# Information and Representation in Computational Social Choice

**MSc Thesis** (*Afstudeerscriptie*)

written by

**Ilan Frank**
(born November 11, 1981 in Gan-Shmuel, Israel)

under the supervision of **Dr Ulle Endriss**, and submitted to the Board of Examiners in partial fulfillment of the requirements for the degree of

**MSc in Logic**

at the *Universiteit van Amsterdam*.

| **Date of the public defense:** | **Members of the Thesis Committee:** |
|---|---|
| *September 6, 2011* | Prof Dr Krzysztof Apt |
| | Dr Ulle Endriss |
| | Umberto Grandi |
| | Prof Dr Benedikt Löwe |
| | Prof Dr Peter van Emde Boas |

Institute for Logic, Language and Computation

**Abstract**

Voting rules are functions aggregating the preferences of voters regarding a set of alternatives, and the theoretical properties of these objects are investigated in voting theory, as well as in computational social choice, where we study their computational aspects.

The question of how much information such rules actually need in order to compute their result has been studied from various angles in recent years, via concepts such as the communication complexity (Conitzer and Sandholm), compilation complexity (Chevaleyre et al.), and informational size (Sato). We review these concepts and analyze the relations between them. That will lead us to a discussion about representation of information: We describe different representations for different voting rules and construct a hierarchy of those representations. We then discuss generalized scoring rules (introduced by Xia and Conitzer), and show how they relate to our previous discussion.

Finally, we build a correspondence between one representation for voting rules and subgroups of the symmetric group, thus showing a connection between group theory and informational representations in voting theory. We use that correspondence to prove a theorem posed by Sato in his paper on informational size.

**Acknowledgements**

First and foremost, I would like to thank my supervisor, Dr Ulle Endriss, for all his help and encouragement, and especially for giving me the freedom to pursue the directions which I found most fascinating, and which ultimately led me to this work.

I also thank the other committee members: Dr Krzysztof Apt, Umberto Grandi, Dr Benedikt Löwe, and Dr Peter van Emde Boas, for all their important remarks, corrections and suggestions.

I thank Dr Vincent Conitzer, Dr Jerome Lang, Dr Shin Sato, and Dr Lirong Xia for clarifying me various details in their work, on which much of this thesis is based.

I am grateful to Mark Polak, Gabriela Aslı Rino Nesin, and Navid Talebanfard for all their useful comments, and to Erik Parmann for \qedhere.

Finally, I want to thank the Huygens Scholarship Programme for supporting me during my studies in the Netherlands.

# Contents

# Chapter 1

# Introduction

Social choice theory in general, and voting theory in particular, are about aggregating preferences of different agents with regard to a set of alternatives. The objects which take the preferences of the agents and produce the collective decision are called voting rules, and they are the main objects of inquiry in the field.

On the formal side of the theory, these objects are usually described as functions which take as input the preferences of the voters and give as output the set of winning alternatives. However, when one actually wishes to describe a specific rule, that description is usually quite "wordy", giving an algorithm rather than a detailed function.

These algorithmic descriptions of voting rules are not necessarily bad. Indeed, in computational social choice, which studies various computational aspects of social choice theory, voting rules can be naturally embedded as computer programs. That description, though, has its limitations. There are many properties of voting rules which we cannot learn from the programs associated with them – or at least it would require unreasonable computational resources. These are theoretical questions such as: Are two voting rules identical (which would require one example to answer negatively, but perhaps going over an infinite number of inputs to answer positively), or on which sets of preferences will two voting rules coincide?

It seems that such questions cannot be answered based on an algorithmic description alone, but require some theoretical analysis of the voting rules, or a mathematical description which is finer than just functions from preferences to alternatives. One way to go about in this direction is to study the way different voting rules use information. The information in social choice theory is the preferences of the agents, and by studying how a voting rule utilizes that information we might have some better understanding of voting rules.

This thesis focuses on the topic of information in computational social choice, studying various problems and descriptions which were suggested in this area, in the hope that such descriptions can bring forth a more detailed representation of voting rules, which will enrich our understanding of the field.

These ideas have already been studied to some degree in recent years. The most formal representation suggested so far, generalized scoring rules, will be described in Chapter 4. Other information-related problems which we will be dealing with are communication complexity and compilation complexity, seeing how they characterize rules and what representations they offer. We will also suggest a direction of inquiry which might lead to an algebraic representation

of voting rules.

We begin by describing voting theory in a more detailed way, giving all the basic formal definitions, and mentioning some of the famous theorems and approaches in the field. We later describe computational social choice, which has been a very productive field in recent years. In the end of this chapter we present an outline of the thesis.

## 1.1 Voting Theory

In this section we give all the definitions we need from voting theory, and describe some of the popular approaches in the field. A more general introduction can be found in Gaertner [9]. We start with the standard basic definitions in voting theory:

**Definition 1.1.1** (Voting framework)**.** *We will be speaking of a set of* ***alternatives*** *(or* ***candidates****) $X$ of size $m$ and a set of $n$* ***voters*** *(or* ***agents****) $N = \{1, \ldots, n\}$.*

*The set of strict linear orders on $X$ is denoted $\mathcal{L}(X)$ (or sometimes, when $X$ is clear, just $\mathcal{L}$). An element $V \in \mathcal{L}(X)$ is a* ***ballot****. $V^i$ denotes the $i$th element in the order $V$.*

*A* ***profile*** *$P = \langle V_1, \ldots, V_n \rangle$ is an $n$-tuple of ballots, where the $i$th element is the ballot submitted by voter $i$. $\mathcal{L}(X)^n$ is the set of all profiles for $n$ voters. We also denote the set of all profiles (for any finite number of voters) by $\mathcal{L}(X)^*$.*

*A* ***voting rule*** *for $m$ alternatives and $n$ voters is a function $\mathcal{F}_{m,n} : \mathcal{L}(X)^n \to \mathcal{P}(X) \setminus \{\emptyset\}$. The set $\mathcal{F}(P)$ is called a winning set, and its elements are the winning candidates/alternatives.*

*For a profile $P = \langle V_1, \ldots, V_n \rangle$ and $x, y \in X$, denote $x \succ_{P,i} y$ if $x$ is preferred to $y$ in $V_i$, and denote*

$$N(x, y, P) = |\{i : x \succ_{P,i} y\}|$$

*So $N(x, y, P)$ is the number of voters who prefer $x$ to $y$. In addition, denote for any $1 \leq j \leq m$*

$$N(x, j, P) = |\{i : V_i^j = x\}|$$

*the number of voters who rank $x$ in the $j$th place in $P$.*

A list describing all common voting rules that are mentioned in this thesis can be found in Appendix A. We give an example of a profile under a few of the voting rules described in the list. Let our candidates be $X = \{a, b, c, d\}$ and let $n = 3$ (4 candidates and 3 voters). Let our profile be:

$$P = \begin{cases} a > b > c > d \\ a > b > c > d \\ b > c > d > a \end{cases}$$

For such a profile we get that the plurality winner is $a$ (2 out of 3 voters rank $a$ first), the Borda winner is $b$ (with a Borda score of 7, higher than that of any

other candidate), and the veto winners are $c$ and $b$ (both never voted last). We also get that for any $x \in \{b, c, d\}$ we have $N(a, x, P) = 2 > 1 = N(x, a, P)$, which makes $a$ also a Condorcet winner, and so the winner in all the Condorcet consistent rules. Finally, $N(a, 1, P) = 2$, which is more than half the votes, and so the Bucklin score of $a$ is 1, which makes it the Bucklin winner.

As was mentioned, voting rules are often described as an algorithm, where $m$ and $n$ are just parameters, but the rule can be fitted for any number of voters and alternatives. That is to say, such a rule $\mathcal{F}$ is actually a function whose domain is $\{\mathcal{L}(X)^* : X \text{ is a finite set}\}$. When we are talking about a rule with a clear mechanism, or when $m$ and $n$ are clear from the context, we will sometimes omit the subscript.

Another remark is about our terminology. There are various definitions and terms in the literature regarding voting rules. While in some places those are defined as they are defined above, in other places a voting rule must elect a single winner for any profile (that is, it does not allow "ties", as our definition does). This type of voting rules is also referred to as "resolute" at times. We will stick to the definition above, as resolute voting rules often add an asymmetric component to the system (we can always have a profile which is symmetric with respect to all candidates, but then choosing a single winner "breaks the symmetry"). Another reason is that any voting rule $\mathcal{F}$ (in our sense) can be made into a resolute by composing some tie breaking rule on top of $\mathcal{F}$.

A lot of the characterization of voting rules in voting theory is done by defining axioms and then looking only at the rules satisfying them. We present some of the more important axioms, some of which we will use extensively in the thesis. The first axiom – anonymity – simply says that the voting rule is symmetric with respect to all voters:

**Definition 1.1.2** (Anonymity)**.** *A voting rule $\mathcal{F}$ is anonymous if for every profile $P = \langle V_1, \ldots, V_n \rangle$ and every $\sigma$, a permutation of the set $N = \{1, \ldots, n\}$ (i.e., a bijection from $N$ to $N$), we have*

$$\mathcal{F}(\langle V_1, \ldots, V_n \rangle) = \mathcal{F}(\langle V_{\sigma(1)}, \ldots, V_{\sigma(n)} \rangle)$$

A rule that is not anonymous, for example, is a rule that always chooses the alternative ranked first by voter $i$, for some fixed $1 \leq i \leq n$. Such a rule is called "dictatorial" and voter $i$ is called "the dictator", for obvious reasons.

Another important axiom which is also about symmetry, though symmetry among alternatives and not voters, is the following:

**Definition 1.1.3** (Neutrality)**.** *Given a permutation $\sigma$ of $X$ and a ballot $V = \langle x_1 > x_2 > \cdots > x_m \rangle$ we define $\sigma V = \langle \sigma(x_1) > \sigma(x_2) > \cdots > \sigma(x_m) \rangle$, which is also a ballot. Given a profile $P = \langle V_1, \ldots, V_n \rangle$ we define $\sigma P = \langle \sigma V_1, \ldots, \sigma V_n \rangle$. In addition we define, for a subset $A \subseteq X$, $\sigma(A) = \{\sigma(x) : x \in A\}$.*

*We say that a voting rule $\mathcal{F}$ is neutral if for every profile $P$ and a permutation $\sigma$ of $X$ we have*

$$\mathcal{F}(\sigma P) = \sigma(\mathcal{F}(P))$$

Neutrality means that the rule "doesn't care" about specific candidates. If $x$ was winning before (given some profile), and we rename $x$ to $y$ (in that profile), then now $y$ will win.

Neutrality and anonymity create a sort of symmetry in the system, which is usually desired. Most if not all the common voting rules are both neutral and anonymous (up to implementation of tie breaking rules). Other important axioms are monotonicity (there are two versions of monotonicity: weak and strong; we will only be interested in the weak one) and positive responsiveness:

**Definition 1.1.4** (Monotonicity, Responsiveness)**.** *For two profiles $P_1, P_2$ and $x \in X$, we say that we get $P_2$ by lifting $x$ in $P_1$, if for any $y, z \neq x$ and $1 \leq i \leq n$ we have $y \succ_{P_1,i} z \Leftrightarrow y \succ_{P_2,i} z$ and $x \succ_{P_1,i} y \Rightarrow x \succ_{P_2,i} y$.*

*A voting rule $\mathcal{F}$ is (weakly) monotonic if for any two profiles $P_1, P_2$ and a candidate $x \in \mathcal{F}(P_1)$, if we get $P_2$ by lifting $x$ in $P_1$, then $x \in \mathcal{F}(P_2)$.*

*A voting rule $\mathcal{F}$ satisfies positive responsiveness if for any two profiles $P_1 \neq P_2$ and a candidate $x \in \mathcal{F}(P_1)$, if we get $P_2$ by lifting $x$ in $P_1$, then $\mathcal{F}(P_2) = \{x\}$.*

Informally, the idea of lifting is that the preferences of the voters in $P_2$ are the same as $P_1$, except that they might rank $x$ higher. Monotonicity means that in such a situation, $x$ should not be doing worse than he was under $P_1$. Positive responsiveness, which is slightly stronger, says $x$ should be doing better. Both the axioms are usually desired properties, as they correlate with our basic intuitions about preference aggregation. Another such property is Pareto efficiency:

**Definition 1.1.5** (Pareto efficiency)**.** *A voting rule $\mathcal{F}$ is Pareto efficient if for any profile $P$, whenever we have candidates $x, y \in X$ such that $N(x, y, P) = n$, $y \notin \mathcal{F}(P)$.*

Pareto efficiency means that when we have a candidate $x$ which all voters prefer to another candidate $y$, we have no reason to elect $y$. The A-rules, N-rules, M-rules, R-rules, and P-rules stand for the sets of anonymous, neutral, monotonic, positively-responsive and Pareto efficient voting rules, respectively. We will also talk about combinations such as NA-rules – neutral and anonymous voting rules.

As was mentioned before, axioms are often used for characterization of rules. May's Theorem, for example, states that the only NAR-rule for the case of $m = 2$ (two candidates) is plurality [13]. In Chapter 3 we will see the work of Sato, where he uses a characterization based on a notion of information.

What we will do in much of this thesis is to try and view voting rules through the information they utilize in the voters' preferences, find "nice" representations for that information, and see how that can assist us in evaluating various computational properties for those rules (Chapters 2 and 3), describing those rules in a more formal way (Chapter 4), or characterizing rules (Chapter 5).

## 1.2 Complexity and Social Choice

Computational social choice, an area which has greatly developed in the last decade or so, studies various computational aspects of social choice theory. In regard to voting theory, among the topics that are studied we find the computational complexity of voting rules (how computationally hard it is to compute the winner according to some voting rule), studying the complexity of manipulating an election (manipulation is a situation where a voter/set of voters can get a better winning set if they don't express their true preferences), and – what will be of special interest to us – coming up with formal descriptions for rules and studying representations of information for different rules. For a review on computational social choice, see [2].

### 1.2.1 Complexity

The scope of this thesis cannot allow for a complete description of complexity theory, but as we will be dealing with several measures of complexity, we give a very brief description of some concepts we will use. Notions of complexity stand for different things in different contexts, but the measures we will encounter will be one of the following two:

- Computational complexity, which is measured (in an informal and simplified way) by the number of computation steps a program requires, or the size of the memory required by the program to perform that computation (this could also be defined for other computational resources, though time and space are the most commonly used).
- Data complexity, which is measured by the number of bits required to store information.

In both cases we will be interested in describing the complexity as a function $f(x)$, where $x$ is the input parameter of the problem. In addition, one is usually less interested in the value of the complexity function for specific input parameters, but rather in the limit behavior of the function when the size of those parameters grows towards infinity. The evaluation of the complexity is then usually divided in two: a search for an **upper bound** and a separate search for a **lower bound**.

Let $f(x)$ be our complexity function, which we are trying to estimate. We say that $h(x)$ is an upper bound of $f$, $f \in O(h(x))$, if for large enough $x$ we always get $f(x) \leq Ch(x)$ for some constant $C$. Similarly, we say that $h(x)$ is a lower bound of $f$, $f \in \Omega(h(x))$, if for large enough $x$ we always get $Ch(x) \leq f(x)$ for some positive constant $C$. We say that $f$ is of the same order as $h(x)$, $f \in \Theta(h(x))$ if for large enough $x$ we always get $C_1 g(x) \leq f(x) \leq C_2 g(x)$ for two positive constants $C_1, C_2$. For a complete definition of the $O$-notation, see [5, Chapter 3].

