Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Models for Iterative Multi-winner Approval Voting

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Javier Francés Martínez

Tutores: Prof. Dr. Markus Brill, Eva Onaindia de la Rivaherrera

2019-2020

Abstract

In this thesis we study the multi-winner voting rules known as Proportional Approval Voting and Phragmén's Voting Rule; as well as their sequential variants. Here voters submit their preferences through approval ballots. These voting rules are applied to an iterative framework, meaning that once the initial vote is submitted voters may deviate from their vote given the current state of the game. The framework for this voting system is defined. We describe heuristics to model the behavior that voters can take, when voting strategically, i.e. how voters can deviate from their initial true preferences to obtain a better outcome. Furthermore, we describe examples of games in which each heuristic introduced causes a change in utility. We also apply the considered heuristics to the sequential variants of the voting rules introduced in this thesis. Additionally, we study whether the different heuristics guarantee (or not) reaching a stable solution. For this reason, we introduce the concept of Stable States. We also introduce and explain the concept of Dynamic Price of Anarchy (DPoA), since it is used to evaluate the outputs of the voting systems. We provide different approaches for the generation of random voter preferences for its use in the experiments. We conduct experiments to assess how each heuristic behaves with each voting rule when changing different experimental values (such as number of voters, number of candidates, committee size or heuristic distribution) and we compare the initial and final winner committees. In such experiments, we play games in which all voters use the same heuristic. On the other hand, we follow this by setting up experiments in which different heuristics play against each other, to investigate how the heuristics presented in this thesis behave altogether. Finally, we end the thesis giving our conclusions on strategic voting in an iterative setting with approval multi-winner voting rules and identify possibilities for future work.

Contents

1 Introduction

1.1 Objectives and Motivation

The problem of social choice is the aggregation of individual preferences about alternative social actions as explained in [1]. In this thesis we will focus on the problem of aggregating the preferences into a multi-winner candidate set, also known as a winner committee. In an ideal world, we would like to aggregate all voters' preferences using a voting rule resistant to manipulation. However, it has been well established that many voting rules are in fact manipulable. In many cases, misreporting your preferences can lead to a more preferable outcome. This misreporting of preferences is also known as strategic voting and can be seen as a *selfish* action taken by a voter. Voters making strategic decision may have an advantage over voters that act truthfully. We will asses the consequences of possible manipulations, setting up experiments to understand how bad would it be to allow voters to manipulate in an iterative setting with an approval voting rule.

Manipulation in voting games has been deeply studied in the game theory field. We know from the Gibbard–Satterthwaite theorem ([8] and [19]) that whenever a voting rule has at least 3 possible outcomes and it is non-dictatorial (if there exists no voter that decides the outcome of the election without taking into account the other voters' preferences) then the voting rule is manipulable. Moreover, in [11] iterative games and more specifically their convergence and their resistance to manipulation has also been deeply studied. However, it focuses on *ranked* preferences, which refer to a set of (strict) orderings over a set of alternatives (or candidates).

We have chosen to study an iterative framework, because even though it is an informal voting process it is widely used in many online polls and surveys. Likewise, it can be found in more conventional settings such as political elections (as stated in [11]), where people may change their opinion after seeing real-time results announced by the media as the day of the election goes by.

1.2 Iterative Voting

Before introducing the iterative framework, we would like to give a brief overview of some game theoretical concepts that are crucial for the understanding of this thesis.

Any game usually consists of the following four elements: a set of players (in our case voters), a set of preferences over the different outcomes of the game, a set utility functions that map those different outcomes to utility values and a set of strategies that can be taken by the players (in our case the votes submitted by voters). The strategy profile is the set of current strategies taken by the players, which we will represent in our voting games as the current voting profile, where a voting profile is just the collection of the current votes submitted by the voters. A Nash Equilibrium is a state of the game in which every player knows the other players' current strategies and none of them have a gain in utility by changing their own strategy, as explained in [12, 13]. This is because every player is playing their best response strategy to the other players' strategies. A strategy s_i is said to be the best response for player i, if for all other possible strategies $s_i^{'}$:

$$g_i(s_i, s_{-i}) \geq g_i(s_i^{'}, s_{-i})$$

, where s_{-i} denotes the set of strategies taken by the other players and $g_i(s)$ is the function that maps the outcome of the game with strategy profile $s = (s_i, s_{-i})$, to a utility value for the player i, such explanation is taken from [7]. Applied to voting games we will say that a Nash Equilibrium is a state in which no voter gains from changing their current vote, given that they know the rest of the votes.

In contrast to other *traditional* voting frameworks, iterative voting allows voters to deviate from their initial true preferences, after casting their vote. This might be beneficial for them as once they are able to observe the current state of the game, they gain the possibility of getting a better outcome from changing their vote and misreporting their true preferences.

In the first phase of iterative voting, all voters vote at the same time. This means that no single voter has information about other voters' preferences, nor the current state of the game. Therefore, it is assumed that the best move for each voter is to vote in accordance with their true preferences. After this initial phase, a winner set is calculated using a specific voting rule (as described in Section 1.4). From now on all voters will know the current state of the game (i.e. the voting profile and the winner set). Even though voters know the set of true preferences they will be acting myopically. Thus, every decision taken by them will be done as if the game ends after it.

The game will go by rounds. In each round a voter will be drawn at random. The voter will then be asked whether they want to change their vote or not, given the current state of the game. After each round, the new winner set will be calculated (if needed) and will be seen by all the voters, along with the current voting profile. It is important that voters are drawn *randomly*, given that if an order is introduced we would grant them the possibility of applying backwards induction to determine the sequence of optimal deviations from their vote [11]. The game will end when no voter wants to change their vote, i.e. they cannot improve the outcome of the game for themselves. When this happens we will have reached a Nash Equilibrium. As explained by Meir in [11], this Nash Equilibrium can be traced from the initial state of the game through a sequence of deviations.

We will consider an Iterative Voting game as a directed graph. Each node in the graph will represent a voting profile, corresponding to a set of winner candidates. Each directed edge in the graph will correspond to a possible deviation of a voter (in its turn) going from one voting profile to another. This deviation may or may not be beneficial. All votes in the new voting profile will remain the same as in the previous one, except for the voter who has deviated from its previous vote. The initial node of the graph, will correspond to the first phase of the game, when all voters submit their true preferences. This initial node can have both outgoing and incoming edges. A Nash Equilibrium will be reached when the game enters a node with no outgoing edges (*terminal node*). This means that no matter which voter we draw, no one will want to change their vote. We will further make restrictions to the concept of Nash Equilibria, with the introduction of stable states (Chapter 3).

1.3 Framework

In this thesis we will focus on approval voting rules applied to an iterative setting. Given a finite set of voters N ($N = \{1, 2, ..., n\}$) and a finite set of candidates C ($|C| = m$), each voter $i \in N$ will give an opinion (approval score) for each candidate in the candidate list.

- 1: Voter i approves the candidate.
- 0: Voter i disapproves the candidate.

We will assume that voters have fixed and complete preferences. Therefore a vote A_i submitted by voter i will consist of a tuple of size m of 1s and 0s. The set of all votes will be denoted by A, while the set of all voters' true preferences will be denoted by TP. TP_i will denote voter i's true preferences, which will not contain any strict ordering between approved candidates, as we do not want to study ranked preferences. In each round the set of votes will be aggregated into a winner set W of size k, by means of a Multi-winner election rule. A Multi-winner election rule is a function that takes as input a voting profile, a set of alternatives (or candidates) and a committee size k, it then outputs a winning committee of size k. The utility (satisfaction) of voter i given the winner set W can be calculated with the following equation.

$$u_i = |TP_i \cap W| \tag{1.1}$$

The *social welfare* will denote the total utility of all voters. It will be calculated using the equation:

$$sw(W, TP) = \sum_{i \in N} u_i \tag{1.2}$$

The set of voters that approve a candidate $c \in C$ (*support of* c) will be denoted by N_c.

1.4 Multi-winner Election Rules

1.4.1 Proportional Approval Voting

The first voting rule that we will be looking at is *Proportional Approval Voting* (PAV)[2]. It is similar to *Approval Voting* rule, a solution for multi-winner elections introduced by

Steven J. Brams and Peter C. Fishburn in [4]. In contrast to approval voting, a voter i's satisfaction, given a winner set W is calculated by:

$$v_i = \sum_{h=1}^{|TP_i \cap W|} \frac{1}{h} \tag{1.3}$$

Where $|TP_i \cap W|$ is the number of approved candidates in the winner set and v_i can be seen as the $|TP_i \cap W|$ harmonic number. The aim of Approval Voting is finding the winner set that maximizes the total utility of all voters, while the aim of Proportional Approval Voting is finding a winner set that maximizes the satisfaction (v_i) of all voters. This total satisfaction is also known as the *PAV-score* of a winner set W, and can be calculated by:

$$s(W) = \sum_{i \in N} v_i \tag{1.4}$$

We can clearly see that PAV follows a proportional satisfaction model; i.e, the more satisfied a voter is, the lower the increase in PAV-score the winner set gains from another of its approved candidates entering it. Voters with few (or none) approved candidates in the winner set gain much more satisfaction, when an approved candidate enters the winner set.

In order to find the candidate set that maximizes the PAV-score, we must iterate through all sets of candidates of size k. Consequently, computing the outcome of PAV is NP-Hard (P. Faliszewski P. K. Skowron and J. Lang. 2015 [6]). For this reason, the problem becomes too computationally expensive when k or C are large. Therefore, we must search for a faster algorithm. Thus, we will now look at a variant of PAV: *Sequential Proportional Approval Voting* (SeqPAV).

1.4.2 Sequential Proportional Approval Voting

The sequential version is an approximation to PAV, but can be computed in polynomial time. SeqPAV turns PAV into a multi round election rule, where k rounds are performed. We first begin with an empty winner set ($W = \emptyset$). In each round we will select the candidate that gives the highest increase to the PAV score when added, from the candidate list of candidates not in the winner set. This increase in PAV score for a candidate c is calculated using:

$$w_c = \sum_{i \in N_c} \frac{1}{1 + |W \cap A_i|} \tag{1.5}$$

After k rounds we will have constructed the winner set of size k. It is important to understand that SeqPAV is an *approximation*, and so it might not always compute the same winner set as PAV, as we will see in the following example. Given the preference profile A, which will be presented as a table like the one below, where from now on an **x** will represent that a voter approves a candidate.

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$
0				x	x
1	x		x	x	
2	x	x	x	x	
3				x	x
4	x		x	x	

The winner set of candidates if PAV is used is $\{c_0, c_3, c_4\}$, while if seqPAV is used instead, the resulting winner set changes to $\{c_0, c_2, c_3\}$.

1.4.3 Phragmén's Rule

Phragmén's Rule is another voting rule that we will be studying, together with PAV. This rule is a multi-winner election rule used to select a k-sized committee given approval votes as explained in [2], which considered the voting rules proposed by Lars Edvard Phragmén in [17], [14], [15], and [16]. The goal of this rule is to find the set of k candidates that have a support equally distributed among all the voters. For every winner candidate we will distribute one unit of load among the voters that approve said candidate. We will distribute the loads so that the following constraints are satisfied, given that $l_{i,c}$ is the load distributed by candidate c to voter i:

i. $\sum_{i \in N} \sum_{c \in C} l_{i,c} = k$,

ii. $\sum_{i \in N} l_{i,c} \in \{0,1\}, \forall c \in C$,

iii. $l_{i,c} = 0$ if c is not approved by i

We build a load distribution L, which is an array of two dimensions. The total load of voter i is calculated by:

$$l_i = \sum_{c \in C} l_{i,c} \tag{1.6}$$

Each load distribution corresponds to a winner set of candidates W such that: $W_L = \{$c $\in C : \sum_{i \in N} l_{i,c} = 1\}$, so every candidate in the winner set must distribute a load of one unit among its support. Because the constraint (i) and (ii) are satisfied, we know for sure that the rule will only select k candidates.

The method of calculating this committee will depend on the way that we evaluate a committee's load distribution. There are two main ways of evaluating the distribution, and even though they will not be used in this thesis, we would like to give a brief explanation.

i. **max-Phragmén:** the winner set corresponding to the load that minimizes the maximum voter load is selected.

ii. **var-Phragmén:** the winner set will be selected that corresponds to the load distribution minimizing the sum of squares ($\sum_{i \in N} l_i^2$) of each individual voter load. It is a very interesting voting rule, because by minimizing this sum we are indirectly minimizing the variance of voter loads, and therefore we are balancing the load distribution.

For both cases a tie-breaking rule can be used in the case that two different committees satisfy the selection criteria. A tie-breaking rule can be decisive, as there can be a more preferable committee between the two committees (e.g. y some external criteria).

Along with PAV, Phragmén's Rule also is NP-Hard [2]. This is why we also introduce a sequential method for this voting rule, known as: *Sequential Phragmén's Rule* [2]. Such sequential variant will help us cope with the complexity of the problem, as it can be solved in polynomial time.

1.4.4 Sequential Phragmén's Rule

The main focus of this voting rule is minimizing greedily the maximum voter load. This method was introduced by Phragmén in [17], [14], [15], and [16] and explained in [2].

The algorithm will perform k rounds, adding a new candidate to the winner committee in each round. The candidate selected will be the one than increases the maximum voter load the least, $l_i^{(j)}$ will denote voter i's load after round j has finished. In the initial round, all voter loads are set to 0; i.e., $l_i^{(0)} = 0 \ \forall i \in N$, as there are no candidates in the winner committee. Then, we introduce the candidate into the winner set that has the greatest approval score. This is because after introducing a candidate we must distribute a load of 1 between the voters that approve said candidate, in the way that minimizes the maximum voter load. Therefore, by distributing the single unit of load between the voters that approve the most popular candidate equitably, we get the lowest possible maximum voter load. In the consequent rounds we still choose the candidate that minimizes the maximum voter load, but this time we have to take into account that voters may not have a load equal to 0. We use the following equation to calculate the new maximal load, given that candidate c is chosen and that we are in the round $(j+1)$:

$$s_c^{(j+1)} = \frac{1 + \sum_{i \in N_c} l_i^{(j)}}{|N_c|} \tag{1.7}$$

Then, after calculating the candidate entering the winner committee, we must update the voters' load vector. We will do this in a way that the voters that belong to N_c will have equal loads:

$$l_i^{(j+1)} = \begin{cases} s_c^{(j+1)}, & \text{if } i \in N_c \\ l_i^{(j)}, & \text{otherwise} \end{cases} \tag{1.8}$$

After k rounds, we will get a load distribution and a winner set that corresponds to said distribution. From both equations we can assume that a voter's load will neither decrease or be lower than 0. Additionally, after j rounds have passed the sum of all voter loads will be exactly j, thus constraints i-iii are always satisfied. The sequential variant of Phragmén's Rule is the voting rule most common Phragmén's Rule variant used. For that reason, from now on, whenever we mention Phragmén's Rule in this thesis, it will imply that we are talking abut the sequential variant.

1.5 Related Work

Work on iterative voting has been studied in a recent book chapter by Meir [11]. This chapter gives a survey on the work done on iterative voting, but it does not present any new concepts. As previously mentioned, this chapter focuses on strict orderings of preferences, while this thesis deals only with approval ballots that do not contain any strict ordering between approved candidates. The framework used for iterative voting as well as the concepts of convergence and equilibrium of games are explained. The chapter goes over possible desirable equilibrium properties certain iterative games may have. Those properties include: finite individual improvement property; any game that has a if the corresponding improvement graph with no cycles, weakly-FIP; if from any initial state of the game there is some path in the improvement graph that reaches a Nash Equilibrium (such games are known as weakly acyclic), and restricted-FIP; if from any initial state and any order of players there is some path in the improvement graph that reaches a Nash Equilibrium (such games are known as order-free acyclic).

Additionally, they also define the concept of Dynamic Price of Anarchy and look at the different DPoAs for voting rules such as Borda, Plurality or Veto. This concept is very interesting for our thesis, as it is a method for evaluating the outcome of games with respect to measuring the efficiency of a voting system and how it deteriorates due to the strategic behavior of its voters. They also go through different types of voting heuristics that could be used in iterative games. This part of the Chapter was very useful for us, as we used it as reference for developing our own heuristics in the Chapter 2 of this thesis. The "Threshold" heuristic from that section, is introduced in this thesis but translating the key aspects of it into our approval voting setting. Which voting rules satisfy the equilibrium properties previously introduced is explained with the following results presented:

- FIP is not satisfied by Plurality, Veto, k-approval ($k \geq 2$), Approval or Borda.
- Restricted-FIP is only satisfied by Approval voting.
- Weak-FIP is satisfied by Plurality, Veto and Approval.

In the final conclusions, they stated that iterative voting provides a natural tool to study strategic voting and introduced the possible future work of designing simple iterative mechanisms that improve efficiency and welfare. Moreover, they finish by concluding that with the use of theoretical analysis, we can understand not only the conditions under which convergence is expected, but also the properties of the attained equilibria.

In [10], by Jobst Heitzig and Forest W. Simmons, an explanation is given for the different ballot types used in different voting rules, one of them being Approval Voting. Moreover, they introduce different approaches for modelling individual preferences. In this article, different possible behavioural types that could be used are described and how each behavioural types works depending on the voting rule being used is also explained. This section of the article is used as inspiration for the heuristics described in Chapter 2.

1.6 Contributions

The main contribution made by this thesis is the study of Multi-winner election rules with approval ballots applied to an iterative setting. As previously mentioned, other academic project have studied iterative voting, but using strict ranked preferences. This is not the case in this thesis, as we will not consider ranked preferences, instead we will contemplate only approval preferences, where every voter identifies a set of candidates they approve of. This is because to the best of our knowledge there is not much research on approval preferences applied to an iterative setting. Moreover, contributions made also include the development of the heuristics considered in Chapter 2, which were created using the heuristics described in [11] as a starting point, in addition to the development of the code that simulates the heuristics considered. The concept of Success Probability introduced in Section 2.5 we also created for this thesis. Furthermore, the two first proposed approaches for random voter preference generation of Chapter 4, were also contributed.

1.7 Outline

Now that we have given an introduction, we will follow by explaining briefly the structure and contents of the thesis. First, we will begin with the chapter of Heuristics, where we will introduce and explain different behaviours that can be used by voters to manipulate. Consequently, we will dive deep into the concept of stable states and understand which heuristics guarantee reaching a stable state. In that chapter we will also introduce the concept of Dynamic Price of Anarchy, a notion introduced by Tim Roughgarden in [18]. In Chapter 4, we will present different solutions to the problem of generating random preferences for voters. This will be used in the next chapter, where we will set up experiment to asses the effect caused by the previously introduced heuristics. In those experiments we will create voting games where different variables are changed, such as number of voters, candidate list and winner committee size. However, such experiments will contain only one type of heuristic in each game played. This will not be the case in Chapter 6, where we will set up experiments in which the different heuristics introduced play against each other. Finally, we will end the thesis with our conclusion of the results.

2 Heuristics

In this part of the thesis we aim to model certain behaviors voters could have, when asked whether they want to change their vote. We also aim to model human behavior existent in real life voting environments. Most of these heuristics will be based on the notion of our voters having a myopic behavior, this means that voters only deviate from their current strategy (i.e., change their approval ballot), when they think it will lead to an improvement in utility.

There are two types of possible manipulations:

- **Subset Manipulation**: in voter i's turn, they can either remain with the same approved set of candidates or make it smaller- by disapproving one of them. Voter i can never undo a subset manipulation in subsequent rounds, as there is no subset manipulation that will make its approval set larger and re-approve the candidates through which the voter manipulated.

- **Superset Manipulation**: in voter i's turn, they can either remain with the same approved set of candidates or make it larger- by approving a previously disapproved candidate. Voter i can never undo a superset manipulation in subsequent rounds, as there is no superset manipulation that will make its approval set smaller and re-disapprove candidates that the voter approved in the manipulation.

For the following examples of heuristics for voter's behaviour we will be using the voting rule: Sequential Proportional Approval Voting, explained in 1.4.2. In case of a tie between candidates, we will be using a *lexicographic* tie-breaking rule:

$$c_0 > c_1 > c_2 > c_3 > c_4$$

Additionally, in order to represent deviations we will use the following notation:

- x denotes that a voter used to disapprove a candidate but now has deviated from its current vote by adding said candidate to the approved candidate set.

- ( ) denotes that a voter used to approve a candidate but now has deviated from its current vote by disapproving said candidate.

The motivation for the second heuristic introduced: Threshold Heuristic, was finding first a heuristic which did not have a high complexity. The reason for this was that we wanted to develop a heuristic which did not have a high cognitive complexity and would be used by voters in real life scenarios that do not want to put much effort on computing a better response for the current voting profile. Then, the subsequent heuristics have a higher complexity, as we wanted to model voters that do want to put effort on computing the better response.

2.1 Truthful Heuristic

The first heuristic that we will talk about is the simplest one. In each iteration the voter chooses to vote according to their true preferences. It can be seen as a lazy behaviour, due to the voter does not putting effort into looking for a better response. Instead the voter using this heuristic decides to never deviate from its true preference and never make a strategic decision. A game consisting of just lazy voters will never leave the initial node of the directed graph. This is because, even though there might be possible deviations, no single voter decides to take one of them.

2.2 Threshold Heuristic

With this heuristic, a voter i is able to disapprove a single candidate that has a percentage of total approval score above (or equal to) a certain threshold and already is in the winner committee. This is based on the idea that by disapproving a popular candidate, it is very likely that said candidate still makes it into the winner set even without i's approval. Additionally, it gives the other approved candidates a larger weight in the voting rule, i.e. they have a higher chance of entering the winner set. This heuristic only performs subset manipulations. A voter using the Threshold Heuristic will not be able to disapprove all their candidates, to avoid voters submitting empty approval ballots.

An important aspect about this heuristic is that deviations are computed without simulating the outcome of the election after the deviation is made. Hence, Threshold voters do not take into account which voting rule is being used. This is crucial, as it means that Threshold voters might deviate even in voting rules that are strategy-proof. A voting rule is said to be strategy-proof, if voters do not benefit from misreporting their true preferences.

2.2.1 Implementations of the Threshold Heuristic

Two variants of this heuristic have been implemented:

- **First Candidate Above Threshold:** a voter disapproves the first found approved candidate that has an approval score above the threshold and is in the winner committee. If none are found, then the voter does not deviate in this iteration. Submits the new vote, otherwise.

- **Highest Candidate Above Threshold:** a voter disapproves disapproves the candidate that has the highest approval score above the threshold and is in the winner committee. If none are found, then the voter does not deviate in this iteration. Submits the new vote, otherwise.

The choice of the threshold value is very important. It models how safe the voter wants to play. The closer the threshold is to 100% the safer the voter plays, hence the less this heuristic will manipulate. This is because there are usually few candidates that have an approval score above high thresholds. However, a threshold too low can be counterproductive, as seen in the example presented in 2.2.3.

2.2.2 Example of increase in utility

There are 5 voters and 5 candidates, the winner set computed must have a size of 3 ($k = 3$). Voter 4 is using the Threshold Heuristic and the rest are using the Truthful Heuristic.

