Orlin. Single transferable vote resists strategic voting. 1991.

[3] R Michael Alvarez and Jonathan Nagler. A new approach for modelling strategic voting in multiparty elections. *British Journal of Political Science*, pages 57–75, 2000.

[4] John R Chamberlin. An investigation into the relative manipulability of four voting systems. *Behavioral Science*, 30(4):195–203, 1985.

[5] Yue Xu, Zengde Deng, Mengdi Wang, Wenjun Xu, Anthony Man-Cho So, and Shuguang Cui. Voting-based multi-agent reinforcement learning. *arXiv preprint arXiv:1907.01385*, 2019.

[6] Ioannis Partalas, Ioannis Feneris, and Ioannis Vlahavas. Multi-agent reinforcement learning using strategies and voting. In *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, volume 2, pages 318–324. IEEE, 2007.

[7] Yue Xu, Zengde Deng, Mengdi Wang, Wenjun Xu, Anthony Man-Cho So, and Shuguang Cui. Voting-based multi-agent reinforcement learning for intelligent iot. *IEEE Internet of Things Journal*, 2020.