The parameters in our voting framework are the number of voters and the number of candidates, so we will take our complexity measures to be functions

of the parameters $m$ and $n$. Estimating the complexity of a function of two parameters is not simple, and so often one makes simplifying assumptions about the nature of the parameters and the connections between them (e.g., $m \ll n$, or taking $n$ to be a certain function of $m$). Those assumptions should also take into account the problem at hand. In many scenarios, for example, it makes sense to assume that the number of voters is much greater than the number of candidates.

This gives us the basic tools for defining measures of complexity, which is precisely what we will do in Chapter 2.

### 1.2.2 Voting Rules and Complexity

We give examples of the complexity of some voting rules, to get some perspective on how difficult it is to compute these functions. When we refer to the complexity of a voting rule in this section, we mean the computational complexity of finding the winner(s) according to the rule, given some profile. We avoid giving proofs and simply describe an algorithm for the rule when that is simple, or mention a result regarding the rule's complexity when it is computationally difficult.

To avoid having to introduce too many definitions from complexity theory, we simply say that a function $f(x)$ takes linear (polynomial, exponential) time if the number of computation steps required by any computer program[1] computing $f(x)$ is in $O(h(x))$ where $h(x)$ is a linear (polynomial, exponential) function of $x$.[2]

We now present algorithms for a few voting rules. The algorithms presented are not necessarily the most efficient, but they are used only to show that the rule can be computed in linear/polynomial time.

**Example 1.2.2.1** (Plurality)**.** Computing the plurality rule takes linear time, since for any profile $P = \langle V_1, \ldots, V_n \rangle$, $\mathcal{F}_P(P)$ can be computed by going over all ballots ($n$ different ballots), and for each ballot $V_i$ adding 1 to the plurality score of candidate $V_i^1$. The winners are then taken to be the candidates with the maximal score.[3]

**Example 1.2.2.2** (STV – see Appendix A)**.** Another rule which can be computed relatively fast is STV. An algorithm for finding the STV winner is the following: We iterate over all the ballots and compute the plurality score as we did for plurality. We then check if there is a candidate with more than half the votes. If not, we find the candidate(s) to be eliminated and keep track of them in a list. Now we go over the ballots again, like in the first time, only whenever we encounter an eliminated candidate we look at the following candidate in the

[1] Usually described as a "deterministic Turing machine".

[2] Strictly speaking, we should have said that $f(x)$ takes *at most* linear time.

[3] We have made some simplifying assumptions, taking addition as something that is performed in 1 computation step, as well as the access to the data. In practice the complexity might be higher, up to a factor of log. We use these assumptions also when describing the following algorithms.

ballot. The process repeats itself until there is a winner. We note that it can repeat at most $m$ times (at every step we eliminate at least one candidate). The whole computation takes polynomial time (it is in $O(m^3n)$ or $O(m^2n)$, depending on the specific implementation).

**Example 1.2.2.3** (Copeland – see Appendix A)**.** We present an algorithm for the Copeland rule, which is a Condorcet consistent rule. First, for every two candidates $x, y$ we compute $N(x, y, P)$ by going over all the ballots and checking for ballot $V_i$ if $x \succ_{P,i} y$ (in which case we increment $N(x, y, P)$ by 1) or $y \succ_{P,i} x$ (and then we increment $N(y, x, P)$ by 1). For every pair of candidates this takes roughly $mn$ steps, and there are $m^2/2$ distinct pairs, so we are in $O(m^3n)$. We then calculate $W(x), L(x), W(x) - L(x)$ for every candidate ($O(m^3)$) and in total we find that this rule can also be computed in polynomial time.

All the rules presented until now can be computed in polynomial time (or lower), and so they all belong to the complexity class known as P. Without going into too many definitions from complexity theory, we can informally say that P contains problems which can be solved in polynomial time, and it is considered to be a class of problems which is reasonable to try and compute.

However, not all voting rules are that simple. It was shown, for example, that the Kemeny rule and Dodgson rule are NP-hard [1]. NP is (again, in an informal way) the class of problems for which we can verify a solution in polynomial time, and it is believed that many of those problems cannot be computed in polynomial time, but require exponential time, which makes the computation impractical (in the language of complexity theory: intractable). As far as we are concerned, a problem is NP-hard if it is at least as difficult as any problem in NP, which means that the Kemeny and Dodgson rules are much more complex than our previous examples.

Note that although Copeland, Dodgson and Kemeny are all Condorcet consistent, and though both Copeland and Kemeny are WMG rules, as defined in Appendix A, there are big differences in their complexity. We will get back to that point in the following chapters. In the rest of the thesis, however, we deal less with computational complexity as we did in this section, and more with data complexity as described earlier.

## Outline of the Thesis

Chapter 2 introduces us to the basic notions of information that we will be covering throughout this thesis: communication complexity (developed for the study of voting rules by Conitzer and Sandholm in [4]) and compilation complexity (first presented by Chevaleyre et al. in [3]). The motivation for these measures will be explained, as well as the methods for determining upper and lower bounds for them.

In Chapter 3 we will encounter a new notion: informational size, first developed by Sato in [14]. We will explore the use of informational size and see

how it can be used to link between compilation and communication complexity. We will also compare various results for common voting rules, in terms of communication complexity, compilation complexity and informational size.

In Chapter 4 we move to a somewhat more abstract discussion about the representation of profiles for different voting rules, with emphasis on representations used for compilation complexity. We explore the relations between some representations, and then we present the generalized scoring rules, developed by Xia and Conitzer in [19], and see how that notion connects with the complexity measures we analyzed.

In Chapter 5 we return to informational size to show how it can be used to give an algebraic representation relating to voting rules. We then use that representation to prove a lemma posed by Sato in his paper on informational size [14].

# Chapter 2

# Information and Complexity Measures in Social Choice

In this chapter we present two main notions of complexity for voting rules – communication complexity and compilation complexity – which we will be working with throughout this thesis. Roughly speaking, compilation complexity measures the size of the most efficient representation of the preferences expressed in a profile, given some voting rule, and communication complexity measures the amount of information sent between voters, when they interact in the most efficient way, in order to compute the winner.

Though these two concepts are not related a priori, we hope to give some clues about the underlying connections between them, as well as to their main differences. Both the complexities are relatively new in the field of computational social choice, so there are not many papers dealing with them yet, though it seems they are becoming popular in recent years.

What is the connection between these problems and information? That question will hopefully become clearer once the problems are described, but in a general sense, we can say that both problems aim at getting a compact representation of the preferences of the voters. We stress that there are more problems which are also somewhat related to information. One example is the possible winners problem, where given a partial profile you are asked to see which candidates can still win and which cannot, and there are more examples to be found, though granted, there is no line dividing between problems which are about information and problems which are not. In this thesis, however, we concentrate mostly on the two problems mentioned above, as well as other notions which we will show are related to them, since, in this author's opinion, these problems are the most directly related to questions of representation of information in voting theory.

In the following chapter we will present another notion – informational size – and show how that can be used to connect the two complexity measures. We then give a complete list of results from the literature regarding the complexity of voting rules.

Another connection between the two concepts will be shown in Chapter 4, where we present generalized scoring rules, and show how compilation and communication complexities relate to different aspects of those.

In each of the sections below we describe the complexity measure, the motivation for defining it, the standard ways in the literature for finding upper and lower bounds on the complexity of voting rules, and give some examples of the complexity of common voting rules. We start with communication complexity.

## 2.1 Communication Complexity

The theory of communication complexity studies processes of distributed computations, where several parties engage in a joint task, each having only part of the information required to complete the task. Introduced by Yao in 1979 [22], the standard model of communication complexity includes two or more agents who are communicating in order to compute the value of some function, where each agent has only part of the input required by the function. The communication complexity of the function is the amount of information needed to be exchanged in the worst case scenario of the most efficient method of communication (a more formal definition will soon follow). A general overview on this topic is given by Kushilevitz and Nisan [11].

Conitzer and Sandholm applied ideas from communication complexity in voting theory [4]. In Conitzer and Sandholm's scenario, our voters communicate with each other in order to compute $\mathcal{F}(P)$ for $P = \langle V_1, \ldots, V_n \rangle$, where the $i$th voter only knows $V_i$ to begin with. The process ends when each of the voters known enough about the other voters' preferences in order to compute $\mathcal{F}(P)$. The compilation complexity of a voting rule is then defined in accordance with the classic definitions of communication complexity theory.

The motivation for this problem, as suggested by Conitzer and Sandholm, is that we are interested in reducing the communication burden – a burden that is expressed both by the amount of information the channel is required to communicate, and by the effort required by the individual voters to transmit their preferences. Another advantage is that in such a sequential protocol, voters don't always need to transmit all of their preferences (in order to calculate the winner), and so there is a higher degree of privacy.

In order to define the problem formally, we must first define the notion of a communication protocol. Most of the following definitions and methods are based on classical concepts in communication complexity theory, and can be found (in a somewhat more general version) in [11].

**Definition 2.1.1** (communication protocol, based on Definition 1.1 in [11])**.** *A* ***protocol*** $\mathcal{P}$ *over domain* $\mathcal{L}(X)^n$ *with range* $\mathcal{P}(X) \setminus \{\emptyset\}$ *is a binary tree where each node* $v$ *is labeled by a function* $f_{i,v} : \mathcal{L}(X) \to \{0, 1\}$ *for some* $i \in N$*, and each leaf is labeled with an element* $A \in \mathcal{P}(X) \setminus \{\emptyset\}$*.*[1]

*The* ***value*** *of the protocol* $\mathcal{P}$ *on the input* $P = \langle V_1, \ldots, V_n \rangle \in \mathcal{L}(X)^n$ *is the label of the leaf reached by starting from the root, and walking down the tree,*

[1] We note that while $\mathcal{P}$ here denotes a protocol, $\mathcal{P}(X)$ denotes the powerset of $X$. As the notations usually appear in different forms and contexts, there is little risk of confusion.

*when at each internal node $v$, labeled by $f_{i,v}$, we walk left if $f_{i,v}(V_i) = 0$ and right if $f_{i,v}(V_i) = 1$.*

*The **cost** of a protocol $\mathcal{P}$ is the height of the tree representing $\mathcal{P}$.*

*The protocol $\mathcal{P}$ **computes the voting rule** $\mathcal{F}$ if for any profile $P$, the value of $\mathcal{P}$ on $P$ is $\mathcal{F}(P)$.*

The idea of the definition is that the tree describes the way in which the voters communicate (recall that before any communication is done, voter $i$ only has knowledge of $V_i$): On a node $v$ labeled by $f_{i,v}$, the $i$th voter transmits $f_{i,v}(V_i)$ to all other voters. Then, according to the result $f_{i,v}(V_i)$ we iterate to one of the two children of $v$, say $u$ labeled by $f_{j,u}$, where voter $j$ will transmit $f_{j,u}(V_j)$ and so on.

We are now ready to define communication complexity:

**Definition 2.1.2.** *Let $\mathcal{P}(\mathcal{F})$ be the set of all protocols computing $\mathcal{F}$. The communication complexity of $\mathcal{F}$ is* $\min\{D : D \text{ is the cost of } \mathcal{P} \in \mathcal{P}(\mathcal{F})\}$.

In words, the (deterministic) communication complexity of a rule $\mathcal{F}$ is the number of bits sent in the worst case scenario of the best protocol for $\mathcal{F}$. Conitzer and Sandholm also defined a notion of nondeterministic communication complexity for voting rules, which we will not discuss here.

Let us inspect some general protocols for voting rules. The preferences of the voters are denoted, as usual, by the vector $\langle V_1, \ldots, V_n \rangle \in \mathcal{L}(X)^n$, and $V_i^j$ is the $j$th alternative in the preference order of voter $i$.

A protocol that will work for any voting rule $\mathcal{F}$ is one where the first voter submits $V_1^1$, then $V_1^2$, etc., until $V_1$ was submitted entirely. Then the second voter submits $V_2^1, V_2^2$ and so on, till all the voters have submitted their preferences. At that point, each voter knows the entire profile $P$, and so can calculate $\mathcal{F}(P)$.

We notice that the order in which the information is transmitted in not important, as long as it is fixed in advanced. We could just as well have had voter 1 submitting $V_1^m$, then voter 2 submitting $V_2^m$, sequentially, and when all voters have submitted their last preference, voter 1 submits $V_1^{m-1}$ and so on.

What is the cost of this protocol? Since there are $m$ candidates, submitting the name of one candidate required $\log m$ bits. Every voter submits the names of $m$ candidates, and there are $n$ voters, so in general the protocol requires $mn \log m$ bits, which is to say the cost of the protocol is $mn \log m$.

If follows from this that $mn \log m$ is an upper bound for the communication complexity of any rule. In our $O$ notation, the communication complexity of any rule is in $O(mn \log m)$. We comment that it is actually enough for every voter to send their top $m - 1$ preferences, as that uniquely determines the last remaining preference. The cost of such a protocol will be $(m-1)n \log m$, but as we are dealing with bounds, where the limit behavior is what interests us, we don't bother with that difference. The following example shows how that complexity can be greatly reduced for some voting rules.

**Example 2.1.3** (Plurality)**.** Take plurality, $\mathcal{F}_P$. If we know the top priorities of all the voters in a profile $P$, we can calculate $\mathcal{F}_P(P)$. Take the protocol

where voter 1 transmits $V_1^1$, then voter 2 submits $V_2^1$, and so on, till the last voter. According to what we said, this is enough to calculate $\mathcal{F}_P(P)$, so this is a protocol for $\mathcal{F}_P$. We had $n$ voters, each transmitting $\log m$ bits for their top rank. This means that the cost of the protocol is $n \log m$, so the compilation complexity of plurality is $O(n \log m)$, which is significantly less than the general bound we had before, $O(mn \log m)$.

The example above gives us the general method for calculating an upper bound for the communication complexity of a rule. We simply present an efficient protocol for the rule, and the cost of that protocol constitutes an upper bound. While this might seem simple in the case of plurality or in the general case, Conitzer and Sandholm have also given some more sophisticated protocols for a number of rules, which we will review later.

The method for finding a lower bound, however, is a bit more complicated, and requires some more definitions.

**Definition 2.1.4** (Fooling set)**.** *A **fooling set** for $\mathcal{F}$ is a set $\{P_1, \ldots, P_k\}$ of profiles, where $P_i = \langle V_{i,1}, \ldots, V_{i,n} \rangle$, which satisfies:*

- *There exists $x \in X$ such that exactly one of the following two holds:*
  1. *for every $1 \leq i \leq k$, $\mathcal{F}(P_i) = \{x\}$.*
  2. *for every $1 \leq i \leq k$, $\mathcal{F}(P_i) = \{y_i\}$ for some $y_i \in X, y_i \neq x$ (the winner can be different for different profiles).*
- *For any $P_i, P_j$ in the set there is some vector $\langle l_1, \ldots, l_n \rangle \in \{i, j\}^n$ such that, if we are in case 1, we have $\mathcal{F}(\langle V_{l_1,1}, \ldots, V_{l_n,n} \rangle) = \{z\}$ for some $z \in X, z \neq x$, and if we are in case 2 we have $\mathcal{F}(\langle V_{l_1,1}, \ldots, V_{l_n,n} \rangle) = \{x\}$.*

The second requirement simply means that when taking two profiles from the set, we can always mix their ballots in such a way that will give us a different winning set.

Standard communication complexity functions take values in $\{0, 1\}$, and so the fooling set method is defined for such functions. To overcome this, Conitzer and Sandholm adapted the definition for voting rules by having the fooling set to be not for the rule $\mathcal{F}(P)$, but rather a function $f(P, x)$ that checks if $x \in \mathcal{F}(P)$ or not.