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$
0	x	x	x		
1	x	x		x	x
2	x			x	
3	x		x	x	x
4	x	x	x		x

From this true preference profile the winner set under seqPAV is $\{c_0, c_1, c_3\}$. The current total utility (social welfare) is 11. When it is voter 4's turn, if she decides not to deviate, the winner set will remain the same and its utility will be 2. But, applying the Threshold Heuristic and using a threshold of 80 percent, she can disapprove the candidate c_0, a popular candidate in the winner set and with an approval score above the threshold. If 4 applies this deviations the new voting profile after its round will be:

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$
0	x	x	x		
1	x	x		x	x
2	x			x	
3	x		x	x	x
4	( )	x	x		x

This new voting profile outputs the winner set $\{c_0, c_1, c_2\}$, as the rest of the voters are using the Truthful Heuristic and there are no candidates that 4 approves with an approval score above the threshold, the game ends with this new winner set. Voter 4's utility has gone up by 1 and the social welfare of the game has remained the same. This has happened as a result of the utility for voters 1 and 2 decreasing by 1 each along with the utility for voters 0 and 4 increasing by the same amount.

2.2.3 Example of decrease in utility

There are 5 voters and 5 candidates, the winner set computed must have a size of 3 ($k = 3$). Voter 2 is using the Threshold Heuristic (with a threshold of 40%) and the rest are using the Truthful Heuristic.

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0		x			
1		x		x	x
2	x				x
3		x			
4		x	x		

From this true preference profile the winner set is $\{c_1, c_4, c_0\}$. The current total utility (social welfare) is 7. Its voter 2's turn, because she is using the Threshold Heuristic she decides to disapprove candidate c_4; a candidate in the winner set and with an approval score equal to 40%. This is the new voting profile:

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0		x			
1		x		x	x
2	x				( )
3		x			
4		x	x		

After this deviation the new winner set is $\{c_1, c_0, c_2\}$. Voter 2's utility has decreased by 1, and the social welfare has decreased to 6. Had voter 2 used a higher threshold, she would not have deviated and its utility would still be 2.

2.3 Iterative Heuristic

The Iterative Heuristic takes it a step further than the previous one. When voters apply the Threshold Heuristic, they do not evaluate whether their utility will increase/decrease in the following round, but they make the assumption that they will probably see an increase in utility after deviating. This is an advantage as it makes voters decide much faster, but (as we have previously seen) it might lead to a worse outcome than before. In Iterative Heuristic, this will never be the case, as the voter will compute their new utility after the deviation and submit the new vote if and only if there is a strict increase in utility. There are two variants of this heuristic: *Iterative Single Heuristic* and *Iterative Up to J Heuristic*.

What is also important to point out about this heuristics, is that the utility of the voters using it will only decrease because of the deviations made by other voters and not the deviations made by themselves.

2.3.1 Iterative Single Heuristic

This heuristic performs the following steps:

1. The voter calculates their utility given the current winner committee.

2. Iterates through each candidate and looks for the one that yields the maximum gain in utility by inverting their approval value.

3. If none is found, then the voter decides not to change its vote. Otherwise, the voter submits a new vote with said candidate's approval value inverted.

The utility of a voter can only increase during the game, unless other voters deviate as well. The voter is able to perform both subset (inverting the approval score of a candidate from 1 to a 0) and superset (inverting the approval score of a candidate from 0 to a 1) manipulations. But can only increase/decrease their approval set by one candidate.

In order to calculate the gain of utility, this heuristic must simulate the outcome of the voting rule, given that it performs a deviation, before taking the decision. Therefore we are exchanging a higher complexity for more intelligence, that will result in the voter taking a more educated decision compared with the Threshold Heuristics, Section 2.2.

2.3.2 Iterative Up to J Heuristic

The variant Iterative Up to J Heuristic is an extension of the previous heuristic that allows voters to make up to j changes to their vote, instead of only making deviations of a single candidate's approval value. Where $j \in \{1, 2, ..., |C|\}$. This can be a great advantage for them, as it gives more flexibility to the voter. However this comes at a prize: complexity increases considerably. Meaning that voters using this heuristic will take longer on deciding how to deviate. For any j chosen the voter must look for the deviation with the highest utility increase from all the set of possible deviations. This set will consist of all possible deviations that make up to j changes to the vote. The result of increasing j is not only increasing the amount of possible deviations considered but also increasing the complexity of the problem.

If $j = |C|$, then we guarantee that this heuristic will always find the best response to the current state of the game, as it will take into consideration all possible manipulations. Using a $j \in \{1, 2, ..., |C| - 1\}$ we can guarantee that the heuristic will find a better response, but not a best response. Additionally, if $j = 1$ the heuristic is exactly the same as Iterative Single Heuristic, as they both look at the same set of possible deviations and take the one that yields the highest increase in utility.

Example of increase in utility

The following matrix represents a true preference profile. There are 5 voters and 5 candidates, the winner set computed must have a size of 3 ($k = 3$). Voter 3 will be using Iterative Up to J Heuristic, with j=2. The rest of the voters will be using the Truthful Heuristic.

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0	x	x	x		x
1		x		x	
2	x				x
3	x	x		x	
4	x	x	x		x

Given this voting profile, the current winner committee is $\{c_0, c_1, c_4\}$. It is voter 3's turn and its current utility is 2. Using the Iterative Up to 2 Heuristic voter 3 decides to manipulate by disapproving candidates c_0 and c_1. The new voting profile is given by:

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0	x	x	x		x
1		x		x	
2	x				x
3	( )	( )		x	
4	x	x	x		x

From this voting profile, the new winner committee is $\{c_0, c_1, c_3\}$. Voter 3's utility has increased by 1. All the candidates voter 3 approves are in the winner committee. This manipulation would not have been possible with the Iterative Single Heuristic, and there is no single change in voter 3's vote that would have yielded a higher increase in utility. It is also a good example to convey that Iterative Single Heuristic does not always compute the best response.

2.3.3 Making Iterative Heuristic Faster

Both of the heuristics explained in this section take a lot of time to compute a new vote, or check if said vote even exists. However, we can take advantage of the fact that voters do not need to compute a new vote given that they have already computed (and not found) a new improving vote for that given state. So, whenever a voter using an Iterative heuristic cannot improve the outcome of the game by deviating, she will save the current state of the game. If the voter does deviate, she will not save the current state of the game, as she might still want to deviate more in subsequent rounds. In its future turns, if no changes are made by other voters (i.e. the state of the game is equal to the one the voter saved), then she automatically decides not to deviate, as she already knows that the outcome cannot be improved by manipulating with the heuristic she is using.

2.4 Trial-and-Error Heuristic

With this heuristic, the voter has the ability of going back to its initial vote, if the deviations the voter has done have led to a decrease in utility. This decrease in utility will be caused by the manipulations of other voters, as it will only perform manipulations that increase its utility directly. The deviation made by voters using this heuristic will be found by selection one deviation at random from the set of deviations consisting of changing one single approval score of the candidate list. In each round, this heuristic will check whether its current utility is lower (or equal) than the initial one (given that the voter has deviated in a previous round).

- If its current utility is lower (or equal) to the initial utility, the voter will submit the initial vote, undoing all the previous manipulations.
- Otherwise, if the current utility has not decreased (or no deviations have been previously made) the voter will choose a random deviation from all possible deviations of one single approval score change. Then, submit the new vote if and only if the utility increases directly after such change.

The complexity of this heuristic is much lower that the one of Iterative Heuristic, as this heuristic only checks for the utility increase of one deviation. Thus, Trial-and-Error voters will simulate the voting game and calculate the winners of the election just once, to get the change in utility after the randomly selected deviation. However, this heuristic does not always compute the best response, given the current state of the game, as it selects a random deviation not from all possible deviations but from a subset.

2.5 Factional Heuristics

A faction consists of a set of voters *f*, that have the same preferences and the same initial vote. There can exist a voter with the same preferences as the faction, that does not belong to it. With a factional heuristic voters can deviate from the voting profile as a group. If a voter i discovers a better reply for the faction to the current state of the game she signals other voters in the faction, who will change to i's vote once their turn comes. The better reply for the faction can be found using any heuristic mentioned previously (Iterative Single Heuristic will be used for now). Before talking about the different implementations of this heuristic we will introduce the concept of *Success Probability.*

2.5.1 Success Probability

As we have previously mentioned, one of the aims of this thesis is modelling human behaviour in voting games. One could imagine real life scenarios in which after a voter has decided for its faction to manipulate in a certain way, the voter needs to convince the rest of the faction that the deviation she has computed is a correct one. For this we will denote the probability that a voter has of successfully convincing the rest of the faction to deviate as the *Success Probability* (SP_f) of the faction. For this two constraint should be satisfied:

- The bigger the faction the lower the success probability, i.e. the harder it is to convince everyone.

- A faction of size 1 must have a success probability of 1, as a voter must always be able to convince itself.

This is the equation we propose to model the *Success Probability*, where f is the set voters that belong to the faction:

$$SP_f = \frac{1}{\sqrt{|f|}} \tag{2.1}$$

Using this function to calculate the Success Probability will output very small values for large factions. However for a very large faction it is very likely that at least one voter will have the chance to convince the rest of the faction.

Theorem 2.1. *As $n = |f|$ tends to infinity, the probability that at least one voter will convince the rest of the faction is 1.*

Proof. The probability that event a occurs (where a means that a voter convinces the rest of the faction) is given by:

$$P(X = a) = \frac{1}{\sqrt{|f|}} =: p \tag{2.2}$$

Hence, the probability that the faction is not convinced (event b) is given by:

$$P(X = b) = 1 - \frac{1}{\sqrt{|f|}} = 1 - p \tag{2.3}$$

Y will denote the number of times a faction is convinced: $Y \sim bin(n, p)$, where $n = |f|$.

$$P[Y \geq 1] = 1 - P[Y = 0] = 1 - (1 - p)^n \tag{2.4}$$

As the number of members of the faction tends to infinity:

$$\lim_{n\to\infty} P[Y \geq 1] = 1 - \lim_{n\to\infty}((1 - \frac{1}{\sqrt{n}})^n) = 1 - \lim_{n\to\infty} n^{-\frac{n}{2}} = 1 - 0 = 1$$

□

2.5.2 Same Vote Factional Heuristic

In this implementation of the Factional Heuristic all voters in the faction must have the same vote when the game reaches a stable state. If a voter i in the faction finds a vote that can yield a better utility than the current one (if every voter in the faction submits said vote), then the voter signals the rest of the voters to change to this new vote. In the subsequent rounds, voters belonging to this faction will deviate to the new vote computed by voter i. When everyone has changed to the new vote, then they will look for a new better response. The faction changes to the new computed vote in a sequential way (each member changes in its turn), as they are not allowed to change all at once.

The decision diagram in Figure 2.1 represents the behaviour taken by a factional voter i using this heuristic during a game.

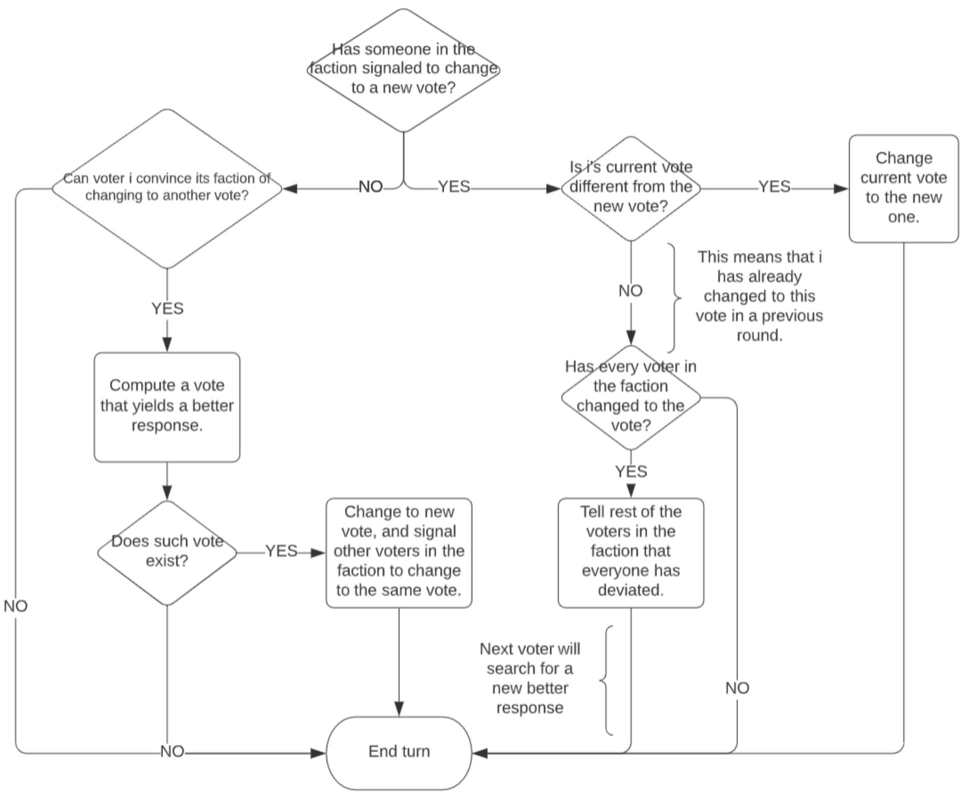

Has someone in the faction signaled to change to a new vote?
Can voter i convince its faction of changing to another vote?
NO
YES
Is i's current vote different from the new vote?
YES
Change current vote to the new one.
This means that i has already changed to this vote in a previous round.
NO
YES
Compute a vote that yields a better response.
Has every voter in the faction changed to the vote?
YES
Does such vote exist?
YES
Change to new vote, and signal other voters in the faction to change to the same vote.
Tell rest of the voters in the faction that everyone has deviated.
NO
Next voter will search for a new better response
NO
NO
End turn

Figure 2.1: Decision Diagram Same Vote Factional Heuristic

In this setting a faction does not consider other voters manipulating, that do not belong to the faction. It can happen that a voter from the faction signals the faction to change to a new vote and an outsider to the faction deviates before everyone in the faction has changed to the new better response. This may lead to the better response computed making utility for the faction decrease. Which is caused by the state of the game not being the same as the one when the better response for the faction was computed. To cope with this problem a new heuristic is introduced (section 2.5.3): *Same Vote Reactive Factional Heuristic*.

Example of increase in utility

The following matrix represents a true preference profile. There are 5 voters and 5 candidates and the winner set computed must have a size of 3 ($k = 3$). Voters 0 and 1 belong to the same faction, and the rest of the voters are using the Threshold Heuristic, with a threshold value of 0.6.

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0	x	x	x		
1	x	x	x		
2		x		x	
3	x		x		x
4	x	x			

The initial winner committee is $\{c_0, c_1, c_2\}$. After the initial round the faction does not want to deviate as they already have a maximum utility of 3. However, in subsequent rounds voters 2, 3 and 4 do deviate (each one in their own turn), which results in the following voting profile:

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0	x	x	x		
1	x	x	x		
2		( )		x	
3	( )		x		x
4	x	( )			

The resulting winner committee from this new voting profile is $\{c_0, c_2, c_3\}$. We can see that the utility of the faction has decreased to 2. In voter 0's turn, she signals voter 1 to disapprove candidate c_0. This is because if all the faction deviates as voter 0 says the winner set will be $\{c_2, c_0, c_1\}$. Therefore, voter 0 deviates, resulting in the following voting profile:

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0	( )	x	x		
1	x	x	x		
2				x	
3			x		x
4	x				

After this it's voter 1's turn and she deviates in the same way as voter 0 has signaled. The new voting profile is:

Voters/Candidates	c_0	c_1	c_2	c_3	c_4
0		x	x		
1	( )	x	x		
2				x	
3			x		x
4	x				

The new winner committee in this next round is $\{c_2, c_0, c_1\}$, and utility for the faction has gone back up to 3. Had voter 1 not followed voter 0's signal, then the winner set would have remained as: $\{c_0, c_2, c_3\}$, and their utility as 2. Thus with this example we show that voters have additional opportunities of improving their utility when belonging to a faction, that would not have been possible had they been on their own.

2.5.3 Same Vote Reactive Factional Heuristic

In this heuristic the faction must still have the same vote for all of its voters when the game ends. Whenever a voter i belonging to the faction receives a signal from another voter of the faction, to change to its vote, i checks whether there has been a deviation by some voter j (that does not belong to the faction) after the signal to change vote was sent. This is done by storing set the approval ballots from the voters that do not belong to the faction when the signaled vote was computed (denoted by: A_{-f}), and comparing it with the set of votes from the voters that do not belong to the faction in each faction's turn. If an outsider to the faction j has deviated, then the voter i (where i belongs to the faction) checks whether the predicted future utility for the faction (given that they all change to the signaled vote) has changed. There are two possibilities:

1. The predicted future utility if all the faction changes to the signaled vote has **decreased** compared to the current faction's utility. This means that the deviation made by the outsider voter j has affected the faction negatively. As a consequence deviating to the signaled vote would lead to a decrease in utility. Then, voter i signals the rest of the faction to undo the computed manipulation and remains with its current vote, because i knows that this computed manipulation is no longer beneficent for the faction. Once every one is back to the previous vote the heuristic starts over, looking for a new better response to the current state of the game.

   We have chosen for this heuristic to wait until everyone has gone back to the previous vote before computing a new deviation, not only because we wanted this heuristic to remain as simple as possible, but also because we think it is advantageous for the faction to wait a few rounds before computing a new deviation. The reason for this is that if voters not belonging to the faction are going to deviate while the faction is going back to the previous vote, we think it is better for the faction to let those outsiders manipulate first, before computing a new better response.

2. The predicted future utility if all the faction changes to the signaled vote has **increased (or remained the same)** compared to the current faction's utility. This means that j's deviation was not disadvantageous for the faction. As a consequence deviating to the signaled vote will not lead to a decrease in utility. Then, voter i changes its own vote to the signaled one, and updates the A_{-f} stored to the current one.

Like in the previous heuristic, whenever the last voter changes to the signaled vote, she tells the rest of the faction that everyone has deviated to such vote. After which the next voter of the faction computes a new deviation. Additionally the *Success Probability* is still used to convince the faction of deviating, every time a voter computes a new better response.

Figure 2.2 shows the decision diagram representing the behaviour of a factional voter i using this heuristic during a game.

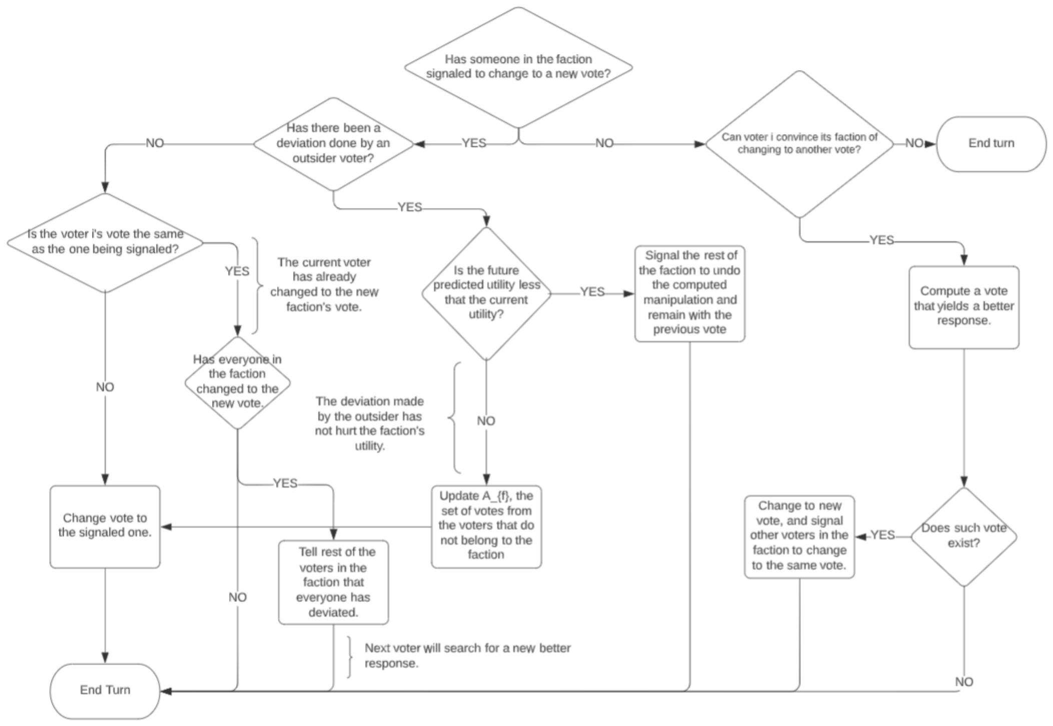

Has someone in the faction signaled to change to a new vote?
YES
NO
Has there been a deviation done by an outsider voter?
NO
YES
Can voter i convince its faction of changing to another vote?
NO
End turn
YES
Is the voter i's vote the same as the one being signaled?
YES
The current voter has already changed to the new faction's vote.
Is the future predicted utility less that the current utility?
YES
Signal the rest of the faction to undo the computed manipulation and remain with the previous vote
Compute a vote that yields a better response.
NO
Has everyone in the faction changed to the new vote.
The deviation made by the outsider has not hurt the faction's utility.
NO
YES
Change vote to the signaled one.
Update A_{f}, the set of votes from the voters that do not belong to the faction
Tell rest of the voters in the faction that everyone has deviated.
NO
Change to new vote, and signal other voters in the faction to change to the same vote.
YES
Does such vote exist?
Next voter will search for a new better response.
End Turn
NO

Figure 2.2: Decision Diagram Same Vote Reactive Factional Heuristic

Example of decrease in utility after deviation

Now we will show an instance of a game where a faction is signaled to change its vote, but because an external voter deviates during the transition to the new vote, it becomes unfavorable. The game consists of 10 voters, 5 of which (voters 0 to 4) belong to the same faction and use the Iterative Single Heuristic to compute better responses. The other 5 voters are using the Threshold Heuristic (with threshold = 0.8). The winner committee has a size of 4 (k = 4) and there is a set of 8 alternatives (m = 8). The following voting profiles shows the initial true preferences of each voter.

Voters/Candidates	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7
0		x	x	x		x	x	x
1		x	x	x		x	x	x
2		x	x	x		x	x	x
3		x	x	x		x	x	x
4		x	x	x		x	x	x
5		x			x	x		
6					x	x		
7		x			x			
8	x		x	x	x	x	x	
9		x		x				x

The initial winner committee is: $\{c_1, c_5, c_3, c_4\}$.Voter 4's turn begins, she belongs to the faction. Using Iterative Single Heuristic, 4 calculates that by deviating as a group and disapproving candidate c_3, the faction can increase their utility from 3 to 4 and change the winner committee to: $\{c_1, c_5, c_2, c_7\}$. Thus, voter 4 deviates, signals the rest of the faction with the new computed vote and stores the current voting profile for voters 5 to 9.

Voter/Candidates	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7
4		x	x	( )		x	x	x

However, before all the faction changes to the signaled vote, voter 8 deviates. Candidate c_5 has 8 votes, and is disapproved by voter 8. This changes the winner committee to: $\{c_1, c_5, c_3, c_2\}$.

Voter/Candidates	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7
8	x		x	x	x	x	( )	

Then comes voter 2's turn. Voter 2 sees that the faction wants to change to a new vote computed by voter 4. Voter 2 does not deviate, as she detects that an outsider has deviated as well and their current utility (4) is greater than the new predicted future utility if the faction deviates together (3). This is because the future predicted winner set will be: $\{c_1, c_5, c_2, c_4\}$, if the manipulation is not ceased. Accordingly, voter 2 does not change its vote and signals all voters in the faction (that have manipulated) to change back to the previous vote.

3 Stable States and Dynamic Price of Anarchy

As it has been previously mentioned in this thesis, reaching a Nash Equilibria is reaching a node in the game with no outgoing edges. Meaning that for all voters there exists no possible deviation that will increase the current utility. However we must introduce the concept of *Stable State*.

3.1 Stable states

A Stable State is a relaxation version of Nash Equilibrium, i.e. all Nash Equilibria are stable states but not all stable states are Nash Equilibria. A stable state is a state in our iterative voting game, where no voter (after applying one of the heuristics previously described) wants to deviate from its current vote. The main difference is that in these states, the voters are only taking into account the deviations that their heuristics allow. So, even though there is a better response for a voter to the current state of the game, the voter might not deviate to that better response as she cannot compute it. This is because the heuristic being used by the voter is considering only the allowed actions and not all possible deviations. We have introduced this concept, as our voters are not always taking the best response possible. It will be also represented as a node with no outgoing edges and the game will end when a stable state is reached, because no single voter will decide to deviate from that voting profile that corresponds to the stable state. From this we must ask the question:

Which heuristics guarantee always reaching a stable state from the initial state, assuming that at least two voter apply said heuristic?

In order to answer this question we divide our problem into two possibilities:

1. **The voters can only deviate using subset manipulations:**

   For this first case, during a voters' turn, they can either stay with the same approved set of candidates or make it smaller. Voters can't go back to a bigger approval set, meaning that once they make a manipulation to their preferences they can never undo it. Each node (voting profile) is only visited once during the course of the game. Therefore, we can assume that our graph is acyclic and we will always reach a node with no outgoing edges, i.e. *we can guarantee always reaching a stable state.* This first possibility is the case for the Threshold Heuristic.

2. **The voters can deviate using both subset and superset manipulations:**

   In this case, voters can either stay with the same approved set of candidates, make it smaller or make it larger. This means that if they make a manipulation to their preferences, they can always undo it and return to previous votes. It may happen that a node is visited more than once during the course of the game. Therefore, our graph no longer has to be acyclic and *we cannot guarantee reaching a stable state.*This second possibility is the case for the rest of the heuristics described in chapter 2.

Moreover, there is the third option of heuristics that only allow superset manipulations. This third option also guarantees reaching a stable state, but will not be considered as there are no heuristics considered in this thesis that follows such behaviour.

Given the previous two possibilities we now know which heuristics guarantee reaching a stable state:

Heuristic	Guarantees reaching a Stable State?
Truthful Heuristic	Yes
Threshold Heuristic	Yes
Iterative Heuristics	No
Trial-and-Error Heuristic	No
Factional Heuristics	Depends on the heuristic being used.

As we have previously explained, Iterative Up To J Heuristic (with $j = |C|$) is very powerful, because it always computes the best response possible. This is important as we can now say that in a game containing only voters with this heuristic, reaching a stable state will be equivalent to reaching a Nash Equilibrium.

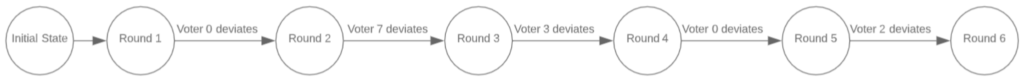
For Trial-and-Error the concept of stable state is not that useful, the reason being that they check one deviation per round and said deviation is drawn at random. So even though a Trial-and-Error voter might say she does not want to deviate and the game ends, if that voter is asked again in a hypothetical subsequent round she might be inclined to deviate, as the deviation drawn in that round increases her utility. Hence the final states reached whenever this heuristic is being used is not guaranteed to be stable.

3.2 Unreachable Stable State Example

Now we will present an example of a heuristic using both types of manipulations that does not reach a stable state: The following matrix represents the truthful preferences of 10 voters.

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$	c$_5$	c$_6$	c$_7$	c$_8$	c$_9$
0	x		x	x	x				x	
1	x	x			x	x				
2		x		x	x	x	x			x
3		x	x			x		x		x
4		x	x					x	x	
5							x		x	
6					x			x		
7				x	x		x			
8		x			x	x				
9				x		x	x			

For this game we will be using the voting rule: *PAV*. A winner committee of size 5 will be selected from 10 possible candidates. All of the voters will be using Iterative Single Heuristic. The initial winners are $\{c_1, c_4, c_5, c_6, c_8\}$. From this initial state of the game, the voters deviate in the following order of turns:



As a consequence, the voting profile changes to:

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$	c$_5$	c$_6$	c$_7$	c$_8$	c$_9$
0	x		x	( )	( )				x	
1	x	x			x	x				
2		x		x	x	( )	x			x
3		( )	x			x		x		x
4		x	x					x	x	
5							x		x	
6					x			x		
7				x	( )		x			
8		x			x	x				
9				x		x	x			

In round 6, before voter 5 deviates the current winners are: $\{c_1, c_2, c_4, c_5, c_6\}$, making its current utility: **1**. Voter 3's current utility is: **3**. Voter 5 applies the Iterative Single Heuristic and discovers that by approving candidate c_7, his utility goes up to 2. However by doing so, voter 3's utility has gone down to 2. The new winners are $\{c_4, c_5, c_6, c_7, c_8\}$, and the new voting profile:

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$	c$_5$	c$_6$	c$_7$	c$_8$	c$_9$
3			x			x		x		x
5							x	x	x	

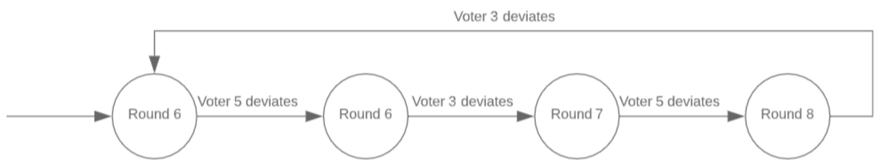
After voter 5's deviation, no matter what voter you draw in round 7 they will all decide to remain with their current vote, except voter 3. Voter 3 had previously manipulated by disapproving candidate c_1 in round 3. Now that its utility has gone down, she decides to undo the previous manipulation by putting c_1 back in its approved set of candidates. Voter 3's utility goes back up to 3, and voter 5's utility back down to 1. This is because the winner set changes to: $\{c_1, c_2, c_4, c_6, c_7\}$, and the voting profile to:

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$	c$_5$	c$_6$	c$_7$	c$_8$	c$_9$
3		x	x			x		x		x
5							x	x	x	

In this round of the game, no matter what voter you draw they will all decide to remain with their current vote, except voter 5. It will undo its previous manipulation (by disapproving candidate c_7) and will make the new winner set: $\{c_1, c_2, c_4, c_6, c_7\}$. By doing so its utility becomes 2, as well as for voter 3. The new voting profile is:

Voters/Candidates	c$_0$	c$_1$	c$_2$	c$_3$	c$_4$	c$_5$	c$_6$	c$_7$	c$_8$	c$_9$
3		x	x			x		x		x
5							x	( )	x	

In the current round it also happens that no matter what voter you draw they will all decide to remain with their current vote, except voter 3. Voter 3 will deviate by disapproving candidate c_1, which will make the state of the game the same as in round 6, as a consequence the actions taken by both voters in all future rounds are exactly the same, and the game will never reach a stable state. The loop can be seen in the following figure:



Had we used another voting rule such as seqPAV or Phragmén's, the game would have reached a stable state. It is important to note that cycles in those two other voting rules also exist. Games in which a stable state is not reached with seqPAV or with Phragmén's will be presented in Chapter 5.

3.3 Dynamic Price of Anarchy

We will set up experiments to asses the behaviour of different heuristics. For that reason we need to measure the quality of the stable state reached. We will do this by means of *Dynamic Price of Anarchy* (DPoA). Using this concept we will compare the social welfare of voters in the initial phase of the game (i.e. when they submit their true preferences) with the social welfare at the end of the game (i.e. when they reach a stable state). By doing so we create a ratio between the quality of the outcome of the game (after a series of deviations) with the outcome of the game under truthful voting, which refers to a truthful voting outcome. The Dynamic Price of Anarchy of a game G with initial winner committee W_I, final winner committee W_F, candidate list C and the set true preferences TP is, calculated by:

$$dPoA(G) = \frac{sw(W_I, TP)}{sw(W_F, TP)} \tag{3.1}$$

Here $sw(W, TP)$ denotes the total utility of all voters (social welfare), it is calculated by 1.2.

Whenever we finish a game with a dPoA above 1, we know that the deviations made by the voters throughout the game have led to a worse outcome in relation to social welfare, i.e. the total utility of the voters has decreased. If the game finishes with a dPoA of 1, it does not always mean that the final winner committee is the same as the initial one, but it does mean that the total utility has remained constant. If the output of the game has a dPoA below 1, then we have a game in which the deviations made by the voters led to a increase in total utility. This is possible as in is not the case neither for PAV nor Phragmén's Voting Rule that their main and exclusive goal is to increase the total utility of the voters.

4 Random Preference Generation

As we want to set up experiments to assess the impact of each heuristic in this voting environment, we must first talk about a way of generating random preferences for our voters. Three possible approaches are suggested, the third of of which taken from [9]. We think the first two approaches are simple ways of generating random voters' preferences, but in the end we recommend using the third approach as it is a better and well-known solution for this problem.

4.1 Normal Distribution for Number of Approved Candidates

In this first solution we will create the random preferences for each voter, by drawing from a normal distribution the number of j candidates a voter approves. Then, selecting uniformly at randomly from the m possible candidates j of them. The normal distribution used for this will have the mean in 5 and a standard deviation of 2.15. The value drawn (j) will calculate the number of approved candidates by calculating: $\frac{j}{10} * m$, and rounding it up to the nearest integer. We have chosen these values, because the voting profiles they generate are are close to some real life databases of approval preference profiles. We will set limits on both sides of the distribution: if the number is larger than 10, then 10 approved candidates will be chosen and if the number is smaller than 0, then 0 approved candidates will be chosen. After doing this for all voters, we will build a voting profile consisting of those preferences to use in the initial phase of the game.

The problem with this method of generating preference profiles, is that as the number of voters increases, the approval score of a candidate will tend to 50%, rendering heuristics like Threshold Heuristic (with a high threshold) useless, as there will be no candidate with an approval score above the threshold. Therefore, for the future experiments where we increase the number of voters we must introduce a new way of generating preferences.

4.2 Normal Distribution for Number of Approving Voters

For this second approach, we will also be using a Normal Distribution with mean at 5 and standard deviation of 2.15. We have also chosen these values as we think they output to a certain extent "realistic" preference profiles. However, we will now use it to calculate the number of voters that approve each candidate. Using $\frac{j}{10} * n$ (where

$n = |N|$ and j is the number drawn from the normal distribution), and rounding it up to the nearest integer we calculate the number of voters that approve a certain candidate. By doing this for every candidate we are building are random initial voting profile for our system. In the experiments in which we increase the number of voters, the number of candidates will remain the same. Therefore, with a large N it will not happen that most of the candidates are approved by 50% of the voters.

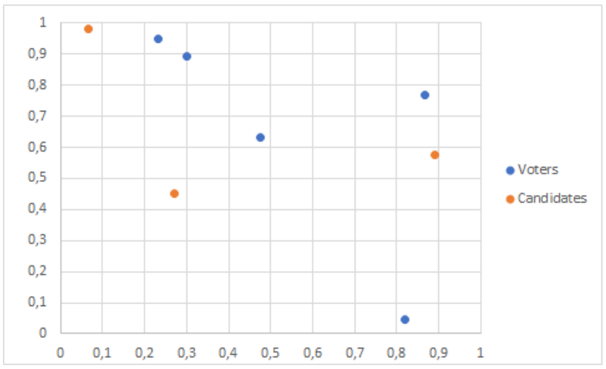
On the other hand, this will not be true when we conduct experiments maintaining the number of voters constant but increasing the candidate list. In this case, we will encounter the problem of our candidates having an approval score that tends to 50%, as the candidate set increases.

4.3 Spatial Euclidean Model

The third approach for preference generation that we provide uses an Euclidean space [0,1]x[0,1]. We represent voters and candidates as points in this space. For this, we first assign to each voter/candidate a random point in the space (drawn uniformly at random). Hence we are creating $|C| + |N|$ random points. After this we compute the distance form each voter to each candidate, and we say that a voter approves a candidate if the Euclidean distance between both points is smaller that some radius r (a parameter that can be changed: $r \in [0, 1]$). Otherwise, the voter does not approve the candidate. The euclidean distance is calculated by:

$$d(v, c) = \sqrt{(x_v - x_c)^2 (y_v - y_c)^2} \tag{4.1}$$

Here is an example of a Spatial Model with 5 voters, 3 candidates and r = 0.6:



This Spatial Model outputs the following preference profile:

Voters/Candidates	c_0	c_1	c_2
0			x
1			x
2	x	x	
3	x	x	
4	x	x	x

5 Experiment Set Up and Analysis of Results

In this chapter we have set up different experiments for each heuristic and each voting rule: sequential PAV and Phragmén's sequential Voting Rule. In each experiment we have chosen an experimental variable (number of voters, number of candidates or committee size) for which we have increased from an initial value to a final value, while setting the rest of them to a constant value. Therefore, three experiments have been conducted for each heuristic and each voting rule:

1. Increasing number of voters.
2. Increasing candidate list.
3. Increasing committee size.

50 instances of the game have been played for each value of the experimental variable being changed, creating a different preference profile for each game played. We have used the Spatial Model for the generation of random approval preferences (with $r = 0.6$). This value was chosen, as after experimenting with different values we concluded that 0.6 generated suitable preference profiles for our experiments. Through these experiments we want to study not only the interactions between the heuristics and voting rules, but also the impact each heuristic has on the outcome of the game. We will also set a limit of 5000 rounds for the games that do not reach a stable state and run into a cycle.

In the following sections we will include graphs to illustrate the results, to avoid this chapter being lengthy and not concise we will only be including the graphs important for each chapter. However, all the results for our experiments of the thesis, as well as the software used to run such experiments, can be found in: `shorturl.at/mtzV6`.

5.1 Threshold Heuristic

We have set up an environment where every single voter will use the Threshold Heuristic (Highest Candidate Above Threshold variant). Each experiment was executed for three different threshold values: 60%, 80% and 95%.

Before going through and analyzing the results it is important to mention, that all the games played with this heuristic reached a stable state. This coincides with the results from chapter 3, which said that heuristics allowing only subset manipulations are guaranteed to reach a stable state.

5.1.1 Increasing the Number of Voters

For the number of voters n, we have chosen the values $n \in \{10, 25, 40, 55, 70, 85, 100\}$. The set of candidates had a size of 10 ($m = 10$) and the winner committee had a size of 4 ($k = 4$). After computing all the games for each voting rule, the following results were obtained.

As seen in Figures 5.1 and 5.2, there was an increase in average deviations as the number of voters increased. Consequently, the average time it takes to play a game as well as the average number of iterations needed to reach a stable state also increased (Figures 5.3, 5.4, 5.9, and 5.10). This is because increasing the number of voters means that there are more of them willing to deviate. We can also see from the results that the lower the threshold, the more deviations being made on average per game by the voters. The reason for this is that the chances of a candidate being approved by 60% of the voters are much higher than being approved by 80% or 95%. Hence, voters using a lower threshold percentage, have more feasible options of candidates to be disapproved.

Of course, as the number of voters increased, so did the average social welfare (Figures 5.5 and 5.6). However, it is important to point out that the average social welfare of the threshold of 60% increased much less than the other two. This can be explained because there are many more deviations being done by the voters, thus the winner committee goes through many changes throughout the game. Hence, the final winner committee at the end is almost always different to the one at the start. Since local deviations done by these kind of voters voters are usually harmful for their own utility and for the total utility of the voters. Thus, the social welfare is decreasing the more deviations are being made. This would also explain why the average Dynamic Price of Anarchy is much higher for the 60% threshold (Figures 5.7 and 5.8) compared to higher thresholds.

Knowing that low thresholds lead to more deviations and more changes in the winner committee, we can understand why the average percentage of voters *strictly* better off with the final winner set was the highest throughout all values of n, compared to the other higher thresholds. The reason for this is that the games played with a higher threshold (like 90%) do not deviate as much from the initial voting profile, and so the final winner committee is much closer to the initial one. Thus, the average percentage of voters *strictly* better off with the final winner committee is never above 0.5%. Therefore, the Dynamic Price of Anarchy for 90% almost never goes higher than 1.02 no matter which value of n we choose.

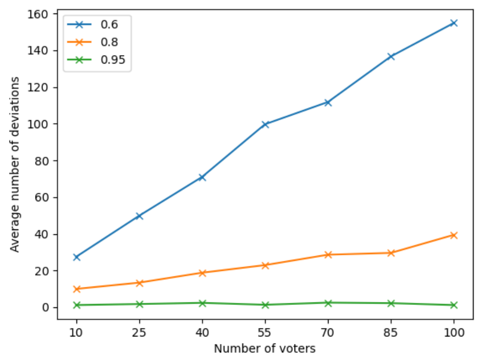


Figure 5.1: Number of Deviations as N increases, SeqPAV.

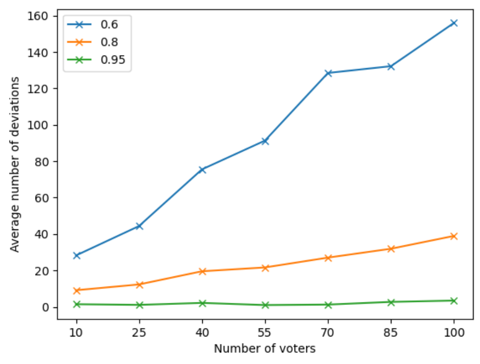


Figure 5.2: Number of Deviations as N increases, Phragmén's Rule.

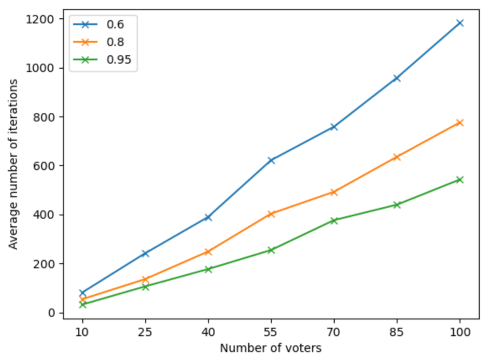


Figure 5.3: Number of Iterations as N increases, SeqPAV.

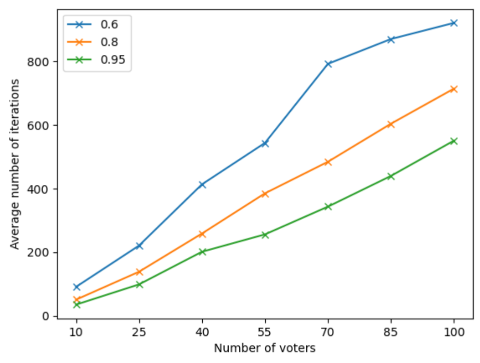


Figure 5.4: Number of Iterations as N increases, Phragmén's Rule.

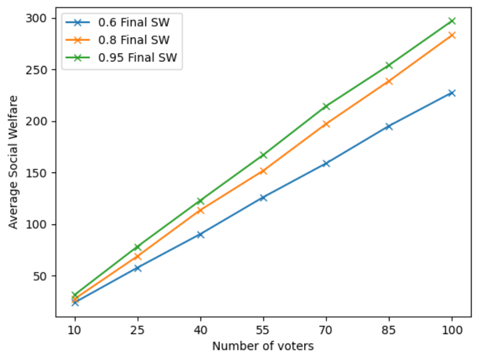


Figure 5.5: Social Welfare as N increases, SeqPAV.

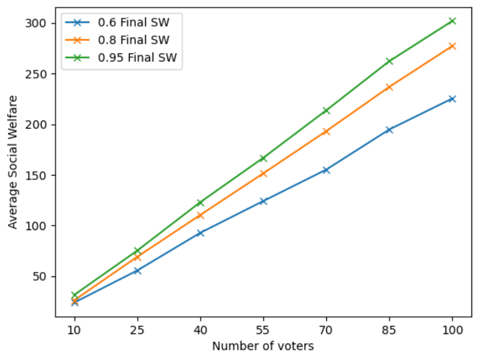


Figure 5.6: Social Welfare as N increases, Phragmén's Rule.

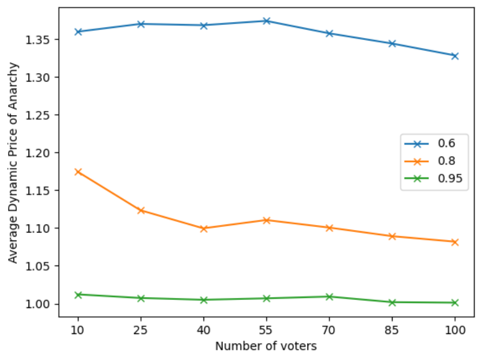


Figure 5.7: DPoA as N increases, SeqPAV.

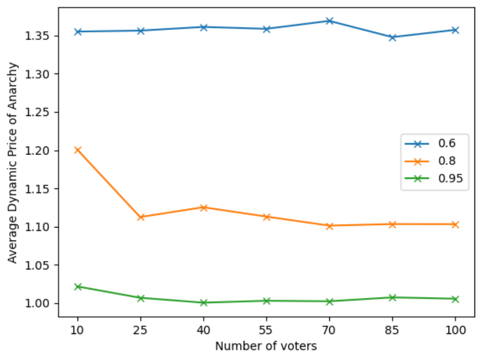


Figure 5.8: DPoA as N increases, Phragmén's Rule.

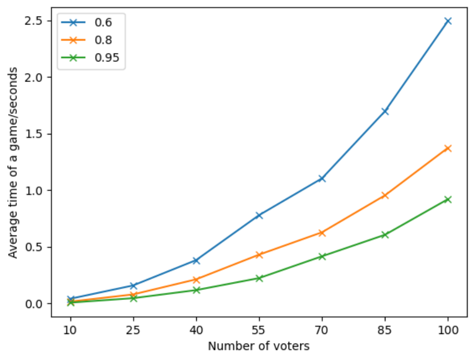


Figure 5.9: Execution Time as N increases, SeqPAV.

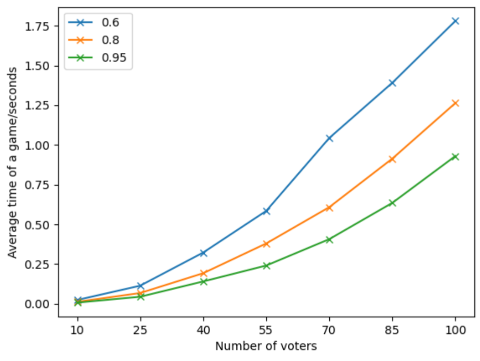


Figure 5.10: Execution Time as N increases, Phragmén's Rule.