The following lemma states the connection between a fooling set and the lower bounds on the communication complexity for voting rules (this is a special version used by Conitzer and Sandholm for the case of voting rules, where the general lemma – which can be found, for example, in [7] – is for any function).

**Lemma 2.1.5.** *If there exists a fooling set of size $k$ for a rule $\mathcal{F}$, then the compilation complexity of $\mathcal{F}$ is at least $\log k$.*

The idea behind the fooling set technique is that the voters cannot "know for sure" what is the result of $\mathcal{F}$ for their preferences, (at least) until they have

determined which of the elements of the fooling set represents their preferences.[2] In a slightly more formal way, the properties of the fooling set entail that any protocol tree computing $\mathcal{F}$ has to have at least $k$ different branches, so the height of the tree is at least $\log k$.

The difference between this version and the general one is that the general lemma relates to the function used in the definition of the fooling set, which for us is $f(P, x)$. The general lemma gives us a lower bound on the communication complexity of $f$. Since in all the profiles in our fooling set there is a single winner, we find that to calculate $\mathcal{F}(P)$ (to find the single winner) is at least as complicated as checking if $x$ is a winner or not, so the communication complexity of $f$ is a lower bound for the communication complexity of $\mathcal{F}$, and so we get the special lemma.

So the search for lower bounds consists of trying to find a large set of profiles which answer to the conditions of the fooling set. We give some examples of upper and lower bounds proved by Conitzer and Sandholm (we don't give the proofs, which can be found in [4], but we present the protocol for the upper bound).

**Example 2.1.6** (Plurality with runoff – see Appendix A)**.** The communication complexity of Plurality with runoff is $\Theta(n \log m)$, which is the same as that of the standard plurality rule. The protocol Conitzer and Sandholm give is the following: Let all voters submit their top candidate, like for plurality ($n \log m$ bits). Now all voters know who are the top two candidates, and they vote on them, sending 1 if they support the first of the two, 0 if they support the second (according to some prearranged ordering). The total cost is $n \log m + n$, which is $O(n \log m)$.

**Example 2.1.7** (Borda)**.** Borda is a positional scoring rule, and while those are considered relatively simple rules, Conitzer and Sandholm showed that Borda is $\Omega(mn \log m)$, and since that is the maximal communication complexity for any rule, we have that it is also $\Theta(mn \log m)$. Borda is especially interesting because, as we will see, it has a very different compilation complexity, comparing to the communication complexity. The same thing can be said for STV, which is next rule we look at.

**Example 2.1.8** (STV)**.** Conitzer and Sandholm showed that the communication complexity of STV is in $O(n(\log m)^2)$ and $\Omega(n \log m)$. We present a protocol for the rule, and the reader can verify that it is of cost $n(\log m)^2$.

Consider the following protocol: Each voter transmits their top candidate ($n \log m$). At this point all the voters can compute which of the candidates is removed (if no candidate has a majority). Let $x_i$ be the removed candidate. Now only the voters who ranked $x_i$ first transmit their second candidate, and so on. By calculating the maximal number of voters who have to resend their new top preference, we get that the total cost is $O(n(\log m)^2)$.

[2] The voters' preferences could be a profile that is not in the fooling set at all, but as it *might* be in the fooling set, this gives a lower bound on the communication complexity

**Example 2.1.9** (Bucklin)**.** One last interesting example is Bucklin (see Appendix A), whose communication complexity is $\Theta(mn)$. A naive protocol for Bucklin could have all voters transmitting first their top candidate ($n \log m$ bits). Then, if no candidate has a majority, all voters transmit their second most preferred candidate, and so on. However, the worst case scenario of such a protocol would have the voters transmitting all their top $\lfloor \frac{m}{2} \rfloor + 1$ candidates, and the upper bound we get is $O(nm \log m)$. Conitzer and Sandholm suggested a more efficient protocol:

Fix an ordering of the candidates, $x_1, \ldots, x_m$. Let $k$ be the Bucklin score of the candidate with the minimal score. We find $k$ in a logarithmic process: Every voter sends $m$ bits, the $i$th bit is 1 if $x_i$ is in the top $\frac{m}{2}$ candidates of that voter, 0 otherwise (w.l.o.g, $m = 2^l$ for some $l$). At this point all the voters can calculate if $k \leq \frac{m}{2}$ of not. If it is smaller, each candidate $i$ sends $\frac{m}{2}$ bits. If $x_{i_1}, \ldots, x_{i_{\frac{m}{2}}}$ are the top $\frac{m}{2}$ candidates of that voter (those are known to all voters), then now the $j$th bit is 1 if $x_{i_j}$ is in the top $\frac{m}{4}$ priorities, 0 otherwise. On the case where $k > \frac{m}{2}$, the voters do the same thing, but this time for the candidates in the bottom half of their ballots, indicating which of them is in the third quarter of the ballot. After every such step, the voters can again pinpoint the location of $k$ better (after the first step we know in which half $k$ is, after that we know in which quarter it is, and so on). When the process ends, the voters know both the value of $k$ and who the winning candidate is. The cost is $O(nm)$.

The complete results will be given in Chapter 3, when we compare them with compilation complexity. In the next section we review the compilation complexity of the rules above, which will show us how different these notions can be.

## 2.2 Compilation Complexity

Compilation complexity was first introduced by Chevaleyre et al. in 2009 [3], and an additional paper by Xia and Conitzer with some more results was published in 2010 [21]. While communication complexity is about efficient communication of preferences, it can be said that compilation complexity is about efficient storage of preferences. The idea of compilation complexity is that given some voting rule $\mathcal{F}$ and a profile $P$, we would like to store $P$ in an efficient manner with respect to $\mathcal{F}$, that is, to store only the information $\mathcal{F}$ actually uses in the profile.

One motivation that Chevaleyre et al. mention for finding the compilation complexity is that it has a practical use in elections which are performed on multiple districts. We could have, for example, a situation where one district has voted, and we want to submit the results to other districts, which haven't voted yet. From studying compilation complexity we can find a condensed form for the information, which would save us the need of sending all the ballots.

Another motivation, no less important, is that compilation complexity gives

some description of the rule which seems to be quite important. This will be further discussed in Chapter 4.

An example that might clarify the notion of compilation complexity is the plurality rule. Though $P$ contains the complete preferences of all the voters, we are only interested in knowing which candidate was ranked first by every voter (and as we shall see, we can settle for even less: how many voters ranked each candidate first).

To formalize this, of course, we need to somehow define formally the concept of "the information used by $\mathcal{F}$". The idea of Chevaleyre et al. was to assume that $P$ is a profile representing just part of our voters (a subelectorate), so we need to keep all the information required such that, given a profile $P'$ of the rest of the population (of an unknown size), we will be able to compute the winner based on the preferences of the entire population.[3] For this we must first define the union of profiles:

**Definition 2.2.1.** *Take any two profiles:*

$$P_1 = \langle V_1, \ldots, V_{n_1} \rangle \in \mathcal{L}(X)^{n_1},\ P_2 = \langle U_1, \ldots, U_{n_2} \rangle \in \mathcal{L}(X)^{n_2}$$

*We define* $P_1 \cup P_2 \in \mathcal{L}(X)^{n_1+n_2}$ *to be* $\langle V_1, \ldots, V_{n_1}, U_1, \ldots, U_{n_2} \rangle$.

Now we can define the formal notion of a compilation function – the function which stores the relevant information (in the definition below, the set $\{0,1\}^*$ is the set of all finite strings over the alphabet $0, 1$, and for $x \in \{0,1\}^*$ we define $|x|$ to be the length of the string $x$):

**Definition 2.2.2** (Compilation function)**.** *Given a voting rule* $\mathcal{F}$*, a function* $\sigma : \mathcal{L}(X)^n \to \{0,1\}^*$ *is a compilation function for* $\mathcal{F}$ *if there is a function* $\rho : \{0,1\}^* \times \mathcal{L}(X)^* \to \mathcal{P}(X) \setminus \{\emptyset\}$ *such that for any* $P \in \mathcal{L}(X)^n$ *and* $P' \in \mathcal{L}(X)^*$ *we have* $\rho(\sigma(P), P') = \mathcal{F}(P \cup P')$.

*For a compilation function* $\sigma$ *we define* $|\sigma| = \max_{P \in \mathcal{L}(X)^n} (|\sigma(P)|)$.

**Definition 2.2.3.** *Given a rule* $\mathcal{F}$*, let* $A$ *be the set of all compilation functions for* $\mathcal{F}$*. We define the compilation complexity of* $\mathcal{F}$ *as* $\min_{\sigma \in A}(|\sigma|)$.

Now, this could also be looked at from another perspective.

**Definition 2.2.4** (Profile equivalence)**.** *Given* $\mathcal{F}$*, we define an equivalence relation on* $\mathcal{L}(X)^n$ *in the following way:* $P \sim_{\mathcal{F}} P'$ *for* $P, P' \in \mathcal{L}(X)^n$ *if for every* $Q \in \mathcal{L}(X)^*$ *we have* $\mathcal{F}(P \cup Q) = \mathcal{F}(P' \cup Q)$.

*The number of equivalence classes generated in* $\mathcal{L}(X)^n$ *under this relation is denoted by* $g(n, m)$.

It is simple to verify that the relation above is indeed an equivalence relation. Chevaleyre et al. point out that the problem of finding the compilation

[3] Xia and Conitzer also inspected some other types of compilation complexity, where we also know the total size of the population, and also when the size of the subelectorate is unknown [21], but we will not discuss those complexities here.

complexity can be looked at as a one-round communication complexity problem, where we have two parties: $A$, representing the subelectorate of $n$ voters whose preferences are $P$, and $B$, the remaining electorate of an unknown size, whose preference are represented by $P'$. In a one-round communication protocol of this sort, $A$ wants to send one message to $B$ so that $B$ can compute $\mathcal{F}(P \cup P')$. The communication complexity is the length of the longest message in the best protocol computing $\mathcal{F}$. Chevaleyre et al. show, based on a [11], that:

**Proposition 2.2.5.** *The compilation complexity of a voting rule $\mathcal{F}$ is* $\lceil \log g(n,m) \rceil$

The proposition means that we just have to be able to describe in which equivalence class our profile is, which would require $\log g(n,m)$ bits.

As for calculating the bounds on the compilation complexity, there are no special "tricks" used usually. In order to find an upper bound, one usually has to give an upper bound for $g(n,m)$. While this is relatively simple for some rules, such as plurality, it can be more complicated for others.

The standard method for giving a lower bound is to find a set of profiles $Y \subseteq \mathcal{L}(X)^n$ whose size is simple to calculate, such that no two profiles in $Y$ are equivalent. That is, for every $P, P' \in Y$ there is a profile $Q \in \mathcal{L}(X)^*$ such that $\mathcal{F}(P \cup Q) \neq \mathcal{F}(P' \cup Q)$. This set $Y$ is the equivalent of compilation complexity for a fooling set.

As we did for communication complexity, we bring some examples from [3] and [21], not giving the proofs, but only explaining the idea of the compilation function giving the upper bound.

First, we note that the number of equivalence classes is at most the number of profiles, which is $|\mathcal{L}(X)^n| = m!^n$, so we get and upper bound of $\log(m!^n) = n\log(m!) \approx nm\log m - nm$, where we used Stirling's approximation for that last equality. So $O(nm\log m)$ is an upper bound for any rule (we note that this is the same as the maximal communication complexity).

Next, if the rule is anonymous, we see that any two profiles are equivalent if for any ballot $V$, the number of people who submitted $V$ is equal on both profiles. We can enumerate all the possible $m!$ ballots, $V_1, \ldots, V_{m!}$, and to associate a profile with a vector $\langle v_1, \ldots, v_{m!} \rangle$, where $v_i$ is the number of people who voted $V_i$ in that profile. This is a vector whose entries are non-negative integers, and their sum is $n$ (the entries have to add up to the number of voters). The total number of vectors satisfying those constraints is given by the number of combinations of $n$ elements from a set size $n+m!-1$, which is $\binom{n+m!-1}{n}$. We get that $g(m,n) \leq \binom{n+m!-1}{n}$, so the compilation complexity of any anonymous rule has an upper bound of $\log \binom{n+m!-1}{n} \approx n\log(1+\frac{m!}{n}) + m!\log(1+\frac{n}{m!})$ (where we again used Stirling's approximation and some algebraic manipulation).

Now we look at the rules whose communication complexity we saw before.

**Example 2.2.6** (Plurality)**.** Plurality gives a score to any candidate, which can be at most $n$, and it is simple to show that any two profiles are equivalent iff the score of every candidate is the same on both profiles. This means that we use a method similar to what we had for anonymous rules: We associate a profile with a vector $\langle v_1, \ldots, v_m \rangle$, where $v_i$ is the score of the $i$th candidate (this can

be done for any positional scoring rule). The number of such vectors satisfies $g(m,n) = \binom{n+m-1}{n}$ (similarly to the anonymous case), and the complexity is in $\Theta(n\log(1+\frac{m}{n}) + m\log(1+\frac{n}{m}))$.

**Example 2.2.7** (Plurality with runoff)**.** We recall that for communication complexity, there was no difference between plurality and plurality with runoff. For compilation complexity, however, that is not the case. Xia and Conitzer showed that plurality with runoff is in $\Theta(m^2 \log n)$ [21]. The upper bound is achieved by taking a compilation function that stores for a profile $P$ $(a)$ the number of people who voted for each candidate ($m \log n$ bits), and $(b)$ for any pair of candidates $x, y$ it stores $N(x, y, P)$ ($m^2 \log n$ bits). As the first element is smaller than the second, we get that the upper bound is given by $O(m^2 \log n)$.

**Example 2.2.8** (Borda)**.** We recall that Borda had the maximal possible communication complexity: $mn \log m$. For compilation terms, on the other hand, we know that two profiles are equivalent iff all candidates have the same Borda score (positional scoring rule), so we identify a profile with $\langle v_1, \ldots, v_m \rangle$, $v_i$ being the Borda score of the $i$th candidate. The score is at most $nm$, and so we get an upper bound $m \log(mn)$. This is also a lower bound, as shown by Chevaleyre et al. [3] and Xia and Conitzer [21], so Borda is $\Theta(m \log(mn))$.

**Example 2.2.9** (STV)**.** For a profile $P$, subset $Z \subset X$ and $x \in X \setminus Z$, define $ntop(P, Z, x)$ to be the number of voters in $P$ who prefer $x$ to any other candidate in $X \setminus Z$. Define $above(P, Z, x)$ to be the number of voters in $P$ who prefer all the candidates in $Z$ to $x$, and prefer $x$ to any other candidate in $X \setminus Z$. Chevaleyre et al. showed that two profiles $P, P'$ are equivalent, with respect to STV, iff $ntop(P, Z, x) = ntop(P', Z, x)$ for any $Z \subset X, x \in X \setminus Z$ [3] iff $above(P, Z, x) = above(P', Z, x)$ for any $Z \subset X, x \in X \setminus Z$ [12]. This means that $g(m, n)$ is the number of different ways of defining the $above(\cdot, \cdot, \cdot)$ function. For a fixed candidate $x \in X$, we can get from each such combination a vector $\langle v_1, \ldots, v_{2^{m-1}} \rangle$, $v_i$ being $above(P, Z_i, x)$, where $Z_i$ is the $i$th subset of $X$ not containing $x$ (according to some enumeration). The sum of the elements must add up to $n$, and so the total number of combinations (for a fixed candidate) is $\log \binom{n+2^{m-1}-1}{n}$. There are at least as many combinations for all the candidates as there are for a single candidate, and if we assume those combinations are independent for all candidates (which is not the case), we get that the number of combinations is at most $\binom{n+2^{m-1}-1}{n}^m$, so the bounds that Chevaleyre et al. found were

$$\Omega(n\log(1+\frac{2^{m-1}}{n}) + 2^{m-1}\log(1+\frac{n}{2^{m-1}}))$$

and

$$O(m\left(n\log(1+\frac{2^{m-1}}{n}) + 2^{m-1}\log(1+\frac{n}{2^{m-1}})\right))$$

We can see that even the lower bound gives a high complexity, when comparing to the relatively low communication complexity of STV. In Chapter 4 we give a somewhat different description of the *above* functions, and use it in Appendix C to derive another upper bound.