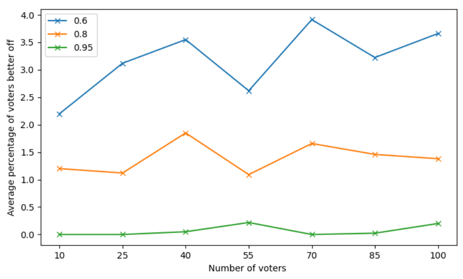


Figure 5.11: Percentage of Voters strictly better off as N increases, SeqPAV.

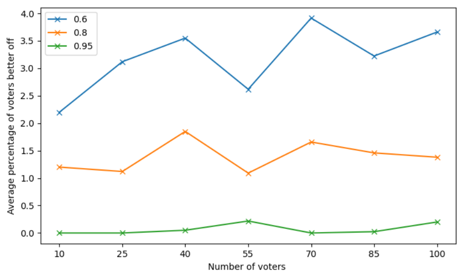


Figure 5.12: Percentage of Voters strictly better off as N increases, Phragmén's Rule.

5.1.2 Increasing the Candidate List

For the number of candidates m, we have chosen the values $m \in \{5, 6, 7, 8, 9, 10, \ldots, 20\}$. The set of voters had a size of 10 ($n = 10$) and the winner committee a size of 4 ($k = 4$). After computing all the games for each voting rule, the following results were obtained. As before, the 60% threshold is still the value that causes the most deviations. Such deviations increase as the number of alternatives goes up, as seen in Figures 5.13 and 5.14. This might happen because by introducing more candidates into the Euclidean space, even though the number of voters remains constant, there are more candidates in voters' approval set, and thus more candidates that are above the threshold. However, for higher thresholds deviations do not increase as much (or not at all). Hence, the average number of iterations also does not increase that much (Figures 5.15 and 5.16, since less deviations leads to less iterations made until the game reaches a stable state.

Increasing the candidate list has a peculiar effect on the social welfare. For high thresholds (such as 95%) it makes the social welfare increase. However, for small thresholds thresholds (such as 60%) it decreases with as the candidate list gets larger. We are using a high approval radius of 0.6, meaning that each time we add a candidate it is more likely that a voter approves it than that it does not. Thus, increasing the candidate list increases considerably the approval set of each voter and leads to higher initial total utilities. Hence, it is expected to see why the final social welfare increases with m for the 95% threshold, as seen in Figures 5.17 and 5.18. The reason for this being that the initial social welfare for such high thresholds never decreases that much, due to the few number of deviations made by voters as the game progresses. On the other hand, given that lower thresholds deviate more, it is more likely that the social welfare (for lower thresholds) decreases as the game goes on. This is because the deviations made by the Threshold voters usually have a negative impact on the total utility. Combining this with the fact that the bigger the candidate list the more deviations occur, it makes sense to see why the social welfare decreases as the candidate list gets larger, whenever a low threshold is used. This results coincide with the fact that the DPoA is increasing with m for lower thresholds, and remaining constant for high thresholds. This behavior for

DPoA and social welfare is present in both voting rules (Figures 5.19 and 5.20).

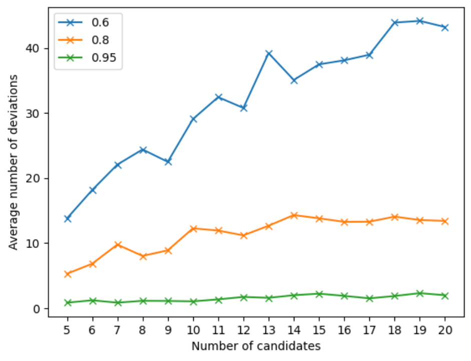


Figure 5.13: Number of Deviations as C increases, SeqPAV.

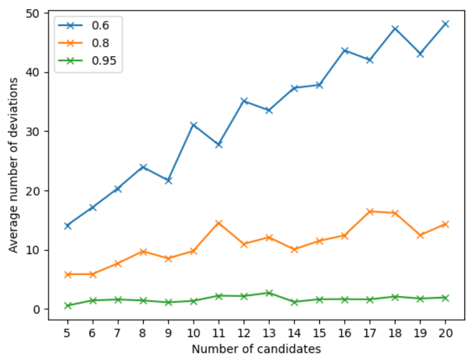


Figure 5.14: Number of Deviations as C increases, Phragmén's Rule.

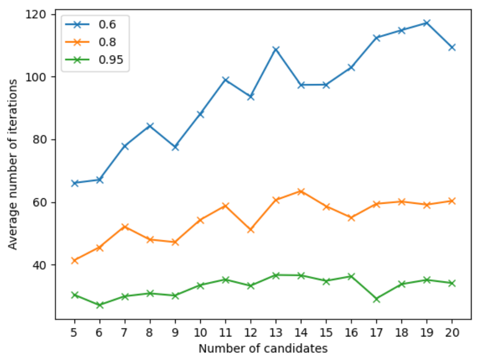


Figure 5.15: Number of Iterations as C increases, SeqPAV.

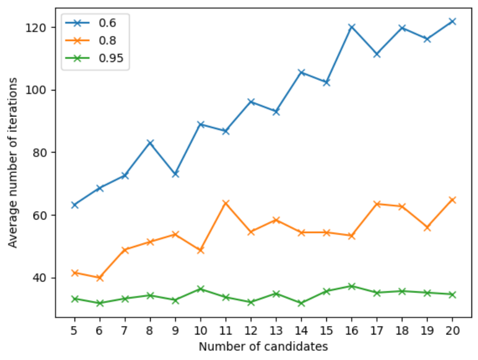


Figure 5.16: Number of Iterations as C increases, Phragmén's Rule.

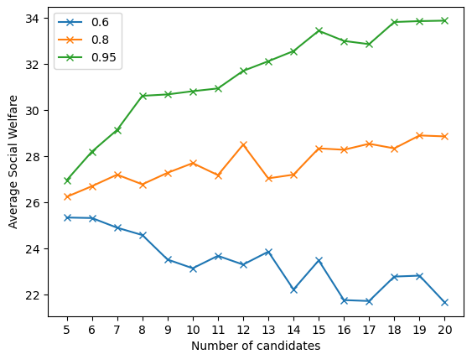


Figure 5.17: Social Welfare as C increases, SeqPAV.

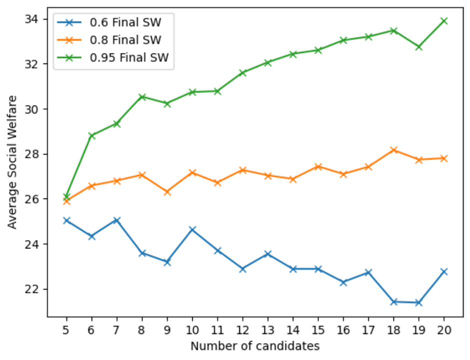


Figure 5.18: Social Welfare as C increases, Phragmén's Rule.

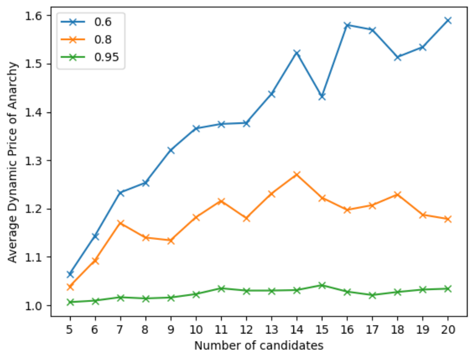


Figure 5.19: DPoA as C increases, SeqPAV.

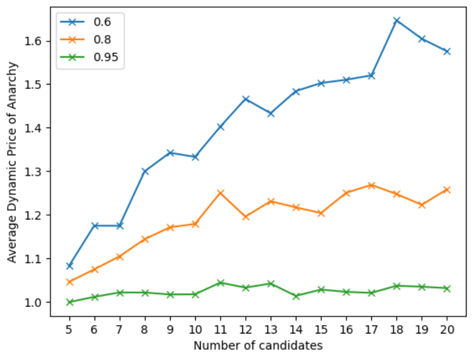


Figure 5.20: DPoA as C increases, Phragmén's Rule.

5.1.3 Increasing the Committee Size

For the number of candidates m, we have chosen the values $m \in \{1, 3, 5, 7, .., 19\}$. The set of voters had a size of 10 ($n = 10$) and the candidate list had a size of 20 ($m = 20$). After computing all the games for each voting rule, the following results were obtained.

The results seen in Figures 5.27 and 5.28 gave us a good understanding on how the DPoA behaves with this heuristic. With a committee size of 1, we had in both voting rules a high average Dynamic Price of Anarchy of 2 for the threshold of 60%, around 1.4 for 80% and 1.08 for 90%. For the first two threshold values, the DPoA is so high because if we change the winner candidate it usually has a great impact on the utilities of the voters, and from what we already know games consisting of lower thresholds tend to go through more deviations, i.e. the winner committee changes more. If a 90% is used, the DPoA does not increase as much, because from what we have previously explained, not many deviations are done. When k tends to m, we see that the deviations made by the voters do not impact the total utility as much. A justification for this might be that if a deviation made by a voter makes a candidate drop out of the winner committee, it will not have as much impact on the total utility. The reason for this being that there are still many more candidates in the winner committee that have remained as winners.

Figures 5.25 and 5.26 show us that as the committee size increases, so does the social welfare. This is because the total utility of the voters is bound to increase if we increase k, given that voters are more likely to be more satisfied with the outcome if the committee is larger.

The number of deviations made by the voters remains constant no matter the value of k chosen. However, the smaller the threshold the more average deviations a game has (Figures 5.21 and 5.22). From this we can now claim that how much a threshold voter deviates, depends on the number of voters and the size of candidate list. Changing the committee size has no effect on average number of deviations or the average number of iterations. The reason for this is that changing the committee size has no effect on the percentage approval voters of the candidates.

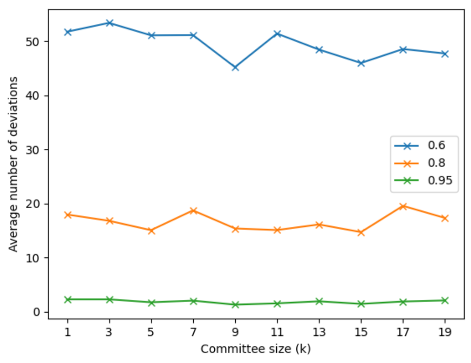


Figure 5.21: Number of Deviations as k increases, SeqPAV.

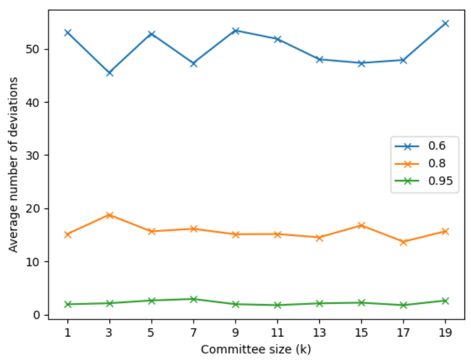


Figure 5.22: Number of Deviations as k increases, Phragmén's Rule.

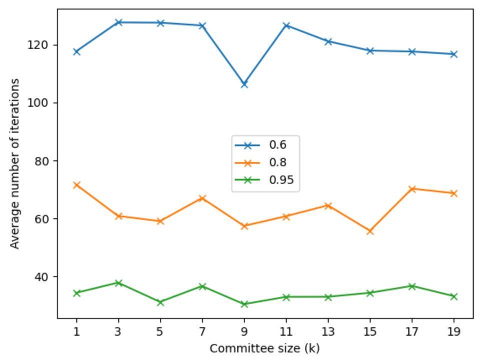


Figure 5.23: Number of Iterations as k increases, SeqPAV.

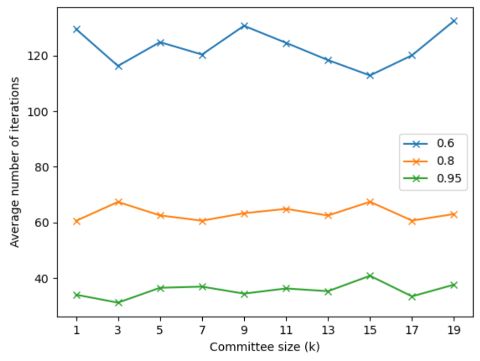


Figure 5.24: Number of Iterations as k increases, Phragmén's Rule.

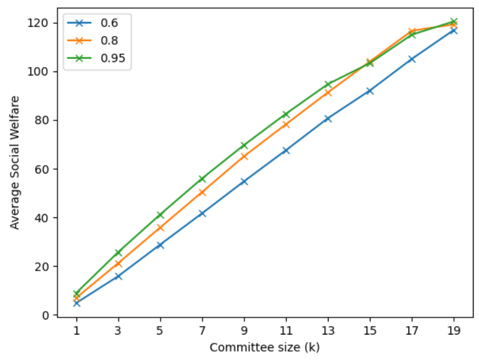


Figure 5.25: Social Welfare as k increases, SeqPAV.

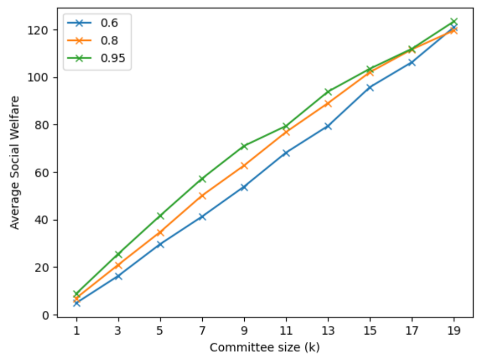


Figure 5.26: Social Welfare as k increases, Phragmén's Rule.

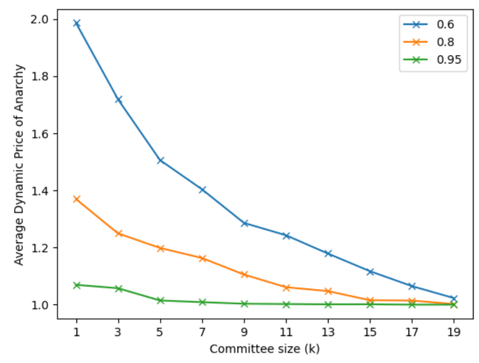


Figure 5.27: DPoA as k increases, SeqPAV.

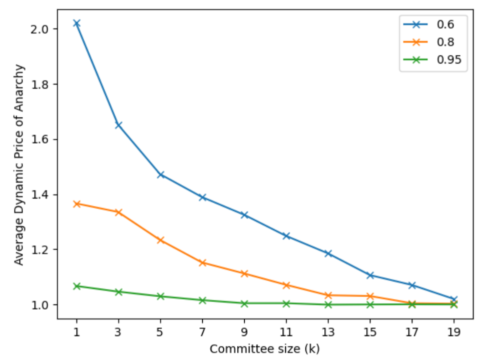


Figure 5.28: DPoA as k increases, Phragmén's Rule.

5.2 Iterative Heuristic

In this section voters will only be using the Iterative Heuristic. We will repeat each experiment for three different variants of the heuristic: Iterative Single, Iterative Up to 2 and Iterative Up to 3.

There are some games that did not reach a stable state, as we will see in the following subsections. To cope with this, a maximum number of 5000 rounds was fixed before computing all the games. For the evaluation, we have decided to include the games that did not reach a stable state, by taking as final state the one in which the game was, when the round limit was reached.

5.2.1 Increasing the Number of Voters

For the number of voters n, we have chosen the values $n \in \{5, 10, 15, 20, 25, 30, 35, 40\}$. The set of candidates had a size of 10 ($m = 10$) and the winner committee a size of 4 ($k = 4$). After computing all games for each voting rule, the following results were obtained.

We have only increased n until 40 (and not 100), given that the heuristics considered in this section have a higher complexity and thus take more time to compute the results of the games.

When looking at the average number of deviations and iterations of games as n increases, the peaks represent instances in which games did not finish with a stable state and had to reach the limit of rounds for the game to end. These games got stuck in loops of manipulations. Hence, the deviations and iterations kept increasing until the round limit was reached. Said round limit is much higher than the usual number of iterations needed to reach a stable state, therefore the peaks in the graphs form (as seen in Figures 5.31 and 5.31).

The average number of deviations (Figure 5.29) with the Iterative Up to 3 heuristic and PAV voting rule starts very high. Although this is unusual, it can be easily explained with the fact that with a small number of voters, the games were running into cycles and reaching the round limit instead of a stable state. The same thing happened with Phragmén's Voting Rule and a high number of voters (Figure 5.30), but in this case it not only happens with Iterative Up to 3 but also Iterative Up to 2. Most of the time no matter the value of n, the higher the j used by the heuristic the more deviations (and so the more iterations) are done on average per game. A justification for this is that the higher the j being used the "smarter" the voter is, so they have a larger set of possible deviations. Therefore, more opportunities appear, where it is beneficent for them to deviate. In accordance with the Threshold Heuristic, it also happens with this heuristic that the more voters we introduce the more deviations they execute, as there usually are more voters willing to manipulate. In turn those deviations open new beneficial deviation possibilities for other voters.

Increasing the number of voters caused an increase in social welfare, given that each time a voter is added to the game the total utility goes up. In both voting rules the difference in social welfare between the three variants of the heuristic does not vary (as seen in Figures 5.33 and 5.34).

For PAV the Dynamic Price of Anarchy is always above 1 (Figure 5.35), meaning that the deviations made by the voters resulted in a decrease in total utility, compared to the total utility of the initial winner committee computed with the voters' true preferences. With Phragmén DPoA usually stays above 1, except for a few games for which there was an increase in total utility (Figure 5.36). A reason for this might be that, since both voting rules being used do not have the only purpose of maximizing total utility, it is possible to find games in which the social welfare increases.

From Figures 5.37 and 5.38, we can see that no matter the value of n we choose or which voting rule we use, the average time to compute the outcome of a game usually increases with n. Additionally, using a higher value for j causes the average time to increase even more. This is because the higher the value of j we are using for the heuristic, the more possible deviations the heuristic has to check if they lead to an increase in utility. Therefore, the time a voter needs to go through all possible deviations to compute a new better response increases.

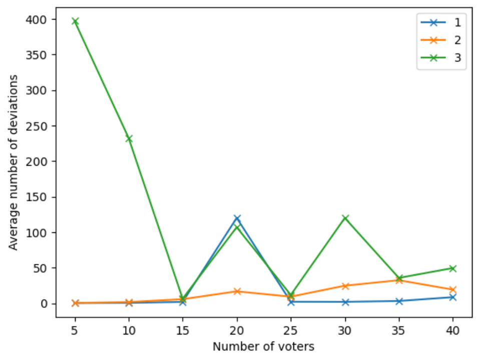


Figure 5.29: Number of Deviations as N increases, SeqPAV.

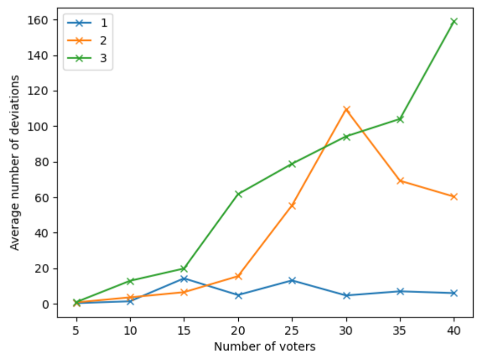


Figure 5.30: Number of Deviations as N increases, Phragmén's Rule.

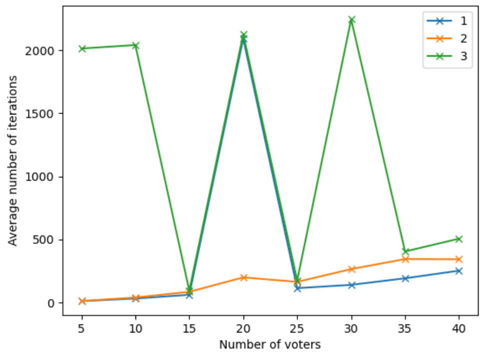


Figure 5.31: Number of Iterations as N increases, SeqPAV.

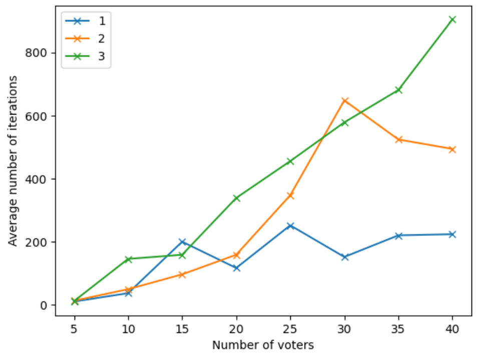


Figure 5.32: Number of Iterations as N increases, Phragmén's Rule.

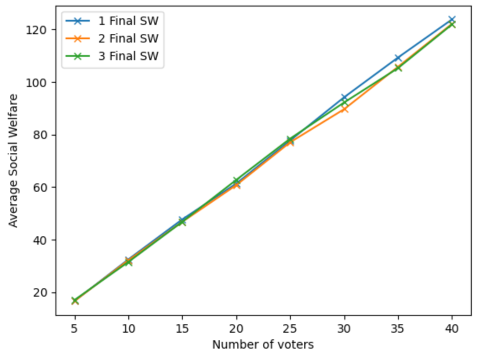


Figure 5.33: Social Welfare as N increases, SeqPAV.

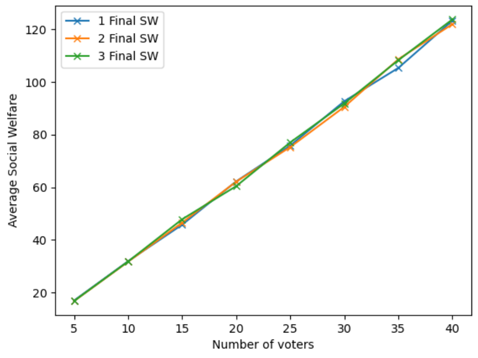


Figure 5.34: Social Welfare as N increases, Phragmén's Rule.

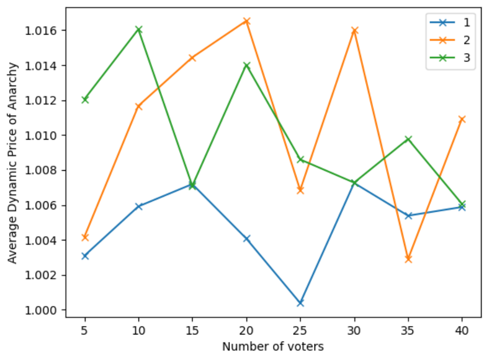


Figure 5.35: DPoA as N increases, SeqPAV.