**Example 2.2.10** (Bucklin)**.** Last, for Bucklin, Xia and Conitzer showed it is enough to have $N(x, i, P)$ for any $x$ and $i$, in order to calculate Bucklin score for every candidate, and the winners. This means we can associate a profile with $m$ vectors, $\langle v_1^x, \ldots, x_m^x \rangle$ for any $x \in X$, where $v_i^x = N(x, i, P)$. For each such vector, the sum of the elements is $n$, which gives us at most $\binom{n+m-1}{m}^m$ different combinations, and the compilation complexity is in $mn \log(1 + \frac{m}{n}) + m^2 \log(1 + \frac{n}{m})$. They also showed that Bucklin is in $m^2 \log(1 + \frac{n}{m})$, which gives a relatively tight bound (when $n$ becomes very large, we have $mn \log(1 + \frac{m}{n}) \approx m$, and so the other term is more dominant).

In the next chapter we compare the compilation complexity results with those of communication complexity, connecting them through the notion of informational size, and in Chapter 4 we will analyze more deeply some of the representations we have encountered here, and discuss the connections between compilation functions of different rules.

# Chapter 3

# Informational Size

Informational size of voting rules was introduced by Sato in 2008 in [14], where he defined the notion of informational size of a rule and used it to characterize voting rules axiomatically. The idea of informational size is somewhat similar to that of compilation complexity, which we saw that we can take to be the number of equivalence classes of profiles, only this notion is about equivalence between ballots and not between profiles.

In the first section we define informational size formally, give some examples, and present Sato's results. In the second section we show how informational size can be used as an upper bound for both compilation complexity and communication complexity, thus linking both using this notion, and in the last section we present a detailed list of results from the literature for both the complexities and compare the two along with informational size.

## 3.1 Informational Size

We start with the basic notion on which informational size is based: ballot equivalence.

**Definition 3.1.1.** *We say that two ballots $V, V'$ are equivalent for voter $i$, $V \sim_i V'$, given a voting rule $\mathcal{F}_{m,n}$, if for any profile $\langle V_1, \ldots, V_{i-1}, V_{i+1}, \ldots, V_n\rangle \in \mathcal{L}(X)^{n-1}$ of $n-1$ voters we have*

$$\mathcal{F}_{m,n}(\langle V_1, \ldots, V_{i-1}, V, V_{i+1}, \ldots, V_n\rangle) = \mathcal{F}_{m,n}(\langle V_1, \ldots, V_{i-1}, V', V_{i+1}, \ldots, V_n\rangle)$$

This means that $V \sim_i V'$ with respect to $\mathcal{F}_{m,n}$, if the voter doesn't mind if they submit $V$ or $V'$, since the rule identifies the two ballots in some sense. We give some examples of this equivalence:

**Example 3.1.2** (Plurality)**.** We claim that under the plurality rule, two ballots $V, U$ satisfy $V \sim_i U$ for voter $i$ iff $V^1 = U^1$. The "if" part is simple, since, as we've seen before, plurality only cares about the number of times each candidate gets elected first, so the order of the candidates ranked from 2 to $m$ is not important.

To see the "only if" direction, take some profile $V'$ such that $V'^1 \neq V^1$. Take $n-1$ ballots $V_1, \ldots, V_{i-1}, V_{i+1}, \ldots, V_n$ such that $V_j^1 = V^1$ for exactly $\lfloor \frac{n-1}{2} \rfloor$ of them, and $V_j^1 = V'^1$ for all the rest ($1 \leq j \leq n, j \neq i$). We get that $V^1 \in \mathcal{F}_P(\langle V_1, \ldots, V_{i-1}, V, V_{i+1}, \ldots, V_n\rangle), V^1 \notin \mathcal{F}_P(\langle V_1, \ldots, V_{i-1}, V', V_{i+1}, \ldots, V_n\rangle)$, so according to the definition above the two are not equivalent.

**Example 3.1.3** (Dictatorial rules)**.** Another example is a dictatorial rule. Take $\mathcal{F}_{m,n}$ which always chooses the alternative ranked first by voter 1. We get that for voter 1 the equivalence relation $V \sim_1 V'$ is just like we saw above for plurality, but for any voter $i \neq 1$ it holds that $V \sim_i V'$ for any $V, V'$, simply because it doesn't matter what they vote.

It is a simple proof to show that $\sim_i$ is an equivalence relation, and from that we get a partition of $\mathcal{L}(X)$ into equivalence classes (i.e., a set of non empty pairwise-disjoint subsets of $\mathcal{L}(X)$, whose union is $\mathcal{L}(X)$). This brings us to the definition of informational size:

**Definition 3.1.4** (Informational size)**.** *Given a voting rule $\mathcal{F}_{m,n}$, denote the set of equivalence classes generated by $\sim_i$ by $\mathcal{M}_{i,\mathcal{F}_{m,n}}$. We call $\mathcal{M}_{i,\mathcal{F}_{m,n}}$ the partition induced by $\mathcal{F}_{m,n}$ with respect to $i$. We say that the* ***informational size of*** *$\mathcal{F}_{m,n}$ is $\sum_{1 \leq i \leq n} |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$ (whenever $\mathcal{F}_{m,n}$ or $i$ are clear from the context, we will omit the use of the subscript).*

*A rule $\mathcal{F}$ in a class of rules $S$ is said to* ***operate on minimal informational requirements in*** *$S$ if there is no other rule in $S$ with a smaller informational size.*

Looking back at the previous examples, we find that the informational size of plurality, for $m$ alternatives and $n$ voters is $mn$ (for each voter $i$, $|\mathcal{M}_{i,\mathcal{F}_P}| = m$, so the total sum is $mn$). For the a dictatorial rule we get informational size of $m + n - 1$: The first voter has $m$ equivalence classes, and every other voter has only one equivalence class. We will present some more examples of informational size for additional rules in the last section of this chapter, compared with compilation complexity and communication complexity.

Sato used the notion to characterize rules axiomatically. Let NA be the set of neutral and anonymous voting rules, NAM the subset of NA which is also monotone, and NAMP the subset of NAM which is also Pareto efficient. In his work, constant rules (i.e., rules that produce the same result for all profiles) are not taken into account. The main theorems he presented in his paper are:

**Theorem 3.1.5** (Theorem 3.1 in [14])**.** *If an NA-voting rule $\mathcal{F}_{m,n}$ operates on minimal informational requirements in NA,*[1] *then*

- *$\mathcal{F}_{m,n}$ has informational size $mn$.*
- *There exists $1 \leq r \leq m$ such that for any voter $i$ and two ballots $V_1, V_2$, $V_1 \sim_i V_2$ iff $V_1^r = V_2^r$.*

**Theorem 3.1.6** (Theorem 3.2 in [14])**.** *If an NAM-voting rule $\mathcal{F}_{m,n}$ operates on minimal informational requirements in NAM, then*

- *The $r$ from Theorem 3.1.5 is either 1 or $m$, and*

[1] Due to a small bug in the proof, the theorem actually applies to a slightly smaller set of rules [16]. See more about this in Chapter 5, where we prove the original theorem with one exception.

- *If $r = 1$ then for any profile $P$ we have $\mathcal{F}_{m,n}(P) \supseteq \mathcal{F}_P(P)$ (every plurality winner is a winner in $\mathcal{F}_{m,n}$), and if $r = m$ then for any profile $P$ we have $\mathcal{F}_{m,n}(P) \supseteq \mathcal{F}_V(P)$ (every veto winner is a winner in $\mathcal{F}_{m,n}$).*

**Theorem 3.1.7** (Theorem 3.3 in [14])**.** *For every NAMP-voting rule $\mathcal{F}_{m,n}$ operating on minimal informational requirements in NAMP, the $r$ from Theorem 3.1.5 is 1 and for any profile $P$ we have $\mathcal{F}_{m,n}(P) \supseteq \mathcal{F}_P(P)$ (every plurality winner is a winner in $\mathcal{F}_{m,n}$).*

The partitions induced by the voting rules will play a bigger role in Chapter 5, but for now we just focus on the measure itself given by informational size. In the next section we explain how to relate that to communication and compilation complexity.

## 3.2 Common Upper Bound for Complexity Measures

In this section we define a measure of complexity, based Sato's informational size and the partition into equivalence classes, which will constitute an upper bound for both compilation complexity and communication complexity. Given $\mathcal{F}_{m,n}$ and preferences of voters $V_1, \ldots, V_n$, we notice that in order to compute $\mathcal{F}_{m,n}(\langle V_1, \ldots, V_n \rangle)$ we don't really need to know the ballots submitted by each voter, but rather it is enough to know for every $V_i$ to which equivalence class of $\mathcal{M}_{i,\mathcal{F}_{m,n}}$ $V_i$ it belongs. We formalize this claim:

**Proposition 3.2.1.** *Given a voting rule $\mathcal{F}_{m,n}$, some number $0 \leq k \leq n$, and ballots $V_1, \ldots, V_k, V_1', \ldots, V_k', U_{k+1}, \ldots, U_n$ such that $V_i \sim_i V_i'$ for every $1 \leq i \leq k$, we have:*

$$\mathcal{F}_{m,n}(\langle V_1, \ldots, V_k, U_{k+1}, \ldots, U_n \rangle) = \mathcal{F}_{m,n}(\langle V_1', \ldots, V_k', U_{k+1}, \ldots, U_n \rangle)$$

*Proof.* We prove the claim by induction on $k$. For $k = 0$ the proposition comes down to $\mathcal{F}_{m,n}(\langle U_1, \ldots, U_n \rangle) = \mathcal{F}_{m,n}(\langle U_1, \ldots, U_n \rangle)$, which is clear. We prove the induction step.

We want to show

$$\mathcal{F}_{m,n}(\langle V_1, \ldots, V_{k+1}, U_{k+2}, \ldots, U_n \rangle) = \mathcal{F}_{m,n}(\langle V_1', \ldots, V_{k+1}', U_{k+2}, \ldots, U_n \rangle)$$

where $V_i \sim_i V_i'$ for any $1 \leq i \leq k+1$. We know that for voter $k+1$ is holds that $V_{k+1} \sim_{k+1} V_{k+1}'$, so

$$\mathcal{F}_{m,n}(\langle V_1, \ldots, V_k, V_{k+1}, U_{k+2}, \ldots, U_n \rangle) = \mathcal{F}_{m,n}(\langle V_1, \ldots, V_k, V_{k+1}', U_{k+2}, \ldots, U_n \rangle)$$

and then we can use the induction hypothesis to get

$$\mathcal{F}_{m,n}(\langle V_1, \ldots, V_k, V_{k+1}', U_{k+2}, \ldots, U_n \rangle) = \mathcal{F}_{m,n}(\langle V_1', \ldots, V_k', V_{k+1}', U_{k+2}, \ldots, U_n \rangle)$$

and from that follows the claim. □

We notice that by taking $k = n$, the proposition implies that for any two profiles $P, P'$ where the $i$th ballots are equivalent with respect to voter $i$, we have $\mathcal{F}_{m,n}(P) = \mathcal{F}_{m,n}(P')$. Informally, the proposition says that as far as the voting rule is concerned, the voters don't choose ballots but rather equivalence classes of ballots. We show how this fact helps us associate informational size with both compilation complexity and communication complexity.

We start with the compilation complexity:

**Proposition 3.2.2.** *Given a voting rule $\mathcal{F}_{m,n}$ inducing the partitions $\mathcal{M}_{i,\mathcal{F}_{m,n}}$, the compilation complexity of $\mathcal{F}_{m,n}$ is less than or equal to* $\log(\prod_{1\leq i\leq n} |\mathcal{M}_{i,\mathcal{F}_{m,n}}|)$

*Proof.* We know we can identify compilation complexity with $g(n, m)$, the number of equivalence classes of profiles. We can use Proposition 3.2.1 to get a bound on the number of equivalence classes of profiles: We take our subelectorate in compilation complexity to be the first $k$ ballots, and the rest $m - k$ ballots are the rest of the population.[2]

We get that if two profiles $P = \langle V_1, \ldots, V_k\rangle, P' = \langle V'_1, \ldots, V'_k\rangle$ satisfy that $V_i \sim_i V'_i$, then adding to them the profile $Q = \langle U_{k+1}, \ldots, U_n\rangle$ will always give the same result, which is the condition of compilation complexity. We get that for $\mathcal{F}_{m,n}$ we have $g(n, m) \leq \prod_{1\leq i\leq n} |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$. From this we get that the compilation complexity of $\mathcal{F}_{m,n}$ is less than or equal $\log(\prod_{1\leq i\leq n} |\mathcal{M}_{i,\mathcal{F}_{m,n}}|)$. □

Now we consider communication complexity:

**Proposition 3.2.3.** *Given a voting rule $\mathcal{F}_{m,n}$ inducing the partitions $\mathcal{M}_{i,\mathcal{F}_{m,n}}$, the communication complexity of $\mathcal{F}_{m,n}$ is less than or equal to* $\sum_{1\leq i\leq n} \log |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$

*Proof.* We suggest a protocol for computing $\mathcal{F}_{m,n}$ with cost $\sum_{1\leq i\leq n} \log |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$. Also from Proposition 3.2.1, we find that it is enough for every voter to submit to the other voters the equivalence class of his ballot. In the end of such a protocol, all voters know what equivalence class was chosen by every voter, and so they can calculate the result. Since voter $i$ has $|\mathcal{M}_{i,\mathcal{F}_{m,n}}|$ equivalence classes, to transmit the selected equivalence class will cost him $\log |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$, and so the total cost of the protocol will be $\sum_{1\leq i\leq n} \log |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$. □

We notice that since $\log a + \log b = \log(ab)$, the two bounds we got are in fact identical, so

$$\log(\prod_{1\leq i\leq n} |\mathcal{M}_{i,\mathcal{F}_{m,n}}|) = \sum_{1\leq i\leq n} \log |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$$

[2]There is a small technicality here, since compilation complexity is defined for rules that work on any profile (on domain $\mathcal{L}(X)^*$), while communication complexity and informational size can be defined for rules of only a finite number of voters. To completely formalize this difference we have to switch to the subset of rules defined for any number of voters, and to adapt the definition of informational size so that two ballots are equivalent iff they do not change the results of any profile for any number of voters. In such a case Proposition 3.2.1 can be extended for any number of ballots $U_1, \ldots, U_{n'}$.

is an upper bound for both compilation complexity and communication complexity. In some sense we can also add that this number does represent "information" better than the simple summation. Consider for example a voter $i$ for which all ballots are equivalent (as for the non-dictators in the dictatorial rule). For such a voter we have $|\mathcal{M}_{i,\mathcal{F}_{m,n}}| = 1$, and $\log |\mathcal{M}_{i,\mathcal{F}_{m,n}}| = 0$, which better describes the fact that there is no information given by the ballot of this voter. This measurement gives the constant rule, for example, informational size 0 (whereas the summation gives us $m$).

The question we would like to ask ourselves next is how tight is our bound?