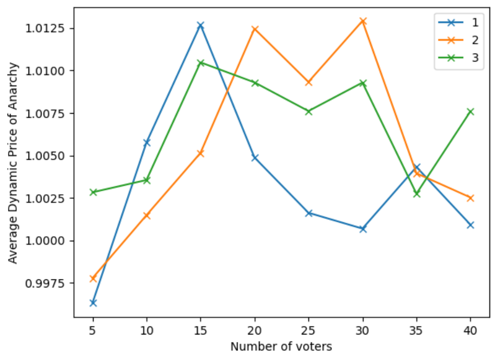


Figure 5.36: DPoA as N increases, Phragmén's Rule.

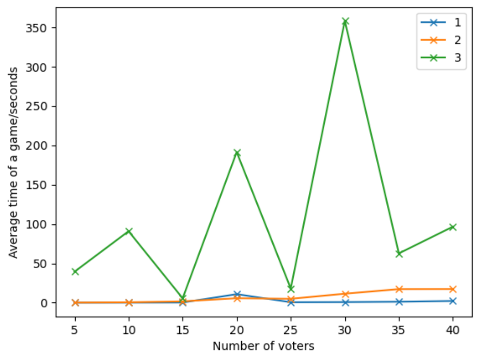


Figure 5.37: Execution Time as N increases, SeqPAV.

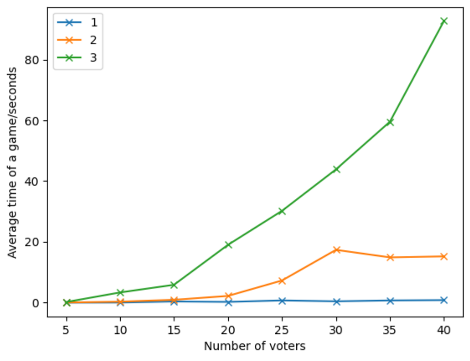


Figure 5.38: Execution Time as N increases, Phragmén's Rule.

5.2.2 Increasing the Candidate List

For the number of candidates m, we have chosen the values $m \in \{5, 6, 7, 8, 9, 10, \ldots, 20\}$. The set of voters had a size of 10 ($n = 10$) and the winner committee a size of 4 ($k = 4$). After computing all the games for each voting rule, the following results were obtained.

In this heuristic it is also the case that the social welfare of the games increases with m (Figures 5.43 and 5.44). The reason for this is the same as the one explained for the Threshold Heuristic. The average social welfare of the games using different values on j increases in the same way for all three values and both voting rules.

Figures 5.39, 5.40, 5.41 and 5.42 show us that for this experiment there are a lot more peaks in number of iterations as well as number of deviations. From this we can say that there where less games reaching a stable state (an more finishing because of the round limit), compared with the games where the number of voters increased. There is no relationship present for this heuristic between the average number of deviations (or iterations) and size of the candidate list. Even in the games with many average deviations we can see that the DPoA always stays close to 1, with the worst value being 1.02 with seqPAV as voting rule and j=3.

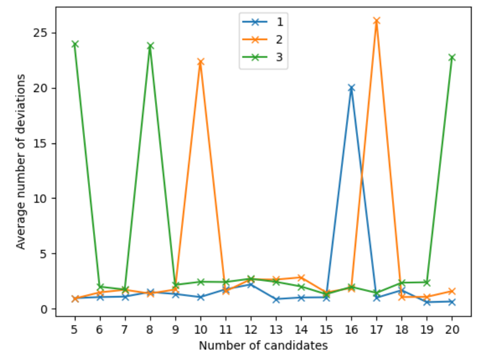


Figure 5.39: Number of Deviations as C increases, SeqPAV.

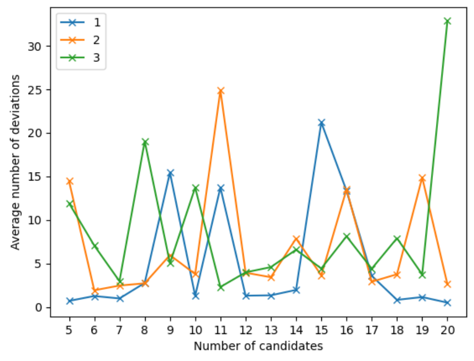


Figure 5.40: Number of Deviations as C increases, Phragmén's Rule.

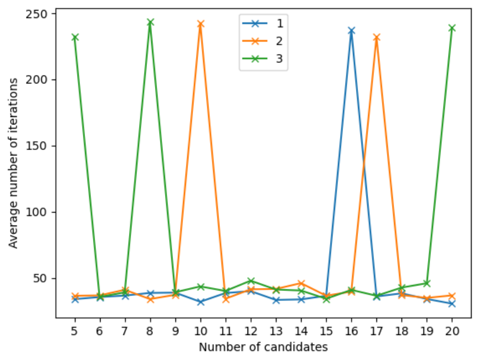


Figure 5.41: Number of Iterations as C increases, SeqPAV.

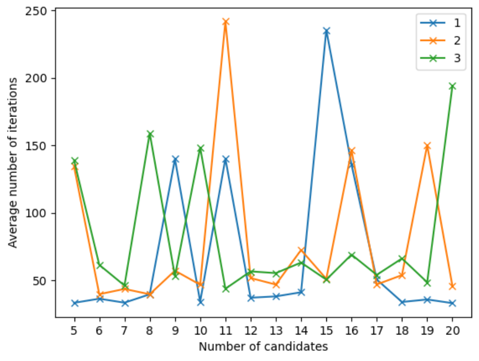


Figure 5.42: Number of Iterations as C increases, Phragmén's Rule.

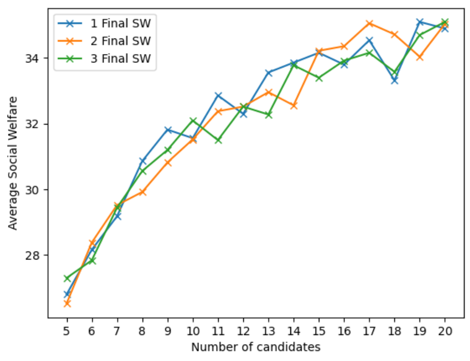


Figure 5.43: Social Welfare as C increases, SeqPAV.

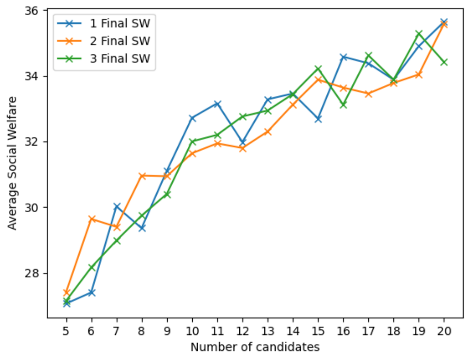


Figure 5.44: Social Welfare as C increases, Phragmén's Rule.

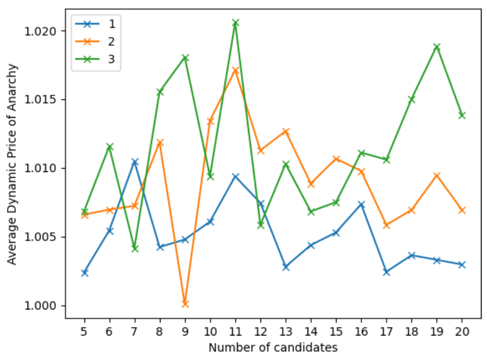


Figure 5.45: DPoA as C increases, SeqPAV.

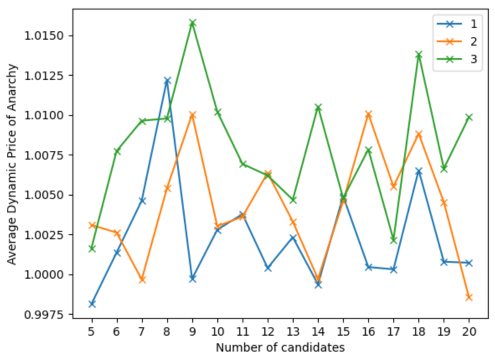


Figure 5.46: DPoA as C increases, Phragmén's Rule.

5.2.3 Increasing the Committee Size

For this experiments we have decided only to test the heuristics: Iterative Single and Iterative Up to 2. The reason for this being that when testing Iterative Up to 3 the games were running into many cycles and so the execution time was too high and unfeasible for the limited time frame of the thesis. For the committee size k, we have chosen the values $k \in \{1, 3, 5, 7, ..., 19\}$. The set of voters had a size of 10 ($n = 10$) and the candidate list a size of 20 ($m = 20$). After computing all the games for each voting rule, the following results were obtained.

As we can see from Figures 5.47 and 5.48, the deviations vary considerably as k is increased. At first with a low k the average number of deviations taken by the voters is close to 0. The reason for this is that with a low value for k, given that we are using a high radius of 0.6 for the preference generation in our Euclidean space, most of the voters will be completely satisfied and will not be willing to deviate from their preferences, hence almost no deviations were made. Additionally if $k = 1$, then both rules act like standard approval voting, which is strategy-proof [4], thus no voter will never be benefited from deviating.

Then, as k reaches a value around 13, the number of deviations increases, and peaks with $k = 13$ for both voting rules. It is at this point during the experiments in which when using Iterative Up to 3 the games were running into too many cycles and not reaching a stable state, i.e. the execution time was not feasible. We can clearly see that with $j = 2$, the number of deviations is much higher than for $j = 1$. This is because games played with $j = 2$ where running into many more cycles (not as many as with $j = 3$) and it was happening more times that the game was not reaching a stable state than with $j = 1$; which had fewer games that did not reach a stable state. With higher values for j the voters have a higher set of possible deviations, i.e. the probability that a voter takes a deviation that leads to a cycle is higher with $j = 2$ than with $j = 1$. Therefore, for the games with a k near 13 the number of deviations and iterations increased until the game reached the round limit (which is much higher than the usual total number of rounds needed to reach a stable state in a game with 10 voters and no cycles). The reason for why more agents were deviating with the values of k close to 13, was that as k increased, the number of completely satisfied voters decreased. Thus, more voters were inclined to deviate and the chances of the games entering cycles increased. Due to the games taking more iterations to finish, the average execution time of a game also increased (as seen in Figures 5.55 and 5.56).

Nevertheless, as k approached the value of m ($m = |C|$), the number of deviations dropped. This is because it makes no sense for the voters to deviate, given that it will not have that big of an impact on the output of the election. This is because they already had most (or even all) of their approved candidates in the winner committee. Therefore, not only the average number of deviations per game decreases, but also the average number of iterations and the average duration of each game. Given that there are less voters inclined to deviate, fewer games are running into cycles and so we do not have the peaks in average number of deviations (or iterations) like with $k = 13$.

From the results seen in Figures 5.53 and 5.54, it is apparent that DPoA is the highest in the games in which more deviations occur. These games correspond to the games with a committee size near 13. This is an expected result, as more deviations means that the winner committee will go through many changes as throughout a game, thus the final social welfare will be different from the initial social welfare.

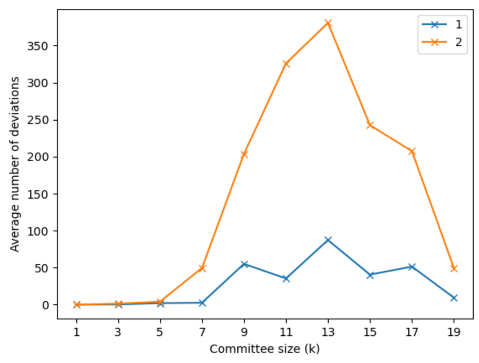


Figure 5.47: Number of Deviations as k increases, SeqPAV.

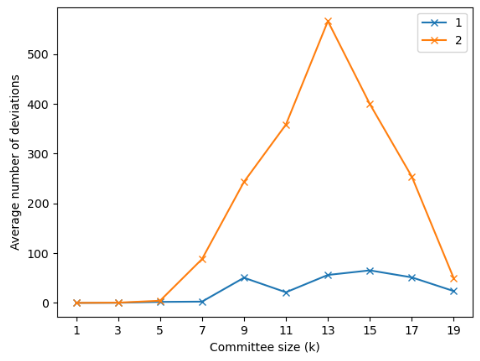


Figure 5.48: Number of Deviations as k increases, Phragmén's Rule.

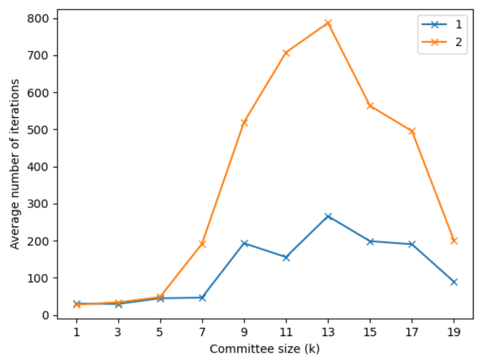


Figure 5.49: Number of Iterations as k increases, SeqPAV.

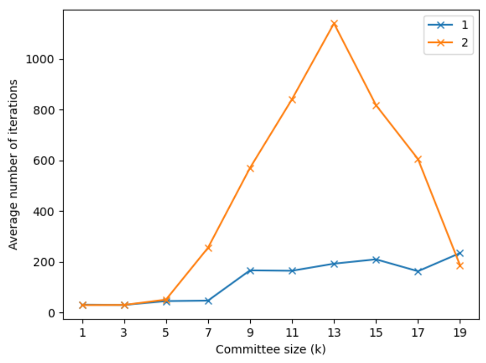


Figure 5.50: Number of Iterations as k increases, Phragmén's Rule.

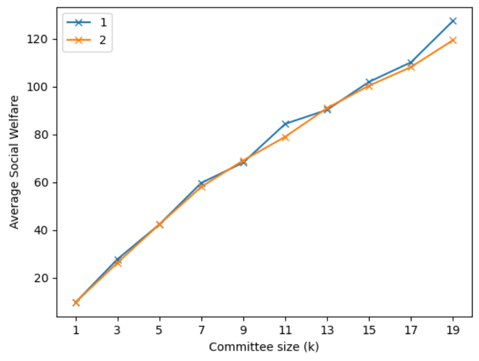


Figure 5.51: Social Welfare as k increases, SeqPAV.

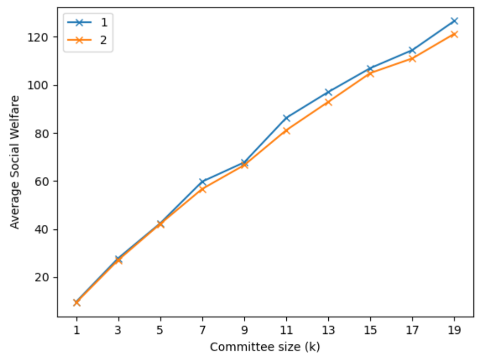


Figure 5.52: Social Welfare as k increases, Phragmén's Rule.

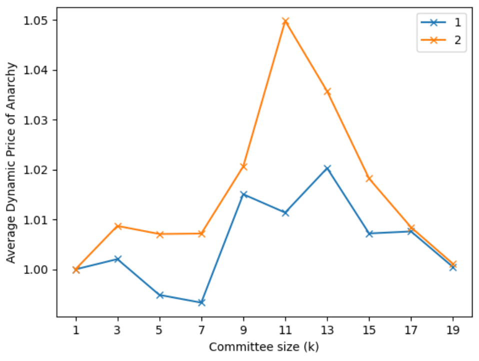


Figure 5.53: DPoA as k increases, SeqPAV.

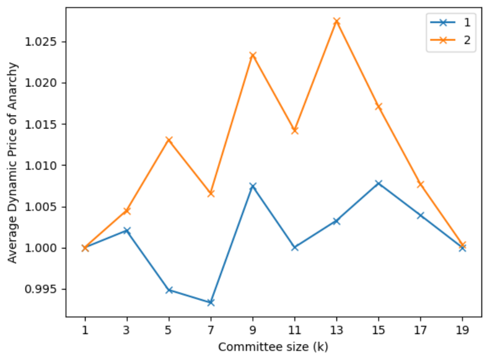


Figure 5.54: DPoA as k increases, Phragmén's Rule.

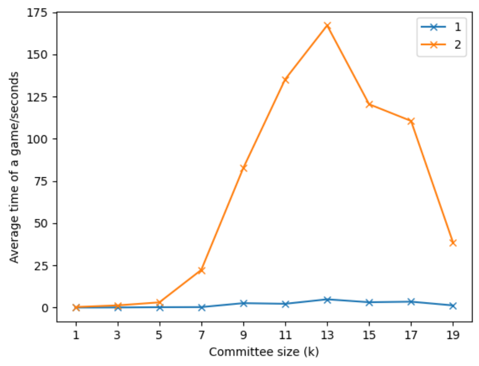


Figure 5.55: Execution Time as k increases, SeqPAV.

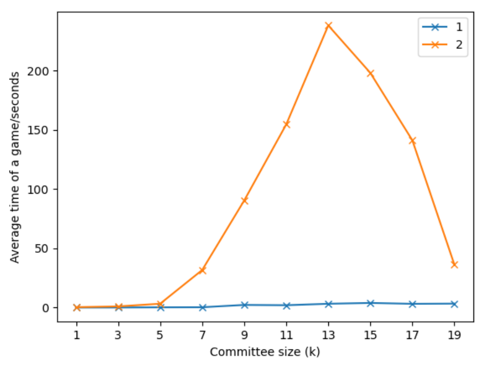


Figure 5.56: Execution Time as k increases, Phragmén's Rule.

5.3 Trial-and-Error Heuristic

For this section we have set up an environment where every voter uses the Trial-and-Error Heuristic. All the games presented in this section ended with a stable state and none of them reached the limit of rounds, which was set to 5000.

5.3.1 Increasing the Number of Voters

For the number of voters n, we have chosen the values $n \in \{10, 25, 40, 55, 70, 85, 100\}$. The set of candidates had a size of 10 ($m = 10$) and the winner committee had a size of 4 ($k = 4$). After computing all the games for each voting rule, the following results were obtained.

Increasing the number of voters caused the average number of deviations in a game to increase (as seen in Figures 5.57 and 5.58). However, in contrast to the Iterative Heuristic, the number of deviations never increase as much. This is logical because the deviation chosen by a T-and-E voter is done at random therefore is very unlikely for the deviation chosen to yield an increase in utility and for the voter to play a deviation. As a result, not many deviations are made on average per game, and no single game runs into a cycle.

Given that in most games there were not many deviations being played, the final winner committee changed little compared to the initial one. This coincides with the results seen in Figures 5.63 and 5.64, where we can see that the average Dynamic Price of Anarchy does not really increase or decrease, hence the deviations made impact little the winner committee.

The average social welfare of the games increases with the number of voters. The reason for this has been previously explained with the Threshold Heuristic. Additionally, given that this heuristic has a low complexity and not many iterations happen on average per game, the average time taken to compute the whole game increases (like it should as n increases), but never reaches a high value compared with more complex heuristics, such as Iterative Up To 3.

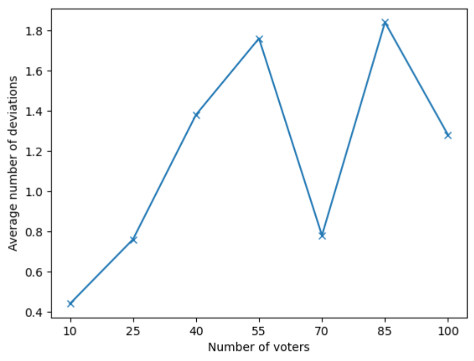


Figure 5.57: Number of Deviations as N increases, SeqPAV.

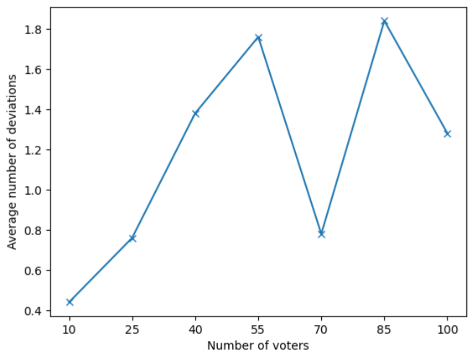


Figure 5.58: Number of Deviations as N increases, Phragmén's Rule.

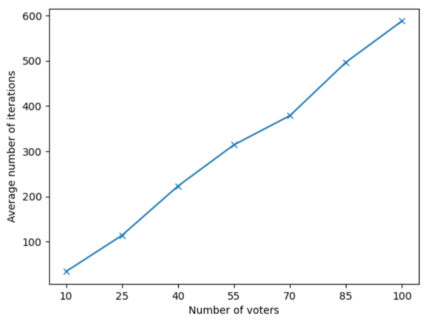


Figure 5.59: Number of Iterations as N increases, SeqPAV.

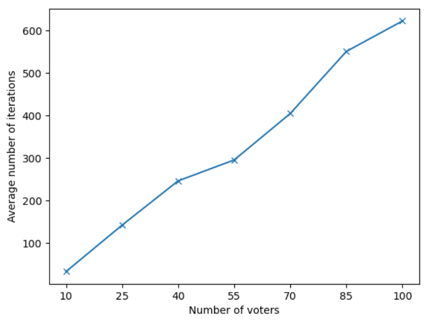


Figure 5.60: Number of Iterations as N increases, Phragmén's Rule.

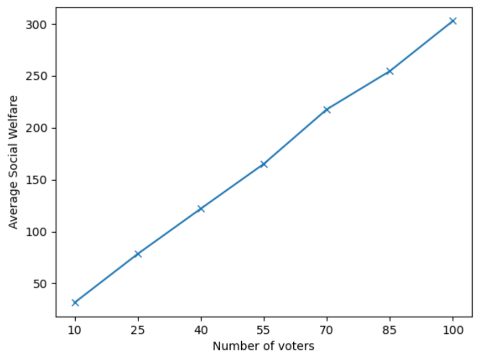


Figure 5.61: Social Welfare as N increases, SeqPAV.

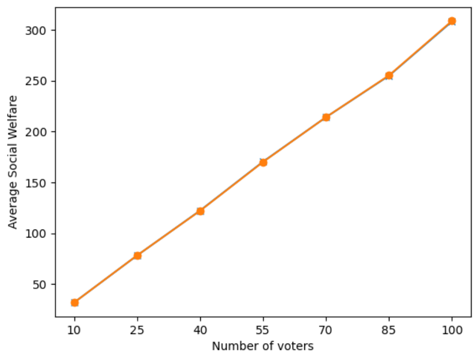


Figure 5.62: Social Welfare as N increases, Phragmén's Rule.

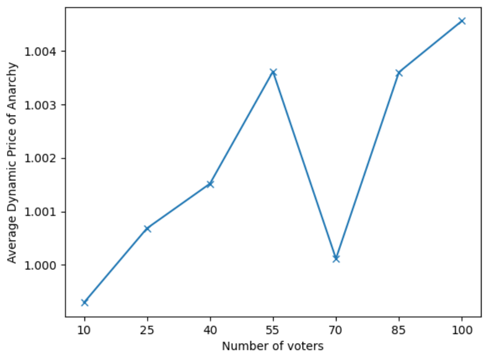


Figure 5.63: DPoA as N increases, SeqPAV.

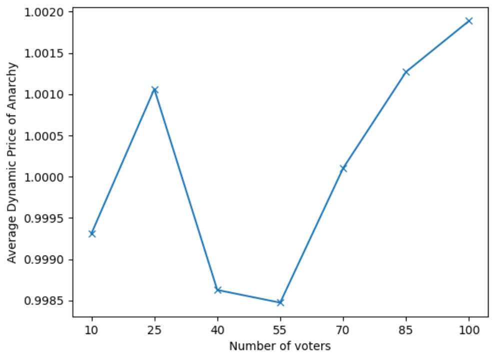


Figure 5.64: DPoA as N increases, Phragmén's Rule.

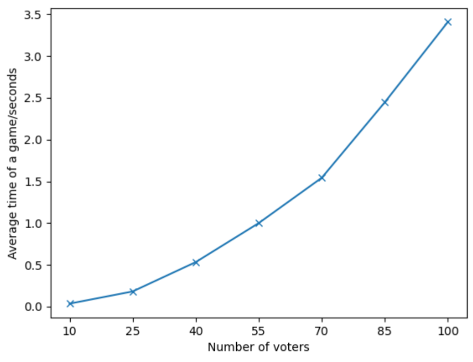


Figure 5.65: Execution Time as N increases, SeqPAV.

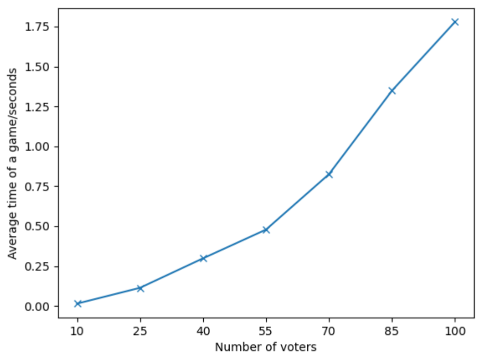


Figure 5.66: Execution Time as N increases, Phragmén's Rule.

5.3.2 Increasing the Candidate List

For the number of candidates m, we have chosen the values $m \in \{5, 6, 7, 8, 9, 10, \ldots, 20\}$. The set of voters had a size of 10 ($n = 10$) and the winner committee a size of 4 ($k = 4$). After computing all the games for each voting rule, the following results were obtained.

In both voting rules, the average number of deviations was at most 0.72 (as seen in Figures 5.67 and 5.68). Meaning that no matter the size of the candidate list, on average the deviations made by this heuristic usually is one (or even none). However, the number of deviations decreased (to a certain extent) as the candidate list got larger. This might be because the number of beneficial deviations is not increasing as much as the number of candidates. Hence the probability of a voter choosing a beneficial deviation during its turn decreases as the candidate list of alternatives gets larger. The DPoA remained close to 1 throughout all values of m (Figures 5.73 and 5.74). The reason for this (as explained before) is that in games with few deviations there is not much difference between the initial and final total utilities, since the winner committee goes through few changes. Therefore the final social welfare is usually very close to the initial social welfare.

With both voting rules, it happens that the social welfare increases as the candidate list gets larger (Figure 5.71 and 5.72). This is a recurring behaviour, as adding more voters increases the total utility of the game, no matter which heuristic or voting rule is being used.

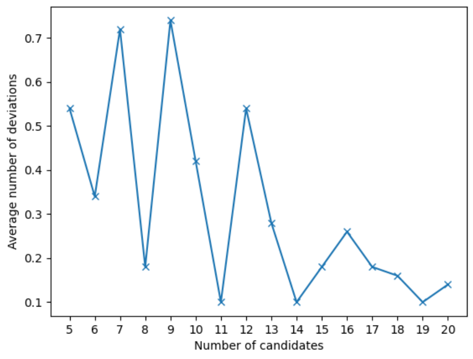


Figure 5.67: Number of Deviations as C increases, SeqPAV.

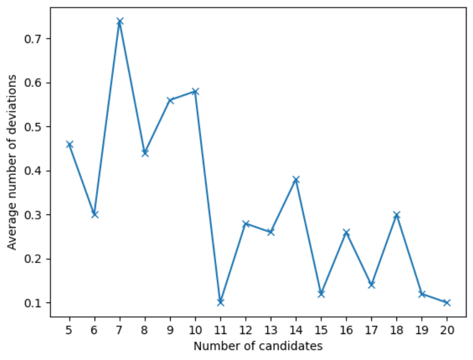


Figure 5.68: Number of Deviations as C increases, Phragmén's Rule.

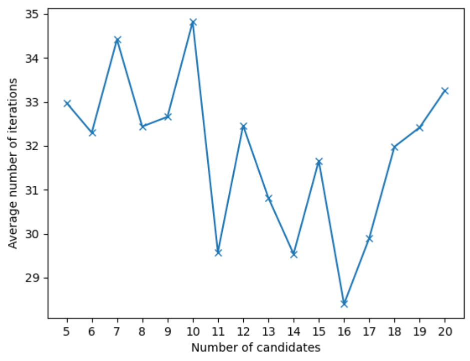


Figure 5.69: Number of Iterations as C increases, SeqPAV.

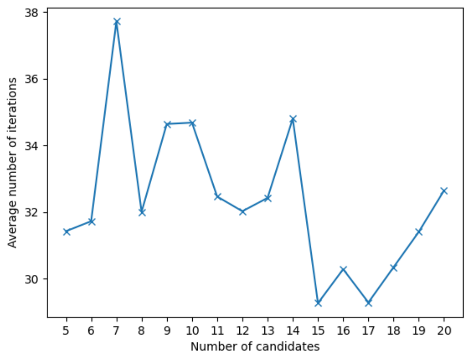


Figure 5.70: Number of Iterations as C increases, Phragmén's Rule.

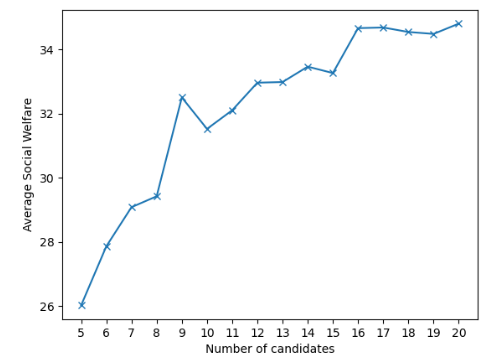


Figure 5.71: Social Welfare as C increases, SeqPAV.

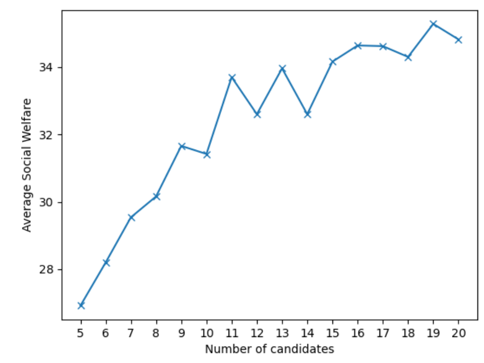


Figure 5.72: Social Welfare as C increases, Phragmén's Rule.

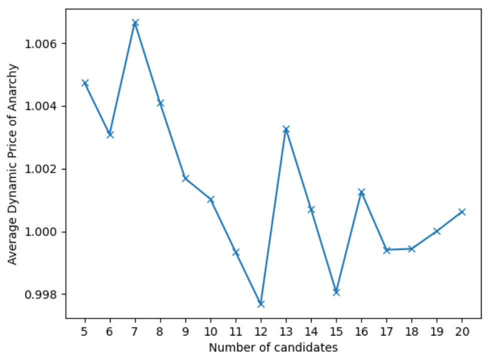


Figure 5.73: DPoA as C increases, SeqPAV.

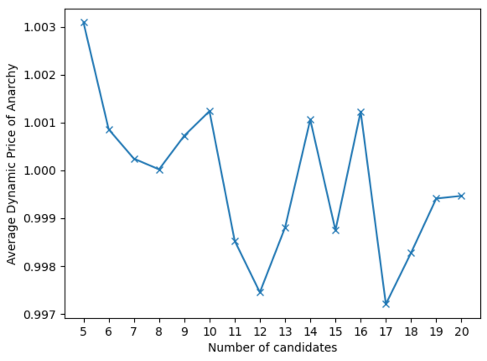


Figure 5.74: DPoA as C increases, Phragmén's Rule.

5.3.3 Increasing the Committee Size

For the committee size k, we have chosen the values $k \in \{1, 3, 5, 7, ..., 19\}$. The set of voters had a size of 10 ($n = 10$) and the candidate list a size of 20 ($m = 20$). After computing all the games for each voting rule, the following results were obtained.

The number of deviations varied as the winner committee got larger, but peaked for both voting rules with a size around 17 candidates. Then it decreased as k got closer to m. It is reasonable to see these results and the explanation is the same as the one presented in the Subsection 5.2.3. In contrast to the games simulated in that section, with Trial-and-Error no single game finished without reaching a stable state no matter the value of k that we tried. Thus the deviations do not increase as much as with the Iterative Heuristic. It makes sense that the T-and-E Heuristic is running into less cycles, as the probability of a voter taking a deviation is much lower that if the Iterative Heuristic is used. The reason being that a T-and-E voter only checks if utility increases for one random deviation per round.

If k is increased also does the social welfare (Figures 5.79 and 5.80), which has an obvious explanation: increasing k increases the total utility of the voters, as there are more candidates in the winner committee.

The DPoA is distanced from 1 the most in the games in which more deviations are taken by voters (Figures 5.81 and 5.82). The reason for this can also be found in 5.2.3.

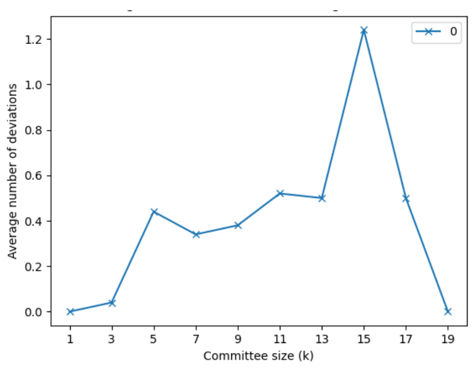


Figure 5.75: Number of Deviations as k increases, SeqPAV.

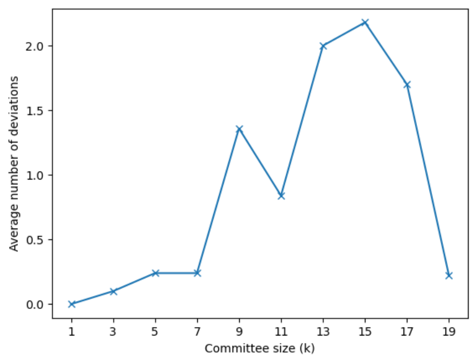


Figure 5.76: Number of Deviations as k increases, Phragmén's Rule.

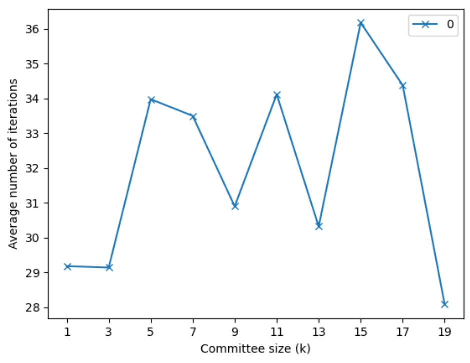


Figure 5.77: Number of Iterations as k increases, SeqPAV.

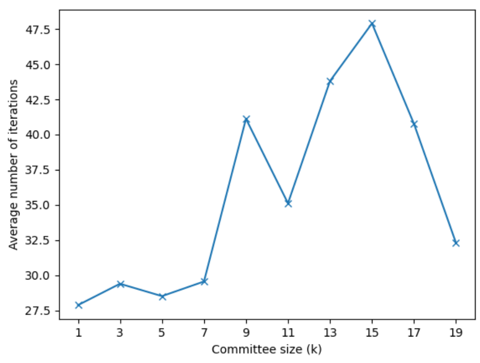


Figure 5.78: Number of Iterations as k increases, Phragmén's Rule.

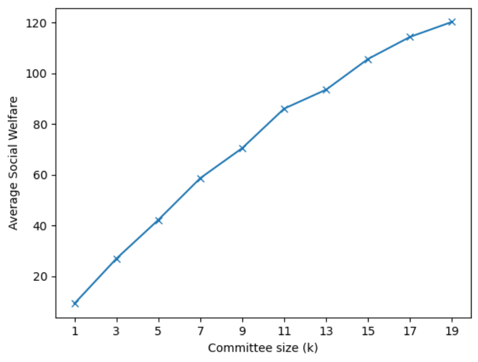


Figure 5.79: Social Welfare as k increases, SeqPAV.

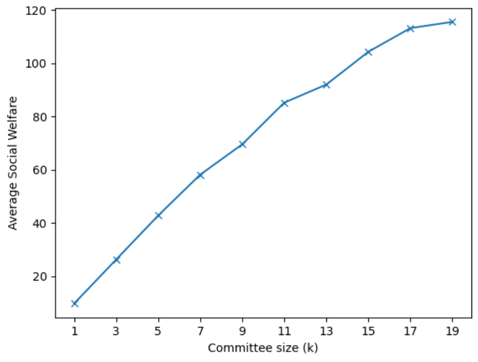


Figure 5.80: Social Welfare as k increases, Phragmén's Rule.

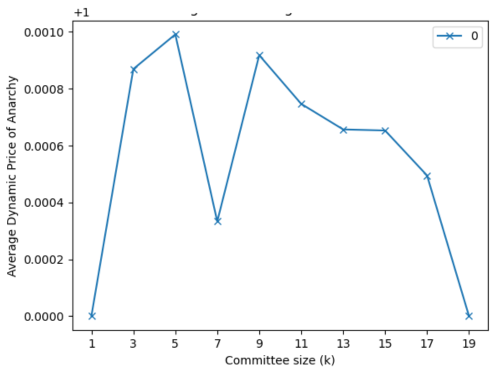


Figure 5.81: DPoA as k increases, SeqPAV.

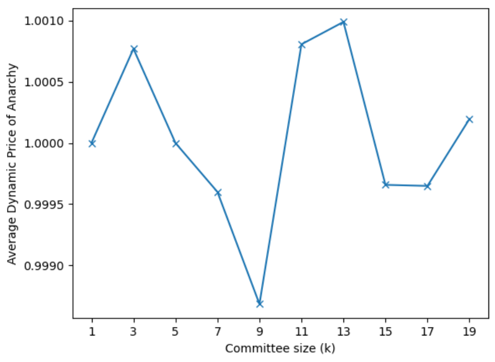


Figure 5.82: DPoA as k increases, Phragmén's Rule.

5.4 Factional Heuristic

We have set up games, where we have a percentage of voters that belong to a faction (and have the same preferences and use the Factional Heuristic). The rest of the voters will use the Truthful Heuristic. While we change the different experimental values, the percentage of voters belonging to the faction will remain constant. Each experiment will be done for a faction of sizes 30%, 60% and 5% (just in the increasing number of voter's experiment) of the total voters. For these experiments we will not be using the reactive variant, as there will be no voter outside of the faction deviating from their true preferences. The heuristic used by the faction to compute a better response will be Iterative Single.

For the following experiments in this section we will not be using the concept of Success Probability introduced in 2.5.1. We have chosen to do this as we think that by using such concept, we will be restricting too much the deviations made by the heuristic. We have also done this, given that the heuristic already performs few iterations and adding even more constraints would negatively affect the experiments' results. Additionally, by not using it, we will be able to have a better understanding on how the Factional Heuristic approaches strategic voting.

Even though we are using a heuristic that allows both types of manipulations, none of the games presented in this section finished without reaching a stable state. This is because all the faction voters using Iterative Single where working together performing the same deviations, and the rest of the voters were not deviating at all. Therefore it is understandable that no game got stuck in a cycle of manipulations.

We also want to mention that, because no outsider is deviating, the faction can only increase its utility. Thus, in these games we want to asses how bad the result can be when the factions deviates.

5.4.1 Increasing the Number of Voters

For the number of voters n, we have chosen the values $n \in \{10, 25, 40, 55, 70, 85, 100\}$. The set of candidates had a size of 10 ($m = 10$) and the winner committee had a size of 4 ($k = 4$). After computing all the games for each voting rule, the following results were obtained.

As seen in the Iterative Heuristic experiment, with the Single variant, the number of deviations made by the voters were not very high. Therefore, it is logical to see that none of the three percentages have high amounts of deviations either (Figures 5.83 and 5.84). Nevertheless, it is interesting to point out that when the faction has a size of 60%, almost no deviations are done. This is because, given that faction's size is very big, it is safe to assume that the faction will be very satisfied with the initial winner committee, since both voting rules aim to output a winner committee that "satisfies" to a certain extent most of the voters. Thus, the faction is not inclined to deviate.

On the other hand, the results show us that for a smaller faction size like 30% there are more games in which the voters deviate. The reason for this being that it is more probable that as smaller faction usually is not completely satisfied with the initial winner committee. Increasing the number of voters makes the deviations made by the 30% faction increase as well, since each time one member of the faction deviates, every voter in the faction does so too.

Something peculiar that is occurring in these experiments, is the fact that for the faction sizes of 60% and 30%, whenever the faction deviates, the DPoA goes below 1, as seen in Figures 5.89 and 5.90. This tells us that the final social welfare of the election is greater than the initial one, since with the new committee found the satisfaction of many voters (that belong to the faction) is increased. Though this must come at a price: even though the utility of the faction is increasing, the utility for some of the truthful voters might be decreasing. The fact that the social welfare has increased does not always mean that we have found a "better" winner committee. This is the reason why many voting rules do not have the sole objective of maximizing social welfare.

Decreasing the faction size to 5% causes the number of deviations to increase. This is because, given that the faction is now much smaller, it is more likely that the faction is not as satisfied with the winner committee as if the size was larger. Thus, smaller factions are more likely to deviate that larger ones. With a size of 5%, it still the case that increasing the number of voters also increases the average number of deviations per game, for the same reason as explained before. In contrast to the other two faction sizes, most of the games have a DPoA above 1, meaning that most games experience a decrease in total utility when the faction deviates. This is because, now that the faction is much smaller, the new winner committee after the faction deviates probably improves the utility for a few number of voters in the game.

For all sizes of faction and both voting rules the social welfare of the game increases as we increase the number of voters (Figures 5.87 and 5.88). This makes sense as increasing the number of voters will increase the total utility of the game. Given that there are not many deviations happening and that the DPoA never really departs from 1 no matter what size we use, it is reasonable to see that the difference in social welfare between the three sizes is not very high.

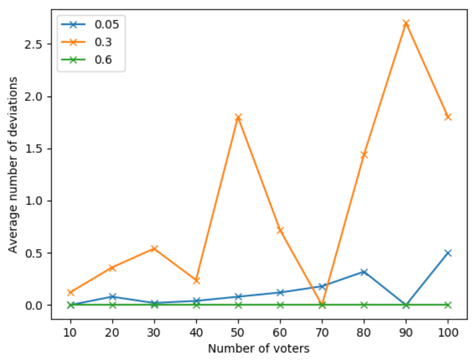


Figure 5.83: Number of Deviations as N increases, SeqPAV.

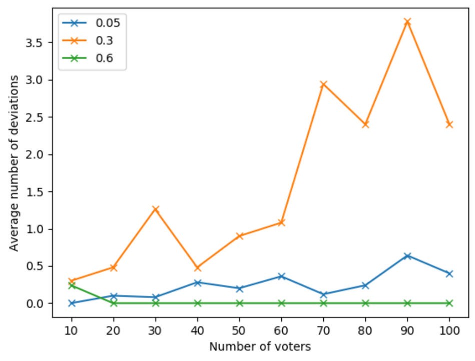


Figure 5.84: Number of Deviations as N increases, Phragmén's Rule.

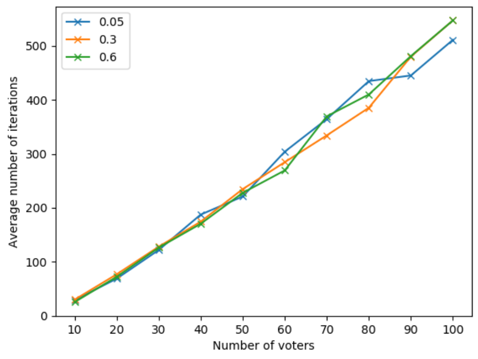


Figure 5.85: Number of Iterations as N increases, SeqPAV.

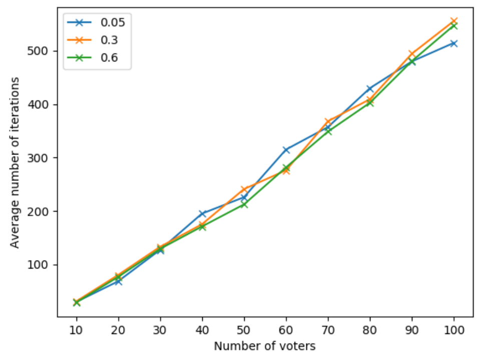


Figure 5.86: Number of Iterations as N increases, Phragmén's Rule.

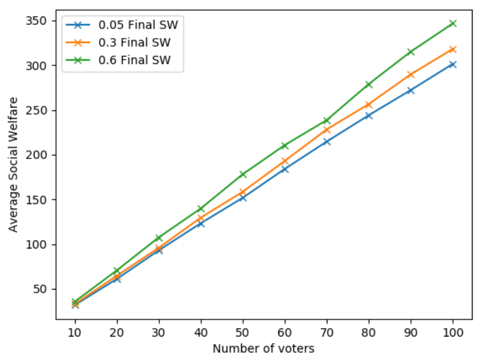


Figure 5.87: Social Welfare as N increases, SeqPAV.

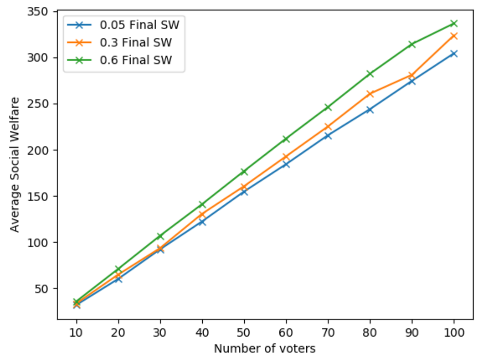


Figure 5.88: Social Welfare as N increases, Phragmén's Rule.

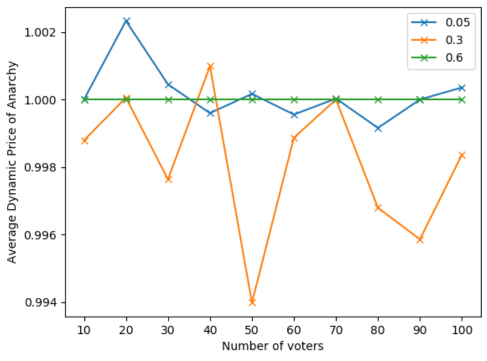


Figure 5.89: DPoA as N increases, SeqPAV.