For communication complexity, we can give some rules whose complexity is exactly that given by this bound. The maximal possible number of equivalence classes per voter is $m!$, and that is when no two ballots are equivalent. Let $\mathcal{F}_{m,n}$ be a rule for which no two ballots are equivalent (for any voter). When taking the logarithm, and using Stirling's approximation, we get:

$$\sum_{1\leq i\leq n} \log |\mathcal{M}_{i,\mathcal{F}_{m,n}}| = \sum_{1\leq i\leq n} \log(m!) = n\log(m!) \approx nm\log m - nm \in O(nm\log m)$$

One example for such a rule is Borda. It is easy to see that for a large enough $n$, given two different ballots $V, V'$, there is a candidate $x \in X$ who is ranked higher in $V$ relative to $V'$, and so we can come up with a profile $P = \langle V_1, \ldots, V_n\rangle$ such that $x$ wins with $P$ when we take $V_1 = V$ but loses when $V_1 = V'$.

In addition, we also know that Borda's communication complexity is $nm\log m$ [3, 21], and so we get that this bound is tight for communication complexity of some rules.

When we consider compilation complexity, we see that we can make our calculation a bit more accurate. We present a lemma proved by Sato:

**Fact 3.2.4** (Lemma 4.1 in [14])**.** *For any anonymous voting rule $\mathcal{F}$, $V \sim_i V'$ iff $V \sim_j V'$ for any $1 \leq i, j \leq n$.*

This means that for any two voters $i, j$ we have $\mathcal{M}_{i,\mathcal{F}} = \mathcal{M}_{j,\mathcal{F}}$, i.e., the partition of $\mathcal{L}(X)$ induced by $\mathcal{F}$ is identical for all voters. In this case we will simply talk about "the partition induced by $\mathcal{F}$", denoted $\mathcal{M}_{\mathcal{F}}$.

Given an anonymous voting rule $\mathcal{F}_{m,n}$, we know there is some integer $l$ such that $l = |\mathcal{M}_{i,\mathcal{F}_{m,n}}|$ for any $i$, so the bound we have is always $n\log l$. But we can do even better. For such a rule two profiles $P = \langle V_1, \ldots, V_n\rangle, P' = \langle V'_1, \ldots, V'_n\rangle$ are equivalent not just when $V_i \sim V'_i$ (since $\sim_i \equiv \sim_j$, we simply write $\sim$), but also whenever there is some permutation $\sigma$ of $\{1, \ldots, n\}$ such that $V_i \sim V'_{\sigma(i)}$. It follows that the important question is not which equivalence class was chosen by each voter, but rather how many voters voted for each equivalence class. Formally speaking, if $P$ is a profile and $M_1$ is an equivalence class, denote by $n(P, M_1)$ the number of voters who submitted a ballot in $M_1$, and by $n(P_{\leq k}, M_1)$ the number of voters among the first $k$ voters who submitted a ballot in $M_1$ ($k \leq n$). The following claim is an extension of Proposition 3.2.1 for anonymous rules.

**Proposition 3.2.5.** *Given an anonymous voting rule $\mathcal{F}_{m,n}$ which induces the partition $\mathcal{M}_{\mathcal{F}_{m,n}} = \{M_1, \ldots, M_l\}$, some number $0 \leq k \leq n$, and ballots $V_1, \ldots, V_k$, $V'_1, \ldots, V'_k, U_{k+1}, \ldots, U_n$ such that $n(P_{\leq k}, M_j) = n(P'_{\leq k}, M_j)$ for every $1 \leq j \leq l$, we have:*

$$\mathcal{F}_{m,n}(\langle V_1, \ldots, V_k, U_{k+1}, \ldots, U_n\rangle) = \mathcal{F}_{m,n}(\langle V'_1, \ldots, V'_k, U_{k+1}, \ldots, U_n\rangle)$$

*Proof.* The idea is to rearrange the ballots of the first $k$ voters so that we satisfy the conditions of Proposition 3.2.1. Let $\sigma$ be a permutation on the voters which fixes $k+1, \ldots, n$, and acts on the first $k$ voters by placing first all the $n(P_{\leq k}, M_1)$ ballots in $V_1, \ldots, V_k$ which are in $M_1$, then all the $n(P_{\leq k}, M_2)$ ballots in $V_1, \ldots, V_k$ of $M_2$, and so on. Let $\rho$ be a permutation that does the same thing for $P'$. The reader can verify that $V_i \sim V'_{\rho^{-1}\sigma(i)}$, and so (from Proposition 3.2.1)

$$\mathcal{F}_{m,n}(\langle V_1, \ldots, V_k, U_{k+1}, \ldots, U_n\rangle) =$$
$$\mathcal{F}_{m,n}(\langle V'_{\rho^{-1}\sigma(1)}, \ldots, V'_{\rho^{-1}\sigma(k)}, U_{k+1}, \ldots, U_n\rangle) =$$
$$\mathcal{F}_{m,n}(\langle V'_{\rho^{-1}\sigma(1)}, \ldots, V'_{\rho^{-1}\sigma(k)}, U_{\rho^{-1}\sigma(k+1)}, \ldots, U_{\rho^{-1}\sigma(n)}\rangle) =$$
$$\mathcal{F}_{m,n}(\langle V'_1, \ldots, V'_k, U_{k+1}, \ldots, U_n\rangle)$$

where the last equality follows from the anonymity of $\mathcal{F}_{m,n}$. □

Given the proposition, we can give the following bound for compilation complexity, based on the informational size and the partitions of ballots:

**Proposition 3.2.6.** *Let $\mathcal{F}_{m,n}$ be an A-voting rule which induces a partition into $l$ equivalence classes, then the compilation complexity of $\mathcal{F}_{m,n}$ is less than or equal to*

$$O(l \log(1 + \frac{n}{l}) + n \log(1 + \frac{l}{n}))$$

*Proof.* Based on Proposition 3.2.5, we can just count the number of votes in every equivalence class. Such a vector is an element of the set $\{\langle v_1, \ldots, v_l\rangle : v_i \geq 0, \sum_{1 \leq i \leq l} v_i = n\}$ (we identify $v_i$ with the number of agents who voted for equivalence class $M_i$. the number of voters must add up to $n$).

The size of that set is given by the number of combinations of size $l-1$ of a set of size $n+l-1$, which is $\binom{n+l-1}{l-1} = \binom{n+l-1}{n}$. By taking the logarithm of this expression, using Stirling's approximation and manipulating the equations, neglecting constants and elements of smaller orders, we get:

$$O(l \log(1 + \tfrac{n}{l}) + n \log(1 + \tfrac{l}{n}))$$

□

We know that for the case of plurality $l = m$, that is, we have $m$ equivalence classes in the partition induced by the rule. We get that the upper bound given by informational size is

$$O(m \log(1 + \frac{n}{m}) + n \log(1 + \frac{m}{n}))$$

which is precisely the compilation complexity of plurality, as was shown by Chevaleyre et al. so this bound is in fact tight for some rules.

In the next section we compare the three different measures of complexity we have.

## 3.3 Communication Complexity vs. Compilation Complexity

First, we note that when the informational size is maximal for an anonymous rule (i.e., no two ballots are equivalent), the number of equivalence classes is $l = m!$ and the compilation complexity we get from informational size is:

$$m! \log(1 + \frac{n}{m!}) + n \log(1 + \frac{m!}{n})$$

which again coincides with the maximal compilation complexity for any anonymous rule, as given by Chevaleyre et al. We denote this maximal bound by $MA_{IS}$, and we denote by $M_{IS}$ the bound we get from informational size for any rule (which we saw is approximately $nm \log m$). We denote by $M_{CMPL}$ and $MA_{CMPL}$ the maximal compilation complexity for any rule and for anonymous rules, respectively, and $M_{CMNC}$ denotes the maximal communication complexity (where anonymity in general does not give a lower upper bound).

Among the different rules we have seen, or for which we have results in communication or compilation complexity, the only rules with informational size less than $MA_{IS}$ are plurality, which we mentioned in the previous section, $k$-Approval, for which the number of equivalence classes $l$ is $\binom{m}{k}$ (every ballot is characterized by the $k$ approved candidates, which is choosing a combination of size $k$ from a set size $m$) and Bucklin, where the number of equivalence classes is $\frac{m!}{\lceil \frac{m}{2} - 1 \rceil !}$ (the highest possible Bucklin score for the Bucklin winner is $\lfloor \frac{m}{2} \rfloor + 1$, so all ballots that agree on the first $\lfloor \frac{m}{2} \rfloor + 1$ candidates are equivalent).

The three figures below show the known compilation complexity result (Figure 3.1) and communication complexity results (Figure 3.2) from the literature, along with the bound we get from informational size for a number of rules (Figure 3.3).

Though these diagrams are a bit hard to compare, we point out some facts that help understand the differences between compilation and communication complexity.

Though they don't measure the same things precisely, we know that all maximal complexities $M_{CMPL}, M_{IS}, M_{CMNC}$ are in $\Theta(mn \log m)$, and that both $MA_{CMPL}, MA_{IS}$ are in $\Theta(m! \log(1 + \frac{n}{m!}) + n \log(1 + \frac{m!}{n}))$. But besides that, as we have seen, several rules behave differently under compilation and communication complexity. Borda has a relatively low compilation complexity but high communication complexity, while STV behaves exactly the opposite.

Another set of rules worth mentioning is WMG. The characterization of the set implies that its compilation complexity is bounded from above by $m^2 \log n$,

which is the number of bits required for storing $N(x, y, P)$ for all $x, y$. The lower bound is identical to that, under some assumptions we will not go into [3, 21].

The interesting thing about these rules is that while they are all quite similar, compilation complexity-wise, some behave very differently when it comes to communication complexity. This also tells us that there is a major difference in what is measured by the two complexities. We also recall that in the introduction we saw that the computational complexity of some of those rules (namely Kemeny and Copeland) is very different. We go back to the connection with computational complexity in Chapter 4.

In the next chapter we will try to show how compilation and communication complexity form two complementary sides of voting rules, each relating to another aspect of the rule.

Regarding informational size, we see that the bound we get from it is not "rich" enough to see differences between many rules, and so almost all common rules are characterized by $MA_{IS}$. This is, at least in some sense, a result of informational size giving an upper bound on both compilation and communication complexity. In the discussion we raise some questions about the quality of the bound given by informational size, in light of the coarse hierarchy it gives us.

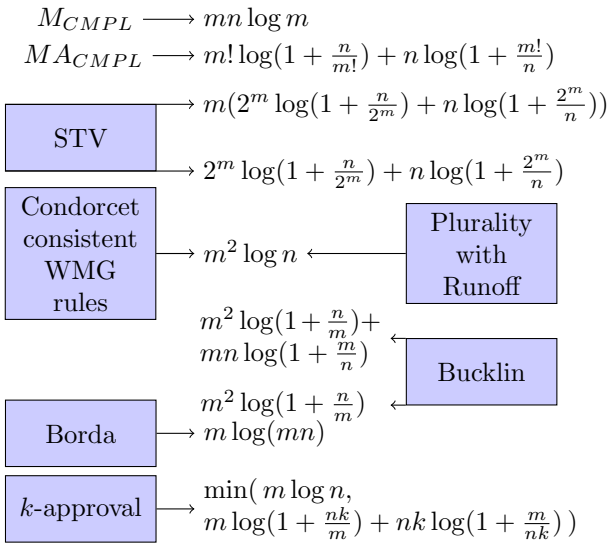


Figure 3.1: Results for compilation complexity (based on [3, 21]). An arrow coming from the top of an element stands for an upper bound, from the bottom for a lower bound, and from the middle for both.

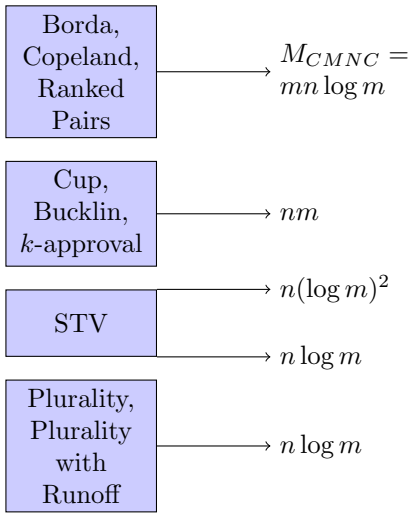


Figure 3.2: Results for communication complexity (based on [4]). An arrow coming from the top of an element stands for an upper bound, from the bottom for a lower bound, and from the middle for both.

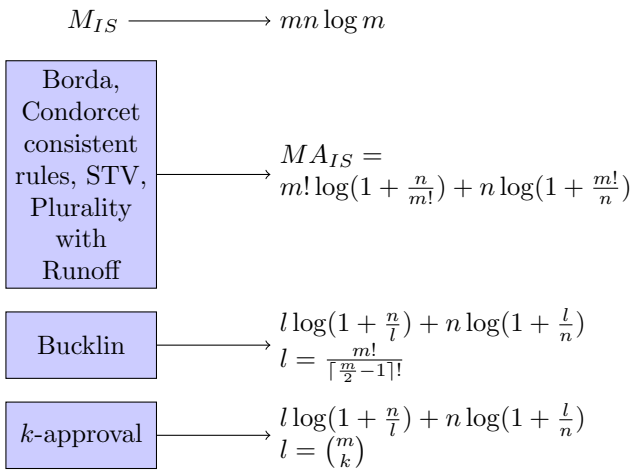


Figure 3.3: The bound given by informational size for various common rules. An arrow coming from the middle of an element stands for an upper and lower bound.

# Chapter 4

# The Problem of Representation

In the beginning of this thesis, we mentioned that we are interested in a more "meaningful" representation of voting rules, that will give us more information about the mechanism of the rules. As we have seen by now, voting rules come in all shapes and sizes. On the one hand, we can define a "random" rule, $\mathcal{F}_r$, which assigns for any profile $P$ some non empty $A_P \subseteq X$, which is determined in an arbitrary way. For such a rule we can never have less than maximal communication or compilation complexity, as not knowing the exact profile would prevent us from knowing what the winning set is. On the other hand, a constant rule, which always chooses the same winning set, represents the other side of the scale, where we don't need to store or transfer any information, and so both the communication and compilation complexity will always be 0.

It is true that by placing axioms on our rules, we can narrow the scale a bit (for example, we saw how the maximal compilation complexity decreases, as soon as we restrict ourselves to the set of anonymous rules). However, it seems like we cannot come up with a unique representation that will be more meaningful than the functional description of rules we are currently using, while capturing enough of the rules we want to address (though we challenge this claim later in the chapter, when we introduce generalized scoring rules).

Another way to go at this might be to study different representations for different rules, and the relations between those representations themselves. That is the approach we take in the following two sections. But for this we must be more clear on what we mean by "meaningful representations".

## 4.1 Defining Representations

We start by defining a representation of preferences. The idea is that our most basic representation is profiles, but given a voting rule, this representation could become somewhat redundant, in the sense that some profiles are equivalent under the rule (in the way we saw for compilation complexity). We would like to consider other structures as representations if they identify equivalent ballots for various rules. A representation will therefore be a sort of a quotient set of the profiles under an equivalence relation induced by some voting rule. We will later see how that gives us a hierarchy of these representations.