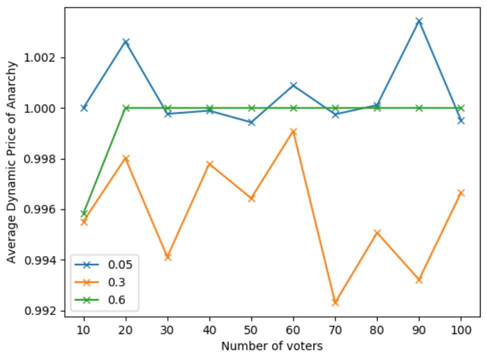


Figure 5.90: DPoA as N increases, Phragmén's Rule.

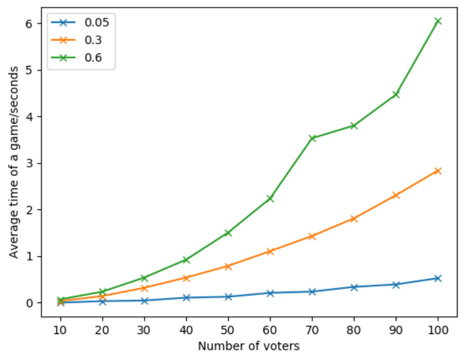


Figure 5.91: Execution Time as N increases, SeqPAV.

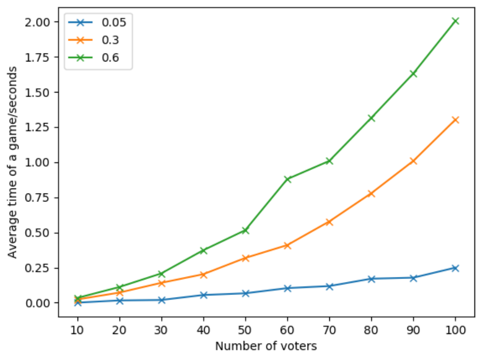


Figure 5.92: Execution Time as N increases, Phragmén's Rule.

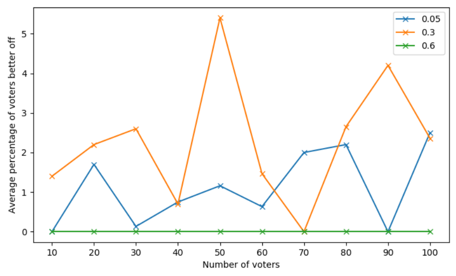


Figure 5.93: Percentage of Voters strictly better off as N increases, SeqPAV.

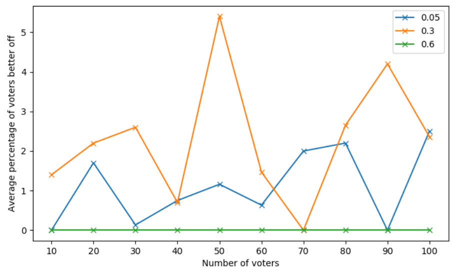


Figure 5.94: Percentage of Voters strictly better off as N increases, Phragmén's Rule.

5.4.2 Increasing the Candidate List

For the number of candidates m, we have chosen the values $m \in \{5, 6, 7, 8, 9, 10, \ldots, 20\}$. The set of voters had a size of 10 ($n = 10$) and the winner committee a size of 4 ($k = 4$). We only used two sizes of factions (30% and 60%) as it does not makes sense to look at a faction of size 5% of 10 voters. After computing all the games for each voting rule, the following results were obtained.

In accordance with the previous experiment, no matter the value for m chosen, the number of deviations that occur on average in a game are low for both sizes and voting rules, especially on the games with a faction size of 60% (Figures 5.95 and 5.96). It also happens that the smaller the faction size, the more deviations they execute. However, in this case, even though the candidate list increases the average number of deviations (as well as the average number of iterations) stays almost constant. If the candidate list increases, also does the social welfare (Figures 5.99 and 5.100), for the same reason as explained in 5.1.2.

Following the results of the previous experiment, here it also happens that the DPoA usually goes below 1 whenever the faction deviates. The reason for this is the same as the one explained in the previous experiment for factional voters. It is interesting to mention that the social welfare for games with a faction of 60% is always higher than the social welfare for the games with a faction of 30%. The reason for this is that in games where a high number of voters have the same exact preferences it is much easier to find a winner committee with a high total utility, than in games where not as many voters have the same preferences.

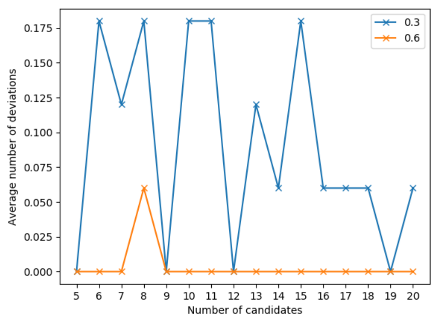


Figure 5.95: Number of Deviations as C increases, Seq-PAV.

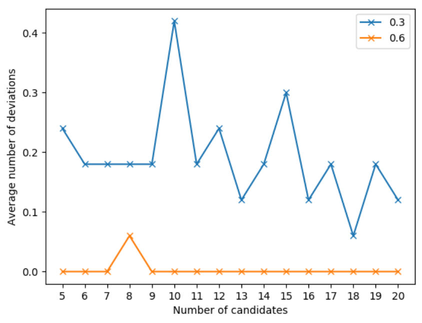


Figure 5.96: Number of Deviations as C increases, Phragmén's Rule.

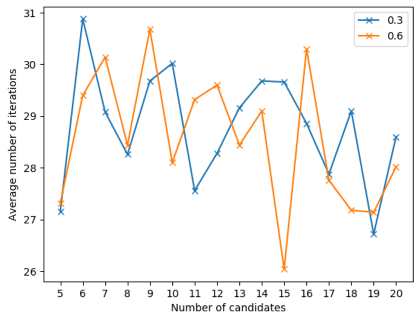


Figure 5.97: Number of Iterations as C increases, SeqPAV.

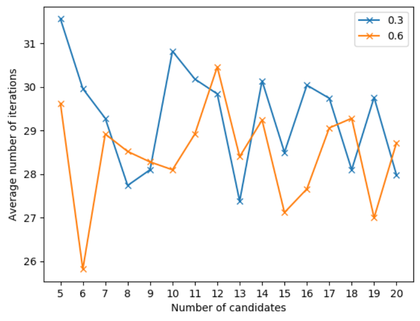


Figure 5.98: Number of Iterations as C increases, Phragmén's Rule.

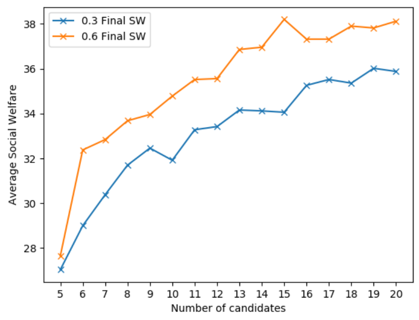


Figure 5.99: Social Welfare as C increases, SeqPAV.

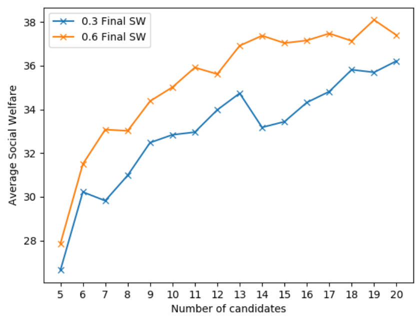


Figure 5.100: Social Welfare as C increases, Phragmén's Rule.

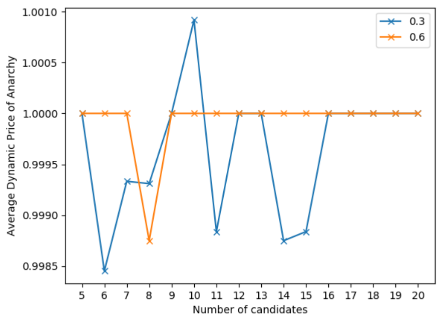


Figure 5.101: DPoA as C increases, SeqPAV.

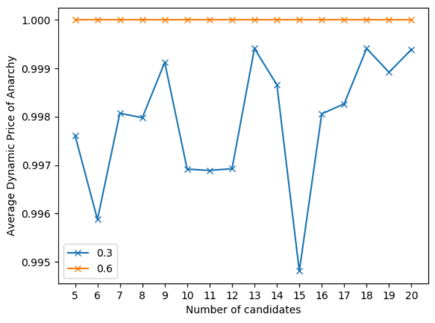


Figure 5.102: DPoA as C increases, Phragmén's Rule.

5.4.3 Increasing the Committee Size

For the committee size k, we have chosen the values $k \in \{1, 3, 5, 7, ..., 19\}$. The set of voters had a size of 10 ($n = 10$) and the candidate list a size of 20 ($m = 20$). For this experiment we will not use the faction size of 5%, for the same reason as in the previous experiment. After computing all the games for each voting rule, the following results were obtained.

As mentioned in both previous experiments for the Factional Heuristic, it is understandable to see how the number of deviations throughout all values for k are never very high (as seen in Figures 5.103 and 5.104). Nonetheless, here the results are different. At first with a small committee size and a faction size of 30%, the number of deviations is almost 0. Then it keeps increasing until the committee size reaches to a value around 13 (in both voting rules). After this, it drops back to 0 deviations, as k approaches m. The explanation for this behavior is the same as the one given for the experiments with Iterative voters, where k in increased. Briefly, with a small committee size, and given that we are using a radius of 0.6 for the generation of approval preferences, it is more likely that the faction is completely satisfied, than that it is not. Thus, the faction is not encouraged to deviate. Moreover, if $k = 1$, both voting rules behave exactly like standard approval voting, which is strategy-proof (such statement can be found in [4]), hence no deviations are being performed. However, as k increases, it is more likely that the faction is not completely satisfied and that it benefits from deviating. Thus, the faction is usually more willing to deviate as k increases. Finally, with a k close to m they are also not very inclined to deviate, as most (or even all) of their approved candidates will already be included in the initial winner committee. Hence, any deviations will have little effect (or even none) on the output of the election.

The DPoA acts in the same way as in the two previous experiments with the Factional Heuristic (seen in Figures 5.109 and 5.110) and the same explanation can be used justify why it drops below 1 whenever the faction deviates as the one given in 5.4.1, no matter what size of committee we use.

It is logical that as the committee size increases so does the social welfare. This is because increasing k leads to a higher utility (in most cases) for all voters, so social welfare is bound to increase as well.

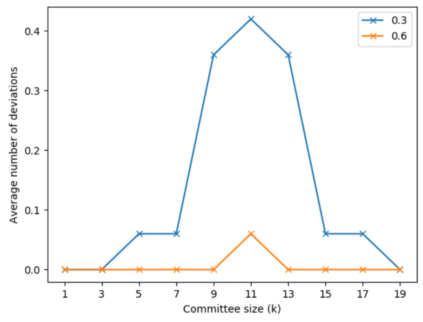


Figure 5.103: Number of Deviations as k increases, SeqPAV.

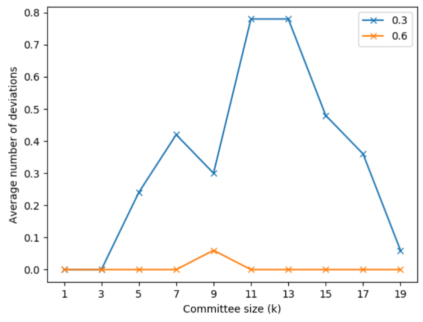


Figure 5.104: Number of Deviations as k increases, Phragmén's Rule.

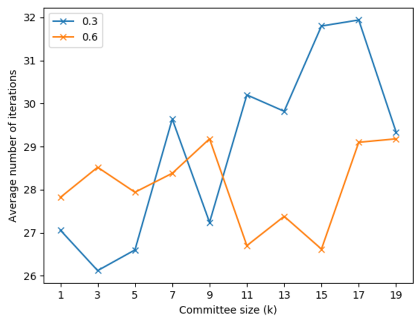


Figure 5.105: Number of Iterations as k increases, SeqPAV.

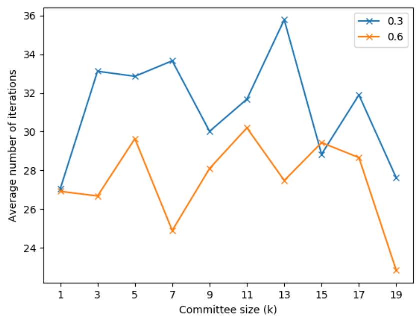


Figure 5.106: Number of Iterations as k increases, Phragmén's Rule.

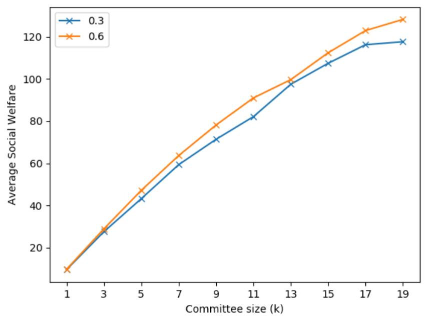


Figure 5.107: Social Welfare as k increases, SeqPAV.

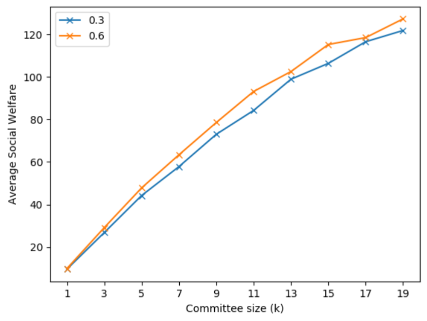


Figure 5.108: Social Welfare as k increases, Phragmén's Rule.

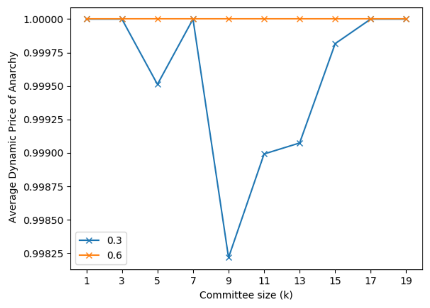


Figure 5.109: DPoA as k increases, SeqPAV.

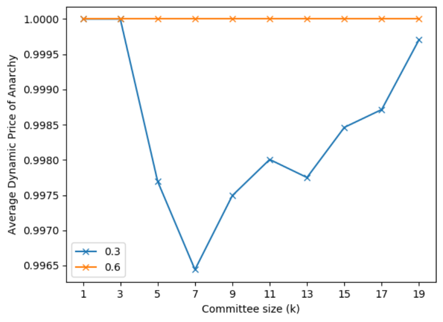


Figure 5.110: DPoA as k increases, Phragmén's Rule.

5.5 Conclusion of the Experiments

Now that we have gone over the experiments and their results, we would like to discuss our conclusions. First we want to point out, that for most of the experiments made, the two different voting rules that we have contemplated in these experiments have similar behavior, when being used in games with voters using the heuristics considered in this thesis, even though they do not always compute the same winner committee for games. However, they deviate the most in games where all voters are using the Iterative Heuristic. One of the reasons for this is that a game using one of the voting rules may end up stuck in a cycle, and this might not be the case if the same preference profile is used, but with the other voting rule. Hence results related to number of deviations, number of iterations or temporal cost of the game diverge. Consequently, experiments in which games where running into many cycles, presented different results and patterns depending on the voting rule being used.

Furthermore, we have seen that usually in games with more deviations the DPoA diverges more from 1. However, more deviations do not always imply that a “smart” heuristic is used. This is shown with the fact that one of the heuristics which tends to deviate the most from the initial true preferences, is the Threshold Heuristic with a threshold of 60%. Such heuristic manipulates more that Trial-and-Error, Factional and in some cases even Iterative Single Heuristic. Additionally, it may even manipulate when the voting rule being used is strategy-proof. This can be seen in 5.1.2, where Threshold voters deviate when the committee has a size of 1 ($k = 1$) even though both voting rules are strategy-proof, as explained in [4]. This behaviour is not present with other heuristics that do take into account which voting rule is being used before computing a manipulation. Yet, we know that the Threshold Heuristic with a low threshold value does not make as educated decisions as the others heuristics, so it would not be used in a realistic scenario.

Another aspect we would like to point out, is the fact that games where many deviations are made tend to decrease the total utility of voters. A reason for this is the fact that the deviations made by individual voters usually mean an increase in their own utility, but mean a decrease in utility for some of the rest of voters; who see one (or more) of their approved candidates drop from the winner committee. As a consequence, games with many deviations tend to have a lower final social welfare (unless a large faction is the one deviating), compared to the initial social welfare. Which in turn means that the game has a DPoA above 1. This is not a good result, as it tell us that our voting system is degrading (in terms of social welfare) due to the selfish behaviour of our voters.

Another conclusion learnt from these experiments, is that not finishing a game with a stable state is more common that initially expected, when using a heuristics that considers a large number of possible deviations and is able to perform both subset and superset manipulations. We have reached this conclusion as games with the Iterative Single Heuristic; a heuristic that considers m deviations per round, finished a few times without reaching a stable state. Alternatively, Iterative Up to 3, a heuristic that considers many more possible deviations per turn, did run into more cycles and had more games where stable states were not reached. When increasing the committee size for the Iterative Up to 3 Heuristic experiment 5.2.3, not reaching a stable state was particularly problematic, as it increased the execution time too much. The reason being that many games where running into cycles and not reaching stable states, which caused a steep increase in the time needed to compute each game. This was not as problematic with the Iterative Single or Iterative Up to 2 heuristics, as seen in 5.2.3, were we can observe that not many games finished by reaching the round limit (and not reaching a stable state).

Moreover, another conclusion derived from the results, is that strategic voting (using the heuristics presented in this thesis) in an iterative setting with an approval-based voting rules might not be as harmful as previously thought. As seen from the results, in most of the experiments games usually have values of DPoA close to 1. The heuristic that leads to games with the highest values of DPoA is the 60% Threshold Heuristic, and even in that case we should not worry as we know that using such heuristic in a normal game (where other voters may use other heuristics) will most likely end up backfiring to the voter using it, i.e. decreasing its utility.

A big difference between the heuristics is the time of execution. The Threshold Heuristic given its low complexity took on average less than 3 seconds to compute a whole game, even with 100 voters and the 60% threshold (which lead to more iterations/deviations than the other threshold values). Heuristics like Trial-and-Error or Iterative Single, despite their higher complexity, took on average below 10 seconds and below a minute respectively to compute the outcome of a game with 100 voters. However, Iterative Up to 3 had a much higher complexity than the previously mentioned heuristics. This combined with the fact that there were many games running into cycles, caused the average running time to increase considerably. This is a very important result because it helps us understand that, even though Iterative Up to J (with a J close to m) is a very powerful heuristic to use, many voters might not be willing to use it in a realistic scenario due to its high computational cost. Therefore, to a certain extent Iterative Up to J being NP-hard, gives the voting rules a sort of protection against strategic voting using this heuristic. This argument presented was also was also studied by V. Conitzer and T. Walsh in [5].

On the other hand, using a highly complex heuristic (such as Iterative Up to J) with a faction could be very powerful. This is because even though computing a better response for the faction would be very computationally complex, it will be compensated by the fact that said new response only needs to be computed once by one member of the faction, given that the rest of the voters will simply copy that deviation.

To conclude this section we want to review a common behavior that has been present in the heuristics: Trial-and-Error, Iterative and Factional. This behaviour refers to the number of deviations done on average per game as we increase the committee size. Looking back at the results for the three heuristics (from Figures 5.47, 5.48, 5.75, 5.76, 5.103 and 5.104) we see that they have a similar pattern. In all three heuristics and both voting rules the number of deviations per game starts low (even 0), then increases until a value for k in the range $[11, 15]$ is reached and then it drops back down to 0. The reason for this pattern is presented in Section 5.2.3. To explain it briefly, with $k = 1$, no voters deviate as both voting rules become strategy-proof. Furthermore, with small k values, given that we are using a high radius of 0.6 for the generation of approval preferences, it is very usual that a voter is completely satisfied, so she is not inclined to deviate and the average number of deviations is low. As k increases, the number of completely satisfied voters decreases, thus it is more likely that a voter improves her utility by deviating from her true preferences. Consequently the average number of deviations per game also increases. However, as k approaches m ($m = |C|$), voters are less willing to deviate, given that it is very probable that all (or almost all) of their approved candidates are already in the winner committee.

Alternatively, for the Threshold Heuristic, we concluded in Subsection 5.1.3 that the committee size has no effect on the average number of deviations done on average per game.

6 Experiment Set Up with Mixed Heuristics and Analysis of Results

In this chapter we will analyze games in which the different heuristics will be playing against each other, in contrast to Chapter 5 were games only contained voters using the same heuristic (except for the Factional experiments). For this purpose, we have simulated a game with 25 players. 5 of the voters used the Truthful Heuristic, 5 voters the Threshold Heuristic (with a threshold of 85%), 5 voters the Iterative Single Heuristic, 5 the Trial-and-Error Heuristic and the other 5 the Factional-Reactive Heuristic; those 5 voters belong to the same faction and will use the Iterative Single Heuristic for computing the better response for the faction. We have chosen these values as we want to analyze which heuristic performs the best so we wanted to give each heuristic the same number of voters. Additionally, we chosen 25 voters as we think it is a reasonable value from which valid conclusions can be drawn, but a small enough value so that the execution time is feasible for the time constraints of this thesis. The preference profile was generated using the spatial model with a radius of 0.6 and 20 instances were played. This experiment was done for both voting rules: sequential PAV and Phragméns. The concept of Success Probability will also not be used for this chapter, because of the same reason as explained in Section 5.4. A summary of the results can be seen in Table 6.1, and will be explained in the following sections.

6.1 Sequential Proportional Approval Experiment

A total of 77 deviations were played by the agents in all the 20 games, which gives an average of 3.85 deviations per game. All the 20 games reached a stable state, and had an average number of iterations equal to 132.7 per game. 6 games ended with a different final winner committee compared to the initial one, the rest of them had the same final and initial winner committees. The average initial social welfare was 73.4 and the average final social welfare was 72.55, thus the average DPoA of the 20 games was 1.012. Not a single game in this experiment ended with a DPoA below 1. Out of the 20 games 15 games ended with the same initial and final social welfare, while 5 ended with a lower final social welfare compared with the initial one. For those 5 games the DPoA was above one. The highest out of those 5 games was game 17; which had a DPoA equal to 1.096.

Focusing on game 17, in total 17 deviations were made by the voters. The game ended with a stable state after 285 rounds. The initial social welfare was 91 and the

	Sequential PAV	Phragmén
Total number of deviations	77	99
Average number of deviations per game	3.85	4.95
Total number of iterations	2654	2742
Average number of iterations per game	132.7	137.1
Number of games that reach a stable state	20	20
Average initial Social Welfare	73.4	74.6
Average final Social Welfare	72.55	74
Average DPoA	1.012	1.008
Worst DPoA	1.096	1.084
Factional voters' deviations	10	20
Threshold voters' deviations	57	50
ISH voters' deviations	6	21
T-and-E voters' deviations	4	8
Truthful voters that experienced an increase in utility (in the 20 games)	0	4
Truthful voters that experienced a decrease in utility (in the 20 games)	7	3
Games in which the faction experienced an increase in utility (in the 20 games)	0	0
Games in which the faction experienced a decrease in utility (in the 20 games)	1	0
Threshold voters that experienced an increase in utility (in the 20 games)	1	2
Threshold voters that experienced a decrease in utility (in the 20 games)	8	4
ISH voters that experienced an increase in utility (in the 20 games)	7	4
ISH voters that experienced a decrease in utility (in the 20 games)	4	5
T-and-E voters that experienced an increase in utility (in the 20 games)	7	4
T-and-E voters that experienced a decrease in utility (in the 20 games)	4	5

Table 6.1: Results table for the experiments with mixed heuristics.

final social welfare was 83. The 17 deviations were made only by the voters using the Threshold Heuristic (voters: $\{10, 11, 12, 13, 14\}$). Each voter using the Threshold Heuristic deviated at least once, and none of the experienced an increase in utility. Only voters 14 and 12 remained with the same final utility compared with the initial one, the rest of the voters using the Threshold Heuristic suffered a decrease in utility. The faction experienced no change in utility as well as the Trial-and-Error voters, even though the final winner committee was different from the initial one. Two of the truthful voters had a decrease in utility by one unit and the rest had no change in utility. Finally, 4 of the Iterative Single voters experienced a decrease in utility and the other one had an increase in utility, although she had not deviated from her initial true preferences.

In all of the 6 games in which the final winner committee was different from the initial one, utility either decreased or stayed the same for Truthful voters after the manipulations made by some of the strategic voters. The first game of the 20 games played, was the only one in which the committee had changed, but the utility for all truthful voters had not decreased. For the rest of the games in which the winner committee changed there was a decrease in utility for at least one truthful voter.

Out of the 20 games the faction deviated as a group in 2 games, hence in total the factional voters deviated 10 times, as it did not happen that the faction had to undo a previous manipulation because an outsider's deviation had affected negatively the signaled vote. In both games in which they deviated, their final utility was the same as the initial one, even though in both cases the final winner committee was not the same as the initial one. The reason for this is that deviations made by outsiders, before or after the faction had manipulated as a group, caused the utility of the faction to decrease. Out of the 20 games, only in one of them occurred that the faction had a final utility lower than the initial utility.

The Threshold Heuristic was the heuristic that deviated the most, with a total of 57 deviations. It is interesting to point out that in all the 20 games, only in one a Threshold voter ended with a higher final utility. However, a total of 8 Threshold voters experienced a decrease in utility throughout the 20 games.

The Iterative Single voters deviated a total of 6 times in all 20 games. From the the 6 games in which the final winner committee was different from the initial one, 3 had at least one Iterative Single vote who experienced an increase in utility, 2 had no difference in utility for all the Iterative Voters and only 1 game had an Iterative Single voter who suffered a decrease in utility.

Among the strategic heuristics the Trial-and-Error heuristic was the heuristic that performed the lowest number of deviations, with a total of 4 deviations in 20 games. Throughout the whole experiment, 7 Trial-and-Error voters had an increase in utility and 4 had a decrease in utility.

6.2 Phragmén's Rule

To a certain extent, the results for Phragmén's voting rule were different. A total of 99 deviations were made in total, which means that on average 4.95 deviations were made per game. This value is higher than with sequential PAV. Additionally, a game took on average 137 iterations to reach a stable state and it is also higher with sequential PAV, which can be expected given the higher number of deviations performed with Phragmén's voting rule. No game ended without reaching a stable state, as with sequential PAV. Out of the 20 games, 15 ended with the same social welfare and 5 with a higher social welfare, compared to the initial social welfare. Similarly to the games played with sequential PAV, no game had a higher social welfare in the final state than the initial social welfare. 5 games ended with a different winner committee and 15 with the same winner committee, compared to the initial winner committee computed from the true preferences submitted by voters in the first round of the game.

The average DPoA was 1.008 and the worst game in terms of DPoA was game 19. Said game started with a total utility of 77 but ended with a final total utility of 71, i.e. the DPoA of game 19 was 1.084. It is reasonable to see that game 19 was game to have the biggest decrease in total utility, as it was the game in which the most deviations were made (a total of 43). Out of those 43 deviations 6 were made by the Trial-and-Error voters, 3 of those voters suffered a decrease in utility and the other 2 had no change in utility. 14 deviations were made by the Iterative Single voters, which had a "worse luck" than the Trial-and-Error voters, as 4 of them experienced a decrease in utility and the other remained with the same utility as in the initial round of the game. The faction deviated 4 times as a group, therefore a total of four times a new better voted was signaled to the factional voters and 20 deviations were made by them. However, they did not accomplish an increase in utility, as their final utility was the same as the initial one. The 3 deviations remaining were made by the Threshold voters. For one of them, there was a decrease in utility, for two of them the utility increased and for the rest it remained unchanged. Surprisingly, the voters that finished the game with the best result were the Truthful voters, as 3 of them experienced and increase in utility and only one suffered a decrease in utility. Therefore, the deviations made by other voters did not affected the Truthful voters as badly as with sequential PAV. This is unexpected, as usually it is the case for voters who are not taking strategic decisions to have a disadvantage against voters who do take strategic decisions.

In contrast to the previous voting rule, with Phragmén's rule utility increased for 4 voters that were using the Truthful Heuristic throughout the 20 games played, even though they did not perform any deviations. Furthermore, utility only decreased for 3 Truthful voters during the 20 games.

The faction voters deviated with this voting rule the same amount of times as with sequential PAV (20 deviations). As previously explained in this chapter, these deviations were performed in game 19, hence the faction only deviated in one single game.

With Phragmén's voting rule, it also happened that the Threshold Heuristic was the one that deviated the most, with a total of 50 deviations done in the 20 games. Nonetheless, in most of the games Threshold voters did not experience a change in utility, given that in the whole experiment only 2 Threshold voters had an increase in utility, while 4 had a decrease in utility.

The Iterative Single Heuristic performed many more deviations with Phragmén's voting rule than with sequential PAV. During the 20 games, a total of 21 deviations were made by this heuristic, but only 4 voters experienced an increase in their utility, while 5 voters suffered a decrease in utility.

Trial-and-Error also did perform more deviations with this voting rule (compared to the deviations made with PAV): a total of 8. Moreover, voters using this heuristic experienced and increase in utility 5 times. Alternatively, 4 voters suffered a decrease in utility in all the 20 games.

6.3 Conclusion of the Mixed Experiments

From these experiments we have drawn conclusion that we would like to comment. First of all, we would like to point out that the games played with sequential PAV showed us that voters using the Truthful Heuristic are in a clear disadvantage. This was not the case when Phragmén's Rule was used, as seen in game 19; where the Truthful voters were the most successful ones. Additionally, the heuristics that were checking whether there was an increase in utility after their deviations (Factional, ISH and T-and-E) usually tend to deviate more when Phragmén's voting rule is being used, rather than sequential PAV.

It is also now clear that using the Threshold Heuristic with a equal or below 85% it is not as beneficial as we initially thought. The reason being that, even though the Threshold Heuristic was the one making the most strategic deviations, it was more common (in both voting rules) that those deviations made led to a decrease in utility rather than an increase in utility. We have not done experiments with mixed heuristics and higher values of the threshold, but we can estimate from the results obtained from Section 5.1 that the Threshold Heuristic would probably not only deviate less, but also make better deviations that would lead to fewer instances in which there is a utility decrease for the Threshold voter. It is reasonable to observe these results from the Threshold Heuristic, as from what we have described in 2.2, voters using this heuristic do not check whether the deviations made by them lead to an increase in utility in the directly subsequent round. This coincides with the conclusions from Chapter 5, where we stated that more deviations does not mean "smarter" heuristic. Furthermore, not only is this heuristic in many cases damaging for the utility of the voters using it, but also for the rest of the voters in the election that are not using it. This is clear given the fact that from both voting rules the worse game, in terms of DPoA, was game 17 from sequential PAV, in which Threshold voters had deviated 17 times.

The two heuristics that have performed the best, have been the Trial-and-Error and Iterative Single heuristics. The two of them were the only heuristics (in seqPAV) that have a higher number of voters experiencing an increase in utility than the number of voters suffering a decrease. With Phragmén's voting rule both heuristics did not perform as well, but still gave positive result in some games.

The results of these experiments have help with consolidating the conclusions drawn with the previous experiments of Chapter 5. In that chapter, we observed that the Factional Heuristic was not making as many deviations as one would expect. This was also perceived in the experiments done in this chapter. The reason for this was given in the previous chapter, where we concluded that in the games where the faction does not deviate as a group it is usually because the factional voters are already satisfied enough and not inclined to deviate. A similar conclusion was studied in [3], which introduced the concept of *Median Voter Theorem*. This concept claims that ”a majority rule voting system will select the outcome most preferred by the median voter”. Even though the voting rules considered in this thesis are not exactly majority voting rules, it still happens in many cases that the outcome of the election pleases a ”median voter”. Therefore, given that the voters of the faction have exactly the same preferences and there are many of them, it is very likely that the ”median voter” has preferences close to the ones of the factional voters (especially for large factions). Hence the faction is not as inclined to deviate.

We now have a stronger intuition that games with more deviations tend to have worse DPoA scores. From the games we have seen in this chapter, we can conclude that the worst ones, in terms of DPoA, have been the games in which the most number of deviations have been played. This is a reasonable result, since games with more deviations tend to lead to lower values of social welfare. The reason for this being that, even though the deviations made by voters cause an increase in their own utility (they are local improvements), those deviations often decrease the total utility for the whole group of voters.

7 Summary and Conclusion

To conclude we would like to go over the work that has been done for this thesis. First of all we have introduced in Chapter 1 the setting of Iterative Voting, describing its main characteristics and stating that it is a good framework to study, even though it might not seem as a conventional as other voting settings, since it can be found in many real life scenarios. As presented in [11], Iterative Voting is a good framework to use in order to analyze how strategic voting can affect the outcome of an election. We also explained that an iterative game can be modelled as a directed graph, where each node represents a voting profile and each edge a deviation made by a single voter. Then, we defined Nash Equilibria as the set of nodes in the directed graph with no outgoing edges.

We combined this framework with two different approval multi-winner election rules: Proportional Approval Voting (PAV) and Phragmén's Voting Rule, in the Section 1.4. Said voting rules can be found in [2]. However, before going into detail on how those different voting rules computed the winner committees, we introduced concepts related to approval multi-winner elections such as approval ballots, winner committee, voting profile, set of voters and set of candidates. After this, we explained both previously mentioned voting rules and introduced their sequential variants. The reason for this being that computing the outcome of a multi-winner election rule is NP-Hard for PAV (as explained in [6]) as well as for Phragmén's Voting Rule (as explained in [2]). These sequential variants were be used in subsequent chapters, in which experiments were done to asses their behaviour with voting heuristics. As possible future work it would be very interesting studying other multi-winner election rules in iterative settings as Approval Voting (AV), Minimax Approval Voting (MAV) or Chamberlin–Courant and Monroe Approval Voting (CCAV and MonroeAV).

We then started Chapter 2 by defining the possible types of manipulations that can be taken by voters, when deviating from their true preferences as: *subset* and *superset* manipulations. Consequently, we introduced the simplest heuristic considered in this academic paper, Truthful Heuristic, which models a voter that never deviates from its true preferences, i.e. never puts effort on computing a better response to the current voting profile. Taking as a starting point the heuristics presented in [11], we continued the chapter by introducing more complex heuristics. The first one Threshold Heuristic, based on the idea that if a voter disapproves a "popular" candidate, there is a possibility that the candidate will remain in the winner committee, but the rest of the voter's approved candidates will gain more weight in the voting rule. Therefore, the rest of the approved candidates have a higher chance of entering the winner committee. This heuristic has a low complexity and a Threshold voter can disapprove a maximum of 1 candidate per round. Two different variants for this heuristic were proposed: First Candidate Above Threshold and Highest Candidate Above Threshold, as seen in 2.2.1.

Then, we introduced the Iterative Heuristic, an even more complex heuristic, since a voter using it simulates the outcome of the game before taking a decision. The reason being that the voter wants to see how its utility changes after taking such deviation. The voter will only make the deviation if there is a strict increase in utility. Therefore, the deviations made by an Iterative voter can only lead to local improvements of utility. We proposed two different variants to this heuristic: Iterative Single (IS), which allows the approval or disapproval of one candidate per round; and Iterative Up to J, which allows the approval or disapproval of up to j candidates per round. We then claimed that out of these variants the only that guarantees computing the best response is Iterative Up to m ($m = |C|$), as it considers all possible deviations in each turn, and takes the one that maximizes utility increase.

The next heuristic we introduced was Trial-and-Error, which was inspired from [10]. This heuristic is equivalent to the IS heuristic when calculating a better response for a voter to the current state of the game. However, instead of looking at all the possible deviations consisting of one change in the approval score of a candidate, the heuristic selects one deviation at random and takes it, only if it leads to an increase in utility. Otherwise, the voter submits its current vote. Additionally, in each turn the heuristic gives Trial-and-Error voters the opportunity of submitting the initial vote (undoing all previous manipulations), if the voter has suffered a decrease in utility, due to other voters' manipulations. The motivation for this heuristic was developing a heuristic similar to the Iterative Heuristic, but with a lower complexity and with the ability to quickly counteract against other voters' deviations.

Another heuristic considered was the Same Vote Factional Heuristic. The difference between the Same Vote Factional Heuristic and the rest of the heuristics presented before, is that we now allow a set of voters (known as a faction) to deviate from their true preferences as a group. The voters belonging to the faction start with the same true preferences and must end with the same vote. In order to calculate a new better response for the faction, any heuristic previously considered can be used. With this heuristic we also introduced the concept of Success Probability , which denotes the probability that a factional voter has of convincing the rest of the faction to deviate in the same way as the voter has calculated. By introducing this concept we wanted to model real life scenarios, where voters belonging to a group may have a hard time convincing the rest of the faction. After this, we introduced a variant to the Same Vote Factional Heuristic: Same Vote Reactive Factional Heuristic. With this new variant, factional voters are able to cancel a manipulation if an outsider's deviation has caused the faction's manipulation to be harmful to the faction's utility. This makes the heuristic more complex, but in turn makes the factional voters "smarter". We also provided decision diagrams for both the Same Vote Factional Heuristic and the Same Vote Reactive Factional Heuristic (Figures 2.1 and 2.2) for the better understanding of these two heuristics. We would like to mention that it would be very interesting, as possible future work, the development of a factional heuristic that allows the faction voters to finish with a different vote. We think this would be very compelling, as it gives the factional much more flexibility when deviating. However, it will increase the complexity of computing a manipulation for the faction considerably.

We continued the thesis in Chapter 3 by looking at Nash Equilibria in our setting. Given that our voters usually do not compute the best response to the current state of the game, we stated that it is not guaranteed to have reached a Nash Equilibrium once a game containing at least one heuristic described in this thesis has finished. Instead, we can guarantee that we have reached a stable state, which is a relaxation of Nash Equilibria. A stable state is a state in our iterative voting game, where no voter (after applying one of the heuristics previously described) wants to deviate from its current vote, even though there may exist a deviation that yields an increase in utility. After introducing stable states, we looked at which heuristics guaranteed reaching a stable state and got the following results: heuristics that perform just one type of manipulations (either subset or superset), such as the Threshold Heuristic guarantee reaching a stable state; while heuristics that are able to perform both types of manipulations do not, such as Trial-and-Error or Iterative Heuristics. Factional guarantees reaching a stable state whenever the heuristic used to calculate the deviations to be done by the faction also does. Consequently, we presented a game in which voters did not reach a stable state, since the game got stuck on a loop of deviations between two Iterative Single voters. We finished Chapter 3, by introducing the concept of Dynamic Price of Anarchy (DPoA), which is used to measure the quality of the stable state reached with respect to the social welfare of the outcome.

In Chapter 4 we presented three different approaches for the problem of generating random approval preference profiles for voters. Two of the approaches suggested were developed by us (4.2 and 4.1) and the third one 4.3 taken from [9]. The third is the most elaborated one, since it uses a Spatial Model for the generation of random approval preferences and is the solution used for this mentioned problem in the subsequent chapters.

Furthermore, we set up experiments for the different heuristics, in which an experimental variable (number of voters, candidate list size or winner committee size) was varied while fixing the other two to constant values. In those experiments, all voter were using the same heuristic and their preferences were generated using the Spatial Model approach, with a radius of 0.6. This values was taken, as after experimenting with other values, 0.6 gave approval preference profiles suitable for the experiments conducted. However, we think it would be very valuable to experiment with different radii, in addition to experimenting with different approaches for random approval preference generation. Each experiment was conducted using both of the sequential variants of PAV and Phragmén's Voting Rule. The intention of these experiments was not only to study the interactions between the heuristics and voting rules, but also the potential impact each heuristic could have on the outcome of the game. Many conclusions were taken from the experiments, and we will only be focusing on the main conclusions drawn.

First of all we wanted to comment that for most of the experiments, the voting rules that were considered in the chapter produced similar results, i.e. they had a similar behaviour when experimenting with the different heuristics. This was not an expected result, given that PAV and Phragmén's voting rules have different objectives and ways for calculating the winner committee.

Secondly, from the results obtained we observed that not finishing a game with a stable state is more common than initially expected, especially when using "smart" heuristics that consider a many possible deviations before taking a decision. This assumption can be backed up with the fact that games with Iterative Up to 3 voters were running into many more cycles (and not reaching a stable state) than other heuristics that consider less possible deviations, such as Trial-and-Error or Iterative Single Heuristics. When increasing experimenting with the committee size this was particularly problematic, given that games with the Iterative Up to 3 Heuristic were running into many cycles. This was so computationally expensive that the experiment was not feasible given the time limit of the thesis. This conclusion is also backed up with the fact that every single game in the Chapter 6 ended with a stable state. In such experiments the heuristic Iterative Up to 3 was not played.

Finally, we would like to point out our conclusion related to Dynamic Price of Anarchy (DPoA). From the results we have seen that usually the games which have the DPoA values further away from one, are the games in which the most deviations occur. The reason being that the more deviations happen on a game tend to cause the winner committee to go through more changes. Additionally, the deviations taken by voters, even though they lead to an increase in their own utility, usually cause a decrease in the rest of the voters' utility. Therefore, the social welfare tends to decreases as the game goes on, and we end with a lower final social welfare compared to the initial one. As a result we usually have a DPoA above 1, which is never a good result (in terms of social welfare), since it is telling us that our voting system is degrading due to the selfish behaviour of our voters.

Due to the computational restrictions, we had to limit some experimental variables (such as number of voters and candidate list), particularly for more complex heuristics such as the Iterative Heuristic. Thus, we consider that it would be a compelling future work, repeating these experiments with higher values of the experimental variables. This would give us a better understanding of how the heuristics behave in this iterative setting.

For the previous experiments made in Chapter 5, each game contained only one type of heuristic, except for the Factional experiments. For this reason, we wanted to see how the heuristics behaved with each other, when competing against one another. Therefore, in Chapter 6 we set up 20 games for each voting rule, with 25 voters, a committee size of 4, a candidate list of size 8 and the Spatial Model approach for generating the random approval preferences (again with a radius of 0.6). Additionally, we gave an equal distribution of heuristics (5 voters using each heuristic) as explained in the introduction of Chapter 6. We used this equal distribution, as we wanted to give all the heuristics the same weight in the election. After simulating the 40 games we can extract the following main conclusions.

In contrast to Chapter 5, the sequential variants of PAV and Phragmén presented more diverse results. From the data obtained in Table 6.1, we can see that the average number of deviations (and consequently the number of iterations) on average per game was higher with Phragmén's Voting Rule that with seqPAV. Moreover, the Thruthful voters were far worse in the games played with seqPAV, as there was no game out of the 20 in which they experience an increase in utility. This was not the case for Phragmén's

Voting Rule. The average DPoA was also slightly better with Phragmén's Voting Rule than with seqPAV.

Finally, the results from these experiments, also helped in consolidating the conclusions drawn from the experiments in the previous chapter. The reason being that the behaviours and patters observed in Chapter 5, such as Factional voter not deviating that much or Threshold voters performing many deviations that lead to high values of DPoA, are also presented in the results from Chapter 6.

As future work we think it would be interesting experimenting with different heuristic distributions, as well as studying real life scenarios with the goal of finding a distribution that approximates how different humans vote.

Acknowledgements

First of all, we would like to thank Prof. Dr. Markus Brill, for allowing to work with this project as well as for being the tutor during this thesis. Additionally, we would like to thank Dr. Anne-Marie George; for being the supervisor of the thesis, and Dr. Jobst Heitzig for his advice and recommendations. Finally, we would like to acknowledge Prof. Dr. Félix Francés for providing with constant help and support.

References

[1] K. J. Arrow. Public and private values. *Human values and economic Policy*, pages 3–21, 1967.

[2] H. Aziz, M. Brill, V. Conitzer, E. Elkind, R. Freeman, and T. Walsh. Justified representation in approval-based committee voting. *Social Choice and Welfare*, 48:461–485, 2017.

[3] D. Black. On the rationale of group decision-making. *Journal of political economy*, 56(1):23–34, 1948.

[4] S. J. Brams and P. C. Fishburn. Approval voting. *The American Political Science Review*, pages 831–847, 1978.

[5] V. Conitzer and T. Walsh. Barriers to manipulation in voting. *Handbook of Computational Social Choice*, pages 127–145, 2016.

[6] P. Faliszewski, J. Lang, and P. K. Skowron. Finding a collective set of items: From proportional multirepresentation to group recommendation. *Artificial Intelligence*, 241, 2014.

[7] P. Faliszewski, I. Rothe, and J. Rothe. Noncooperative game theory. *Economics and Computation*, pages 41–122, 2015.

[8] A. Gibbard. Manipulation of voting schemes: a general result. *Econometrica: journal of the Econometric Society*, 41(4):587–601, 1973.

[9] G. Hang. Bachelor's thesis: Evaluating the representativeness of approval-based ranking rules. 2015.

[10] J. Heitzig and F. W. Simmons. *Efficient democratic decisions via nondeterministic proportional consensus.* Final report in preparation, can be found in: https://arxiv.org/abs/2006.06548.

[11] R. Meir. Iterative voting. *Trends in Computational Social Choice*, pages 69–86.

[12] J. Nash. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49, 1950.

[13] J. Nash. Non-cooperative games. *Annals of mathematics*, pages 286–295, 1951.

[14] E. Phragmén. *Proportionella Val: en valteknisk studie.* Hökerberg, 1895.

[15] E. Phragmén. Sur la théorie des élections multiples. *Öfversigt af Kongliga Vetenskaps-Akademiens Förhandlingar*, 53:181–191, 1896.

[16] E. Phragmén. Till frågan om en proportionell valmetod. *Statsvetenskaplig Tidskrift*, 2(2):297–305, 1899.

[17] E. Phragmén. Sur une méthode nouvelle pour réaliser, dans les élections, la représentation proportionnelle des partis. *Öfversigt af Kongliga Vetenskaps-Akademiens Förhandlingar*, 51(3):133–137, 1894.

[18] T. Roughgarden. Selfish routing and the price of anarchy. *Twenty Lectures on Algorithmic Game Theory*, page 145–158, 2016.

[19] M. A. Satterthwaite. Strategy-proofness and arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of economic theory*, 10(2):187–217, 1975.