**Definition 4.1.1.** *Let $S = \{S_n\}_{n\in\mathbb{N}}$ be a family of sets. We call $S$ a **representation of preferences** if we have functions $f_n^S : \mathcal{L}(X)^n \to S_n$ which send each $n$ voter profile to an element of $S_n$, and functions $g_n^S : S_n \to \mathcal{L}(X)^n$, sending elements of $S_n$ to $n$ voter profiles, which also satisfy $f_n^S(g_n^S(s)) = s$ for any $s \in S_n$.*

Notice the definition implies that $g_n^S$ is one-to-one and $f_n^S$ is onto. The reader might be reminded of the definition of a compilation function. Indeed, $f_n^S$ is very similar to a compilation function $\sigma$, but instead of defining $\rho$ over $S_n \times \mathcal{L}(X)^*$, we defined the functions $g_n^S$ which send us back to the profile space.

The similarity between $f_n^S$ and the compilation function $\sigma$ can be made even more clear if we tweak the definition of compilation function a bit. The original definition was for $\sigma : \mathcal{L}(X)^n \to \{0,1\}^*$, with a function $\rho : \{0,1\}^* \times \mathcal{L}(X)^* \to \mathcal{P}(X) \setminus \{\emptyset\}$ satisfying $\rho(\sigma(P), Q) = \mathcal{F}(P \cup Q)$.

We can define the compilation function to be not one function, but a family of functions $\{\sigma_n\}_{n\in\mathbb{N}}$ for which there is a function $\rho : \{0,1\}^* \times \{0,1\}^* \to \mathcal{P}(X) \setminus \{\emptyset\}$ satisfying $\rho(\sigma_n(P), \sigma_{n'}(Q)) = \mathcal{F}(P \cup Q)$ for any two profiles $P, Q$ for $n, n'$ voters. One can verify that Proposition 2.2.5 still holds under this definition, and so do all the previous results for compilation complexity. Under this view, the condition $\rho(\sigma_n(P), \sigma_{n'}(Q)) = \mathcal{F}(P \cup Q)$ can be interpreted as a form of commutativity between $\mathcal{F}$ acting on the union of profiles and $\rho$ acting on $\sigma$-images of profiles.

Now it is clear that $f_n^S$ can be identified with $\sigma_n$, where $S_n = \{\sigma_n(P) : P \in \mathcal{L}(X)^n\} \subset \{0,1\}^*$, and $\rho(a,b) = \mathcal{F}(g_n^S(a) \cup g_n^S(b))$ for two strings $a, b \in \{0,1\}^*$. We will come back to these ideas when we talk about generalized scoring rules. In this section, though, we will be less interested in the functions $f, g$ (the variants of $\sigma, \rho$) and more interested in the set $S$ and the relations between different representations.

We define the connection between a representation and a voting rule.

**Definition 4.1.2.** *A representation $S$ is **sufficient** for a voting rule $\mathcal{F}$ if for any $n$ voter profile $P$ and $n'$ voter profile $Q$ we have $\mathcal{F}(P \cup Q) = \mathcal{F}(g_n^S(f_n^S(P)) \cup Q)$. A representation $S$ is **necessary** for $\mathcal{F}$ if for any distinct $s_1, s_2 \in S_n$ there is some $n'$ voter profile $Q$ such that $\mathcal{F}(g_n^S(s_1) \cup Q) \neq \mathcal{F}(g_n^S(s_2) \cup Q)$.*

In terms of the quotient sets we mentioned earlier, a representation is sufficient if it doesn't identify two profiles which are not equivalent. A representation is necessary if it identifies all the profiles which are equivalent (that means it is really the quotient set of the $\sim$ relation induced by the rule).

We present a list of various representations of preferences which are found in the literature, and which we have already seen in one form or another. These sets are used either by Chevaleyre et al. [3, 12] or by Xia and Conitzer [21], though they define them only in as much as they need to get the compilation complexity results, and in some cases the definitions of the sets are a bit different. At this point we will only define the sets (to some level of accuracy), and later we will show that they satisfy our requirements.

- Profiles (P): The representation we start from is $\mathcal{L}(X)^n$, which can be called trivial in some sense. Though it is sufficient for all voting rules, and is necessary only for rules which are "informationally bad", meaning we cannot condense any information. However, as soon as we know that the rule at least obeys anonymity – as most common rules do – we can switch to a (slightly) better representation:

- Anonymous profiles (AP): As we've seen when we looked at compilation complexity, we can see this as the set of all vectors of length $m!$ with non-negative entries that sum up to $n$, where the entry $v_i$ for some vector is the number of people who voted $V_i$ (according to a chosen enumeration of ballots).

- ABOVE: We define the set ABOVE to be all the functions $f : X \times (\mathcal{P}(X) \setminus \{X\}) \to \mathbb{N}$ satisfying:

  I. $\sum_{1 \leq i \leq m} f(x_i, \emptyset) = n$

  II. $\forall x \in X, Z \subseteq X, \quad x \in Z \Rightarrow f(x, Z) = 0$

  III. $\forall Z \subseteq X, Z \neq \emptyset \quad \sum_{1 \leq i \leq m} f(x_i, Z) = \sum_{x_j \in Z} f(x_j, Z \setminus \{x_j\})$

  For $f \in$ ABOVE, $x \in X$ and $Z \subset X$, $f(x, Z)$ is to be interpreted as the number of voters who prefer all the candidates in $Z$ to $x$, and prefer $x$ to any other candidate in $X \setminus Z$. In light of this, the first requirement corresponds to having a profile of $n$ voters, the second requirement means that we have linear orders ($x$ cannot follow $x$), and the third requirement is a sort of transfer condition from $k$ tuples to $k+1$ tuples, for $0 \leq k < m$. We will show that this set coincides with the different combinations for $above(P, Z, x)$ mentioned by Chevaleyre et al. [3, 12] in the context of STV (and is therefore sufficient and necessary for STV) and prove some more properties about this set.

- Weak semimagic squares (WSM): We define a weak semimagic square (WSM-square) of order $m$ and sum $n$ as a matrix $r$ of size $m \times m$, whose entries are all non-negative integers (not necessarily distinct), and the sum of every row equals the sum of every column equals $n$. We denote the entry in the $i$th row and $j$th column by $r(i, j)$. We denote the set of all WSM-squares by WSM.[1]

  We identify $r(i, j)$ with the number of voters who rank the $i$th candidate in the $j$th place. This set is used by Xia and Conitzer [21] for calculating the compilation complexity of Bucklin.

[1] We call these squares "weak semimagic" to tell them apart from the standard magic squares, where one usually also requires that the entries are distinct integers, ranging from 1 to $m^2$ – hence the sum is determined from the order, satisfying $n = \frac{m(m^2+1)}{2}$ – and in addition the sum on each of the main diagonals is also required to be equal to $n$. The 'semi' means we drop the restriction from the diagonals, and 'weak' means we allow entries to range from 0, and have repetitions.

- Weighted majority graphs (WMG): This is the set $\{G(P) : P \in \mathcal{L}(X)^n$, where $G(P)$ is the graph constructed as described in Appendix A, based on $N(x, y, P)$.

  As was described when we compared compilation and communication complexity, WMG is sufficient for many Condorcet consistent rules (though not all), and it is necessary for some of them. The set is used by both Chevaleyre et al. [3] and Xia and Conitzer [21].

- Scoring vectors for positional scoring rules (PSR): This actually includes many different representations, each representation associated with a different scoring vector. We know each positional scoring rule is associated with a scoring vector $\langle v_1, \ldots, v_m \rangle$, so the representation for that rule is the set of vectors

  $$\{\langle u_1, \ldots, u_m \rangle : \forall 1 \leq i \leq m \; u_i = \sum_{1 \leq j \leq m} N(x_i, j, P) v_j \text{ for some } P \in \mathcal{L}(X)^n\}$$

  Positional scoring rules are widely used in the literature. For simple vectors (like that of plurality) the corresponding set is easy to characterize, but it is more complicated for other vectors (as in the case of Borda, for which both Chevaleyre et al. [3] and Xia and Conitzer [21] calculated compilation complexity bounds).

By no means is this list supposed to be an exhaustive description of all the sufficient and necessary representations for voting rules. In addition, while some of those structures are simple to characterize as well as to count (as in the case of scoring vectors for the Plurality rule), others are more complicated (if we could count them all, we would have the compilation complexity for many rules). We can also see that in some cases our representation is profile-dependent (namely, for WMG and the general PSR representation, where we rely on the set of profiles in the definition of the representation), and for other cases the definition is profile independent.

Mapping these representations is based mostly on pragmatic accounts, trying to balance between wanting to capture as many rules as possible, and having an interesting representation you can work with. The work here is focused mostly on the WSM and ABOVE representations.

In the next section we discuss the connections between these representations, but first we must show that these are all indeed representation. While this might be trivial for the WMG and PSR, where, as we explain, our set is the image of the function $f_n^S$ on the profiles, it is less trivial for ABOVE and WSM.

**Proposition 4.1.3.** *The sets P, AP, WMG, and PSR are all representations of preferences.*

*Proof.* For the set of profiles P there is nothing to prove (the functions $f_n, g_n$ are just the identity function). We define explicitly the functions $f_n^{AP}$ and $g_n^{AP}$ for the case of AP, while for WMG and PSR only $f_n^S$ are explicit and not $g_n^S$, and we have to use the definition of the set.

**AP:** Fix an ordering of the ballots $V_1, \ldots, V_{m!}$. $f_n^{AP}(P) = \langle v_1, \ldots, v_{m!} \rangle$ where $v_i$ is the number of voters in $P$ who voted $V_i$. $g_n^{AP}(\langle v_1, \ldots, v_{m!} \rangle)$ is a profile where the first $v_1$ voters vote $V_1$, the next $v_2$ voters vote $V_2$, etc. Since the sum of the elements in $n$, we get an $n$ voter profile (and it's simple to check that $f_n^{AP}(g_n^{AP}(s))$ for any vector $s$ in AP).

**WMG** Given a profile $P$, we can generate a majority graph based on $N(x, y, P)$ for any two candidates $x, y \in X$, which gives us $f_n^{WMG}$. As we have defined WMG to be the set of majority graphs which correspond to some profile, we know that any element of WMG corresponds to at least one profile. Given an ordering of the profiles, make $g_n^{WMG}(s)$ the first profile that correlates to $s \in$WMG (this is of course a non-constructive definition, resulting from not having a clear characterization of WMG).

**PSR** As this category includes different scoring rules, we will have different $f_n^S$ functions (based on different scoring vectors $\langle v_1, \ldots, v_m \rangle$, though all are of the form $f_n(P) = \langle \sum_{1 \le j \le m} N(x_1, j, P)v_j, \ldots, \sum_{1 \le j \le m} N(x_m, j, P)v_j \rangle$), and different $g_n$ functions as well. For the cases which are simple to characterize (these are mostly rules similar to plurality, $k$-approval etc.) we can give an explicit $g_n$, but for other cases we have to define $g_n$ in a non-constructive way, in the same way we did for WMG.

□

We now turn to the ABOVE representation, which is slightly more complicated. We will define functions $f_n^{ABOVE}$ which take an $n$ voter profile $P$ and return a function $f_P \in ABOVE$. In accordance with what we said earlier, we will want $f_P(x, Z)$ to be the number of voters in $P$ who prefer all the elements of $Z$ to $x$ and prefer $x$ to any other element in $X \setminus Z$, which is precisely $above(P, Z, x)$ used by Chevaleyre et al. for STV [3, 12], which we mentioned in Example 2.2.9.

Our characterization of ABOVE is different from the *above* function and somewhat beneficial since it is profile-independent, and in addition it helps in analyzing various properties of that set. In Appendix C we derive a new upper bound for STV, based on this representation.

We should show, however, that we can actually get $f_n^{ABOVE}(P) = f_P = above(P, \cdot, \cdot)$ for every profile, which amounts to showing that $above(P, \cdot, \cdot)$ satisfies the conditions of ABOVE, and that no other function does it.

**Proposition 4.1.4.** *The set ABOVE is a representation of preferences.*

*Proof.* We define $f_P(x, Z) = above(P, Z, x)$, based on the notation of Chevaleyre et al. from Example 2.2.9. We will show that this function satisfies the conditions of ABOVE, and so we get $f_n^{ABOVE}(P) = f_P$. The first two conditions are obvious, and the last one is also simple to understand: Given a nonempty $Z \subset X$ and $x \in X$, we need to show that

$$\sum_{1 \le i \le m} f_P(x_i, Z) = \sum_{x_j \in Z} f_P(x_j, Z \setminus \{x_j\})$$

But this simply translates to

$$\sum_{1\leq i\leq m} above(P, Z, x_i) = \sum_{x_j\in Z} above(P, Z \setminus \{x_j\}, x_j)$$

which holds for any profile (both sides of the equations are just the number of ballots whose first $|Z|$ candidates are the elements of $Z$).

Now, let $f$ be a function in ABOVE, we show that there is a profile matching it (which would mean that $f = above(P, \cdot, \cdot)$ for some $P$). Fix an order on the candidates, $x_1, \ldots, x_m$. We construct all the ballots simultaneously, starting from the top: $f(x_1, \emptyset)$ ballots starting with $x_1$, $f(x_2, \emptyset)$ ballots starting with $x_2$ etc. Condition I on ABOVE guarantees that we have $n$ ballots after this step. At the $k + 1$ step we take the first set $Z \subset X$ of size $k$ (order on sets induced from the order on candidates). Let $N(Z)$ be the number of (partial) ballots in our profile consisting of $Z$. We know that $N(Z) = \sum_{x_j\in Z} f(x_j, Z\setminus\{x_j\})$, and from condition III we get that $N(Z) = \sum_{1\leq i\leq m} f(x_i, Z)$, so we just place the candidates for which $f(x_i, Z) > 0$, according to the order. This means that we can continue all the tuples of size $k$ to tuples of size $k + 1$, and so after $m$ steps we are done. It is simple to check that $f_n^{ABOVE}(P) = f$. □

Last, we show that WSM is also a representation.

**Proposition 4.1.5.** *The set WSM is a representation of preferences.*

While this claim might seem simple (and one direction is indeed simple, namely finding $f_n^{WSM}$), the proof that any WSM-square fits to some profile is surprisingly long, and it is worth mentioning that trying to build a profile from such a square in a naive greedy way might fail, which is why the proof does it differently.

*Proof.* Fix an ordering of the candidates, $x_1, \ldots, x_m$. We have $f_n^{WSM}(P) = r$ where $r(i, j) = N(x_i, j, P)$ is the number of voters who ranked $x_i$ in the $j$th place. Since every voter has to rank $x_i$ somewhere, $\sum_{1\leq i\leq m} r(i, j) = n$. Since every voter has to rank someone in the $j$th place, $\sum_{1\leq j\leq m} r(i, j) = n$, so $r$ is a WSM-square of order $m$ and sum $n$.

Next, we show that we can come up with a profile, given a WSM-square, though we don't give an explicit function $g_n^{WSM}$. We make use of the following lemma:

**Lemma 4.1.6.** *Given a WSM-square $r$ of order $m$ and sum $n > 0$, we can find a permutation $\sigma$ of $\{1, \ldots, m\}$ such that for every $1 \leq i \leq m$ we have $r(i, \sigma(i)) > 0$.*

In words, the lemma says that in every WSM-square with a positive sum, we can find $m$ entries, all of them positive, and no two of them share the same raw or column. As the proof is (surprisingly) not trivial, and it uses some concepts

not directly related to voting theory (flow networks), the proof and the required definitions are presented in Appendix B. We now show we can come up with a profile, given the lemma.

Given a WSM-square $r$ of order $m$ and sum $n > 0$, we will construct a profile $P$ that corresponds to $r$ in $n$ steps. On every step we take the $\sigma$ given to us by the lemma. We add a ballot $V$ to $P$, where $V$ is the following: $\langle x_{\sigma^{-1}(1)} > x_{\sigma^{-1}(2)} > \cdots > x_{\sigma^{-1}(m)} \rangle$. We then subtract 1 from the $(i, \sigma(i))$ entry for every $1 \leq i \leq m$ (the lemma guarantees that all these entries are positive, so we can in fact do that). It is simple to see that what we get, after the reduction, is another WSM-square of order $m$, but with sum $n - 1$. We can repeat this process, and the lemma guarantees that we finish this after $n$ steps, with a WSM-square of sum 0.

Verifying that $f_n^{WSM}(P) = r$ is left to the reader. □

By better understanding these representations and the relations between them, we gain more insights into the compilation complexity of the rules utilizing them, as well as to the rules themselves, and so that is what we try to do next.

## 4.2 Hierarchy of Representations

We try to give a short analysis of the relations between the representations we described. We define what it means for one representation to be finer than another.

**Definition 4.2.1.** *We say that a representation $S$ is **finer** than a representation $T$ if there are functions $f_n^{S,T} : S_n \to T_n$ satisfying $f_n^{S,T}(f_n^S(P)) = f_n^T(P)$ for any $n \in \mathbb{N}$ and $n$ voter profile $P$.*

Notice that the definition implies that $f_n^{S,T}$ is onto. It also follows from the definition that, naturally, the set of profiles P is the finest representation. If also follows that if $A$ is finer than $B$ and $B$ is finer than $C$, we get that $A$ is finer than $C$, since we can define $f_n^{A,C} = f_n^{B,C} \circ f_n^{A,B}$, and we get that for any $n$ and $P$:

$$f_n^{A,C}(f_n^A(P)) = f_n^{B,C}(f_n^{A,B}(f_n^A(P))) = f_n^{B,C}(f_n^B(P)) = f_n^C(P)$$

From this, together with some relations we will soon prove, we will get Diagram 4.1, where an arrow from $S$ to $T$ means that $S$ is finer than $T$ (we present only arrows between adjacent representations, and not arrows we get by composition).

We prove the relations in the diagram hold:

**Proposition 4.2.2.** *We have the following relations between the representation introduced so far:*

1. *P is finer than AP*
2. *AP is finer than ABOVE.*

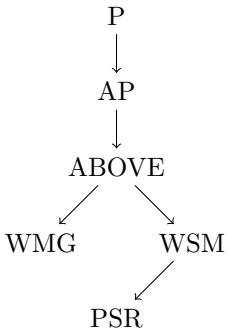


Figure 4.1: Hierarchy of representations for common voting rules. An arrow from $A$ to $B$ means that $A$ is finer than $B$.

3. *ABOVE is finer than WMG.*

4. *ABOVE is finer than WSM.*

5. *WSM is finer than PSR.*

*Proof.* As we said, P is always the finest representation, so Claim 1 is trivial. We prove the other cases. In all cases, our function $f_m^{A,B}$ will simply be $f_n^B \circ g_n^A$, and so we have to show that for any profile $P$, $f_n^B(g_n^A(f_n^A(P))) = f_n^B(P)$. Alternatively, we can show that whenever two profiles $P, P'$ satisfy $f_n^A(P) = f_n^A(P')$, we have $f_n^B(P) = f_n^B(P')$. To see that this is equivalent, denote $P' = g_n^A(f_n^A(P))$. From the definition of representation it follows that $f_n^A(P') = f_n^A(P)$, and from this follows the equivalence.

2. It's simple to see that given a profile $P$, the order of the ballots has no affect on $above(P, Z, x)$, and so $f_n^{ABOVE}(g_n^{AP}(f_n^{AP}(P))) = f_n^{ABOVE}(P)$.

3. We need to show that for $P, P'$ such that $f_n^{ABOVE}(P) = f_n^{ABOVE}(P')$ we have $N(x, y, P) = N(x, y, P')$. This is the case, because

$$N(x, y, P) = \sum_{\substack{Z \subset X, \\ x \in Z}} above(P, Z, y) = \sum_{\substack{Z \subset X, \\ x \in Z}} above(P', Z, y) = N(x, y, P')$$

4. It's enough to show that for any two profiles $P, P'$ such that $above(P, Z, x_i) = above(P', Z, x_i)$ for all $x_i, Z$, we have $N(x_i, j, P) = N(x_i, j, P')$. But

$$N(x_i, j, P) = \sum_{\substack{Z \subset X, \\ |Z| = j-1}} above(P, Z, x_i) = \sum_{\substack{Z \subset X, \\ |Z| = j-1}} above(P', Z, x_i) = N(x_i, j, P')$$

so ABOVE is finer than WSM.

5. Take a scoring vector $\langle v_1, \ldots, v_m \rangle$. We have to show that for any $P, P'$ such that $f_n^{WSM}(P) = f_n^{WSM}(P')$ we get the same score for every candidate. For a candidate $x_i$, the score of $x_i$ is:

$$\sum_{1 \leq j \leq m} N(x_i, j, P) v_j = \sum_{1 \leq j \leq m} N(x_i, j, P') v_j$$

so we have the claim. □

We further mention that we cannot add additional arrows to Figure 4.1 (beyond arrows which we get using composition). For example, WSM is not finer than WMG. Consider the profiles

$$P = \begin{cases} x_1 > x_2 > x_3 > x_4 \\ x_3 > x_4 > x_1 > x_2 \end{cases} \quad P' = \begin{cases} x_3 > x_2 > x_1 > x_4 \\ x_1 > x_4 > x_3 > x_2 \end{cases}$$

The profiles satisfy $f_2^{WSM}(P) = f_2^{WSM}(P')$ but $f_2^{WMG}(P) \neq f_2^{WMG}(P')$. Similarly, take:

$$P = \begin{cases} x_1 > x_2 > x_3 > x_4 \\ x_4 > x_3 > x_2 > x_1 \end{cases} \quad P' = \begin{cases} x_2 > x_1 > x_4 > x_3 \\ x_3 > x_4 > x_1 > x_2 \end{cases}$$

These profiles satisfy $f_2^{WMG}(P) = f_2^{WMG}(P')$ but $f_2^{WSM}(P) \neq f_2^{WSM}(P')$, so WMG is not finer than WSM.

We still need to say something about how sufficient and necessary these representations are, and with respect to which voting rules. Connecting this to the hierarchy we displayed, we want to find for any rule $\mathcal{F}$ the representation which is sufficient to compute $\mathcal{F}$ and it is the least fine representation that does that (that would be the closest to a necessary representation).

We know that ABOVE is sufficient and necessary for STV, and some new results by Chevaleyre et al. point out that it might also be sufficient for other rules [12]. WMG is sufficient for many Condorcet consistent rules, as we saw in the case of compilation complexity, though it is not necessary for all.

WSM is sufficient for Bucklin, though it is not necessary (a Bucklin winner has to be found within the first $\lfloor \frac{m}{2} \rfloor + 1$ columns, and so any two squares that identify on those are equivalent for Bucklin). However, PSR representations are not fine enough for Bucklin, and we can also think of Bucklin variations for which that representation is necessary (one where the Bucklin score is computed not according to where the candidate gets more than half the votes, but rather when they get all the votes, for example).

PSR are sufficient and necessary for all positional scoring rules.

To conclude this section, we have shown how the information utilized by voting rules can be put into different structures, and those structure form a hierarchy of a sort. The study of the representations and the relations between them can assist in estimating measures such as compilation complexity, which we can try to get either from counting the elements of the representation (as we do for ABOVE in Appendix C, to get an upper bound for the compilation complexity of STV), or from looking at another representation which is close to it in the hierarchy whose size is known, and using the mapping between the two representations to estimate the size of the unknown representation (we elaborate

on this in the final discussion). Another advantage is that it gives a more formal way of looking at voting rules, thus seeing similarities and differences between different rules.

In the next section we present another formalism for voting rules, which has elements related to both compilation complexity and communication complexity.

## 4.3 Generalized Scoring Rules

In the previous sections we mentioned how one informative representation for all voting rules seems unlikely, and so we've focused on describing different representations and the relations between them. In this section we try to challenge that claim, by presenting a way of describing voting rules which seems more informative than the functional form, and yet captures a very large portion of those rules.

The idea we will be presenting is that of generalized scoring rules (GSR), developed by Xia and Conitzer [19, 20]. We will present the definitions and the results Xia and Conitzer had for those rules, and in the following section we will show how this connects to both compilation and communication complexity. We start from defining the term:

**Definition 4.3.1** (Generalized scoring rules [19, 20])**.** *Let $k \in \mathbb{N}$ and $K = \{1, \ldots, k\}$. Let $\mathcal{K} = \{K_1, \ldots, K_q\}$ be a partition of $K$ into $q$ sets. We say that two vectors $a, b \in \mathbb{R}^k$ are equivalent, $a \sim_{\mathcal{K}} b$, if for every element $K_l$ of the partition, $l \leq q$, if $i, j \in K_l$ then we have $a_i \leq a_j \Leftrightarrow b_i \leq b_j$.*

*Let $\mathcal{K}$ be a partition of $K$, we say that a function $g : \mathbb{R}^k \to X$ is compatible with $\mathcal{K}$ if for any $a, b \in \mathbb{R}^k$, $a \sim_{\mathcal{K}} b \Rightarrow g(a) = g(b)$.*

*Let $f : \mathcal{L}(X) \to \mathbb{R}^k$ and $g : \mathbb{R}^k \to X$, where $g$ is compatible with a partition $\mathcal{K}$ of $K$. $g$ and $f$ define the following voting rule: $GS(f, g)(V_1, \ldots, V_n) = g(\sum_{i \leq n} f(V_i))$. We say that $GS(f, g)$ is of order $k$ and compatible with $\mathcal{K}$.*

We will say that $GS(f, g)$ computes $\mathcal{F}$ if $\mathcal{F}(P) = GS(f, g)(P)$ for any $P$, and we will say in such a case that $\mathcal{F}$ is GSR.

Examples of GSR can often be taken from the compilation complexity examples, when the function $f$ is taken to be $f_1^S$ from the previous sections, for the representation $S$ which is appropriate for the voting rule (we will get back to this when we connect GSR with compilation and communication complexity), and one simply has to check that some $g$ is compatible with some partition of the output of $f$. Xia and Conitzer showed that all positional scoring rules, STV, Copeland, maximin, ranked pairs and Bucklin are GSR.

For example, in order to turn Copeland into a GSR, take $k = m(m-1)$, and our vectors are of the form $\langle v_{1,2}, v_{1,3}, \ldots, v_{1,m}, v_{2,1}, v_{2,3}, \ldots, v_{m,m-1} \rangle$ (that is, an entry $v_{i,j}$ for any distinct $i, j \leq m$). We take $f(V)_{i,j} = 1$ if $x_i$ is preferred to $x_j$, 0 otherwise. We get that $f(P)_{i,j} = N(x_i, x_j, P)$, and we know that we can compute the Copeland winners from that, so we define $g$ to do that. We

can verify that the trivial partition into one set is sufficient here (though there are better partitions as well).

Xia and Conitzer first characterized, for this set of rules, the probability with which a random profile can be manipulated by a coalition of voters, as a function of the number of candidates, voters and manipulators [19]. In their second paper [20], they gave an axiomatic characterization of voting rules which can be presented as GSR, which we next define:

**Definition 4.3.2.** *A voting rule $\mathcal{F}$ is **locally consistent** on a set of profiles $S$ if for any $P_1, P_2 \in S$ such that $\mathcal{F}(P_1) = \mathcal{F}(P_2)$ we have $\mathcal{F}(P_1 \cup P_2) = \mathcal{F}(P_1) = \mathcal{F}(P_2)$.*

*$\mathcal{F}$ is $t$-consistent if there is a partition of the set of all profiles $\mathcal{L}(X)^*$ into $t$ sets, $\{S_1, \ldots, S_t\}$, such that $\mathcal{F}$ is locally consistent on each $S_i$.*

*$\mathcal{F}$ has finite local consistency if it is $t$-consistent for some $t \in \mathbb{N}$.*

The theorem that Xia and Conitzer proved was not only that rules with finite local consistency are GSR, but it also tied the degree of consistency of those rules with the maximal order of the GSR associated with them:

**Theorem 4.3.3** (Theorem 1 in [20])**.** *A voting rule $\mathcal{F}$ is GSR iff it is anonymous and finitely locally consistent. In addition, let $t$ be the degree of consistency of $\mathcal{F}$, then there is a GSR of order at most $\frac{t(t-1)m(m-1)}{4}m!$ that computes $\mathcal{F}$.*

Xia and Conitzer also proved, in the same paper, the degree of consistency of various rules. It is worth mentioning, though, that the upper bound we get from the local consistency is, in many of their examples, much higher than the actual minimal order for the rule.

It seems therefore that GSR can capture a variety of voting rules (though not all: Dodgson, for example, doesn't satisfy finite local consistency). But what can we get from this representation? We try to answer this in the next section, when we show how GSR connect to the representations we spoke of earlier, and to communication and compilation complexity.

## 4.4 Connecting GSR with Complexity Measures

Any GSR is composed of two parts: $f : \mathcal{L}(X) \to \mathbb{R}^k$ and $g : \mathbb{R}^k \to X$. While $f$ is more about aggregating the information given by the profiles and representing it in a form appropriate for the rule, $g$ is more about processing that form and calculating the results. We will show that in some sense, $f$ corresponds to the compilation function in the compilation complexity problem, while $g$ gives a protocol for the communication complexity problem. In this section we investigate the nature of this correspondence. We start with the connection between GSR and compilation complexity, which is the clearer one.

Unlike compilation complexity, which is defined for any voting rule, GSR correspond to a smaller class of voting rules: namely, as we have seen, anonymous rules which have the finite local consistency property.

However, if we restrict our view to these rules alone, we notice that the requirements of $f$ in the GSR are quite similar to those of the compilation function. What do we mean by "quite similar"? We try to give a more formal meaning:

We want to be able to define a compilation function $\sigma$ for a voting rule $\mathcal{F}$, given functions $f$ and $g$ such that $GS(f,g)$ computes $\mathcal{F}$, and we want to be able to define the functions $f$ and $g$ for $\mathcal{F}$, given a compilation function $\sigma$.

First, given functions $f$ and $g$, we can construct $\sigma$: For $P = \langle V_1, \ldots, V_n \rangle \in \mathcal{L}(X)^n$, let $\sigma(P) = f(P) = \sum_{1 \leq i \leq n} f(V_i)$. We then define

$$\rho(\sigma(P), P') = \rho(f(P), P') = g(f(P) + f(P')) = g(f(P \cup P')) = \mathcal{F}(P \cup P')$$

There is one problem with this construction, and that is the fact that $f$ generates a vector of real numbers, and $g$ operates on a vector of reals. Computationally speaking, of course, there is no way of representing all possible reals, and no way of performing arithmetical operations on them, so $\sigma$ and $\rho$ cannot be taken to be exactly the functions we described. We have to "round" the values in the vector at some point (turning them not necessarily to integers, but they must be represented by a finite number of digits, so it has to be a rational number).

This process of approximating real values using rational numbers might become problematic. Take a positional scoring rule with a non-rational vector, for example $\langle \pi, 2, 1, 0 \rangle$. It can be shown that no matter how good our approximation is, we can always find a profile with enough voters where the rounded values give a result which is different from the one of the rule itself.

In practice, though, we see that all the representations in Section 4.2 use integer numbers, with the exception of positional scoring rules, which could be anything (and yet, the popular ones are integers).

On the other direction, we would like to construct $f$ and $g$ given $\sigma$. This, however, is not a simple task. In general, we are not guaranteed that $\sigma$ satisfies the requirement of additivity which must hold for $f$, that is, $f(P \cup P') = f(P) + f(P')$. Indeed, it is not even clear what is the meaning of additivity for a function that takes values if $\{0,1\}^*$.

If we look at the representations in Section 4.2, which correlate with the compilation complexity conditions for equivalent profiles, then they all satisfy the additivity constraint. Take such a representation $S$, then the elements of $S$ can be described as a vector of numbers of some length, and the functions $f_n^S$ satisfy $f_n^S(P) + f_{n'}^S(Q) = f_{n+n'}^S(P \cup Q)$.

However, not every representation is necessarily given is such a comfortable form of numerical vectors. The theorem of Xia and Conitzer guarantees that for every anonymous rule with finite local consistency we have such a representation, but there is no necessity that it is a minimal representation (in the terminology we used in the beginning of the chapter – necessary representation). At this point the question of which conditions we need in order to have additivity for a necessary representation remains open.

Another problem in this construction is that even once we have an additive representation, that is not enough to guarantee the compatibility condition for $g$. This question also remains open for now. We will go back to this in the final discussion.

We now move to comparing $g$ with the communication protocol in the communication complexity problem.

As we have seen in the case of STV, the size of the representation for the compilation complexity can be substantially bigger than the actual computational complexity or communication complexity required for calculating the winner. When we inspect this discrepancy via GSR, we understand that $g$ doesn't have to go through all the $k$ values in the vector, but can rather look each time at a specific value / subset of values, from which it can determine which values to look at next. That is, if $f(P) = \langle v_1, \ldots, v_k \rangle$, we can define a function that tells us what is the next value $g$ is going to inspect, given the previous values.

Formally: Given a profile $P$ and a set of indices $Z = \{i_1, \ldots, i_{k'}\} \subseteq \{1, \ldots, k\}$, denote by $f(P)|_Z$ the vector $\langle v_{i_1}, \ldots, v_{i_{k'}} \rangle$.

Now define recursively $g_1 = Z_1 = \{i_1\}$ for some index $i_1$, and

$$g_{j+1}(f(P)|_{Z_j}) = \begin{cases} \emptyset & \text{If } f(P)|_{Z_j} \text{ is enough to calculate } g \\ \{i_{j+1}\} & \text{otherwise, for some } i_{j+1} \notin Z_j \end{cases}$$

and

$Z_{j+1} = Z_j \cup g_{j+1}(f(P)|_{Z_j})$

Of course, there are many ways of defining these series (depending on the choice if $i_{j+1}$), so we take a series $g_j, Z_j$ that stabilizes as least as fast as any other series.

We now present a communication protocol given $f$ and $g$. At first, each voter $j$ calculates $f(V_j) = \langle v_1^j, \ldots, v_k^j \rangle$, and then transmits $v_{i_1}^j$ to all other voters ($\{i_1\} = g_1 = Z_1$). Now all the voters have $f(P)|_{Z_1}$, and so they can all calculate $g_2(f(P)|_{Z_1})$. Next they transmit the entry in the new index and so on. At the end of the process it is guaranteed that we have calculated $g(f(P))$, and have accessed no more information than that of $g$. This means that the number of values accessed by $g$ times the size of entries of the vector for single ballots times $n$ gives us an upper bound on the communication complexity.[2]

There is one sense in which this result is not surprising. It is known in communication complexity theory, that there is a connection between the circuit complexity of Boolean functions and the communication complexity of relations based on those functions (for details, see [11, Chapter 10]). The connection between the $g$ from GSR and the communication complexity is just another aspect of the connection between computational complexity and communication complexity. We go back to this in the discussion in the end of the thesis.

To conclude, we showed that under reasonable assumptions on the generalized scoring rules (namely, that $f$ only takes integer or rational values), we can generate both compilation functions and communication protocols. We also know that for all the examples of GSR given by Xia and Conitzer, the represen-

[2] In this protocol we assumed that $f$ gives only rational numbers, as we did for the compilation complexity.

tations we presented earlier in the chapter are used by the generalized scoring rules (by taking $f = f_1^S$ for the appropriate representation[3]).

On the other direction, however, it seems that constructing $f$ and $g$ given a compilation function (or a communication protocol) is more complicated, which implies that those complexity measures alone do not give a formalization as strong as that of GSR.

[3]Though sometimes $f$ also adds some elements which we get by adding or subtracting elements in $f_1^S(V)$.

# Chapter 5

# Group Correspondence

In Chapter 4 we saw different representations for voting rules, which were mostly related to compilation complexity. In this chapter we consider an algebraic representation which is related to informational size, introduced in Chapter 3.

Remember that the equivalence of ballots, as defined for informational size, induced partitions of the ballot space. In this chapter we establish a connection between informational size and group theory, by constructing a one-to-one correspondence between partitions induced by neutral and anonymous voting rules and subgroups of the symmetric group (a special type of group in group theory, which will be defined in the next section) satisfying some condition.

We will see how this correspondence can be used to carry results from group theory (which is a well-studied field) into voting theory, proving a claim posed by Sato in his paper on informational size [14]. No less important is the outlook we get from the correspondence on various notions in voting theory, and their relation with group theory, which is a surprising connection.

We now turn to some basic definitions and facts, mostly from group theory.

## 5.1 Basic Definitions

Most of this section is devoted to various definitions and facts from group theory which we will need in the proof of the correspondence. Before we go into the group-theoretic definitions, though, we recall some concepts related to informational size.

We saw in Chapter 3 how, given a voting rule $\mathcal{F}_{m,n}$ for $m$ alternatives and $n$ voters, we get $n$ partitions of the ballot space $\mathcal{L}(X)$, which we denoted $\mathcal{M}_{i,\mathcal{F}_{m,n}}$ for $1 \leq i \leq n$, where $\mathcal{M}_{i,\mathcal{F}_{m,n}}$ is the partition corresponding to the equivalence relation $\sim_i$ for ballots (for the exact definition, see Chapter 3).

We say that the voting rule $\mathcal{F}_{m,n}$ induces the partitions $\mathcal{M}_{i,\mathcal{F}_{m,n}}$, and the elements of each partition are sometimes referred to as equivalence classes. Our correspondence will be defined for these different partitions, so they are the objects which interest us most.

There are two important lemmas proved by Sato in his paper, which we will be using. One is Fact 3.2.4, which was already mentioned in Chapter 3, which says that for the case of an anonymous rule (A-rule) we have $\mathcal{M}_{i,\mathcal{F}_{m,n}} = \mathcal{M}_{j,\mathcal{F}_{m,n}}$ for every two voters $i, j$. In this case we can omit the subscript $i$ and write simply $\mathcal{M}_{\mathcal{F}_{m,n}}$, *the* partition induced by $\mathcal{F}_{m,n}$.

The second lemma is:

**Fact 5.1.1** (Lemma 4.2 in [14]). *Let $\sigma$ be a permutation of the elements of $X$, the set of alternatives. Denote by $\sigma(l)$ the result of $\sigma$ acting on the linear order $l$, i.e., if $l = x_1 > x_2 > \cdots > x_m$, then $\sigma(l) = \sigma(x_1) > \sigma(x_2) > \cdots > \sigma(x_m)$, and for a set of linear orders $M$, denote $\sigma M = \{\sigma(l) : l \in M\}$. For any neutral voting rule (N-voting rule) $\mathcal{F}$, if $M \in \mathcal{M}_{i,\mathcal{F}}$ then $\sigma M \in \mathcal{M}_{i,\mathcal{F}}$.*

Through most of the chapter we will work with neutral and anonymous rules (NA-rules), and we will also assume our voting rules are non-constant, that is, for any rule $\mathcal{F}$ there are two profiles $P, P'$ such that $\mathcal{F}(P) \neq \mathcal{F}(P')$.

We now turn to the definitions and facts we need from group theory. All the following definitions and facts can be found in basic group theory texts, such as [17], and though all the definitions we will need appear here, a reader who wishes to see some more examples and explanations can find them there.

**Definition 5.1.2** (Group). *A **group** is a set with an operation: $(G, \cdot)$, where $G$ is the set and $\cdot : G \times G \to G$ is the operation, such that*

- *$\cdot$ is associative: $a \cdot (b \cdot c) = (a \cdot b) \cdot c$*
- *$G$ has a unit element $e$, such that for all $g \in G$, $g \cdot e = e \cdot g = g$*
- *$\cdot$ is closed under inverse elements: For all $g \in G$ there is $g^{-1} \in G$ such that $g \cdot g^{-1} = g^{-1} \cdot g = e$*

*The **order** of a group $G$ is the number of elements in $G$.*

We will usually omit the $\cdot$ notation, and write simply $gh$ instead of $g \cdot h$.

**Definition 5.1.3** (Subgroup). *A **subgroup** $H$ of $G$, denoted $H \leq G$ is any subset of $G$ that is also a group with the restricted operation.*

*Given $H \leq G$, a **(left) coset** of $H$ in $G$ is any set of the form $gH = \{g \cdot h : h \in H\}$ where $g \in G$. The **index of $H$ in $G$** is the number of different cosets of $H$ in $G$.*

*A subgroup $H \leq G$ is said to be **proper** if $H \neq G$, denoted $H < G$. $H$ is said to be **non-trivial** if $H \neq \{e\}$.*

We remark that there is an analogous definition of right cosets. The difference between left and right cosets is not relevant for our correspondence, as long as we choose one of the two and stick to it. We will use left cosets, and refer to them simply as cosets.

**Definition 5.1.4** (Normal subgroup). *A subgroup $N \leq G$ is **normal** if for any $g \in G$ and $n \in N$ we have $gng^{-1} \in N$.*

**Fact 5.1.5.** *Given $H \leq G$, let $G/H = \{gH : g \in G\}$ denote the set of cosets of the subgroup. $G/H$ constitutes a partition of $G$, which we will call "the partition induced by $H$".*[1]

[1] In group theory, the notation $G/H$ is usually reserved for the cases where $H$ is normal, where it can be shown that this set constitutes a group. We will only be interested in this set as a set and not as a group, so we use the notation for any $H$.

**Fact 5.1.6.** *Let $H < G$. The order of $H$ is a divisor of the order of $G$.*

**Definition 5.1.7.** *Given a subset $S$ of a group $G$, $g^{-1}Sg = \{g^{-1}sg : s \in S\}$ is called* ***the conjugate of*** *$S$* ***by*** *$g$.*

**Fact 5.1.8.** *A conjugate of a subgroup is a subgroup.*

**Definition 5.1.9** (Group homomorphism)**.** *Given two groups, $(G, \cdot)$ and $(H, *)$, a mapping $\varphi : G \to H$ is called a* ***homomorphism*** *if for any $g_1, g_2 \in G$ we have $\varphi(g_1 \cdot g_2) = \varphi(g_1) * \varphi(g_2)$.*

**Definition 5.1.10** (Kernel)**.** *Given groups $G, H$ and a homomorphism $\varphi : G \to H$, the* ***kernel*** *of $\varphi$ is the set $\{g \in G : \varphi(g) = e_H\}$, where $e_H$ is the unit element of $H$.*

**Definition 5.1.11** (The symmetric group)**.** *Denote by $Perm(X)$ the set of all permutations of $X$. $Perm(X)$, together with the operation of composition of permutations, constitutes a group. For a set $X = \{1, \ldots, m\}$ containing $m$ elements, that group is usually denoted $S_m$, and is referred to as* ***the symmetric group of*** *$m$* ***elements****.*[2] *The elements of $S_m$ are usually denoted by $\sigma, \rho$ etc.*

One can identify $S_m$ with permutations on any set $X$ of size $m$ (i.e., with $Perm(X)$), through a bijection from $\{1, \ldots, m\}$ to $X$. Since we would want to consider permutation of alternatives, and since taking the bijection into a formal account incurs a clutter of notation, we will throughout this chapter assume that the set of alternatives is $X = \{1, \ldots, m\}$.

**Definition 5.1.12** (Group action)**.** *Given a group $G$ and a set $X$, we define* ***an action of*** *$G$* ***on*** *$X$ to be a function $\cdot : G \times X \to X$, which satisfies:*

- *for all $g, h \in G$ and $x \in X$ we have $(gh) \cdot x = g \cdot (h \cdot x)$.*
- *$e \cdot x = x$, for the unit element $e \in G$.*

*We denote the action by $g \cdot x$ for $g \in G, x \in X$, or (more often) just by $gx$.*

The function takes pairs of elements, one from $G$ and one from $X$, to an element of $X$. This can be viewed as a process in which every element of $G$ "moves the elements of $X$ around". From the properties of groups and group action it follows that $gx = gy$ iff $g^{-1}gx = g^{-1}gy$, which is iff $x = y$. In other words, every element of $G$ induces a permutation of the elements of $X$. We can therefore treat an action of $G$ on $X$ as a mapping $\rho : G \to Perm(X)$, and in fact, the properties of group actions guarantee that $\rho$ is a group homomorphism. Here are some examples of group actions which we will use in our correspondence (verifying that these are indeed actions is mostly left for the reader):

[2] Usually in the literature the notation for the symmetric group is $S_n$. As we reserve $n$ for the number of voters, and $m$ for the alternatives, and since we are interested in actions on the set of alternatives, we are using $S_m$ instead.

1. A trivial example is the symmetric group $S_m$ acting on the set $X$, where $\sigma x$ for $\sigma \in S_m$ and $x \in X$ just moves $x$ to its image under the permutation $\sigma$.

2. A slightly more complicated example is $S_m$ acting on $\mathcal{P}(X)$, the powerset of $X$. This is a simple extension of our first example, except that now every element $\sigma$ acts on a set of elements $A \subseteq X$, taking it to the set $B = \{\sigma x \in X : x \in A\}$.

3. We can define the action of $S_m$ on the set of linear orders $\mathcal{L}(X)$, as described before, by taking, for any linear order $l = \langle l_1 > l_2 > \cdots > l_m \rangle$, $\sigma(l) = \langle \sigma(l_1) > \sigma(l_2) > \cdots > \sigma(l_m) \rangle$.

   We show that this is an action. Clearly, for any $l \in \mathcal{L}(X)$, we have $e(l) = l$, and for any $\sigma, \rho \in S_m$ we have

   $$\sigma(\rho(l)) = \sigma(\langle \rho(l_1) > \cdots > \rho(l_m) \rangle) = \langle \sigma(\rho(l_1)) > \cdots > \sigma(\rho(l_m)) \rangle =$$

   $$\langle (\sigma\rho)l_1 > \cdots > (\sigma\rho)l_m \rangle = (\sigma\rho)l$$

   We will use this action in the correspondence we define in the next section. Whenever notation allows for it, we will drop the parentheses and write simply $\sigma l$.

4. By further extending that last action, we can define an action of $S_m$ on a profile, where the image of $\sigma P$ is the profile composed of all the images of the ballots in $P$ (keeping my mind that a ballot is just a linear order), as defined in the previous action, i.e., for $P = \langle V_1, \ldots, V_n \rangle$ we have $\sigma P = \langle \sigma V_1, \ldots, \sigma V_n \rangle$.

5. Another important action we will use is the action of $S_m$ on left cosets (this can actually be defined for any group $G$, though we will only deal with $S_m$, which also has some more interesting properties with respect to this action): Let $H \leq S_m$ be a subgroup, and let $S_m/H = \{\sigma H : \sigma \in S_m\}$. We define the action of $S_m$ on $S_m/H$ as $\sigma(\sigma' H) = (\sigma\sigma')H$ for every $\sigma \in S_m$ and $\sigma' H \in S_m/H$. Thus each element of $S_m$ defines a permutation of the set of cosets $S_m/H$.

We mention that though this is the first time we are presenting these actions formally, in some sense we have already been implementing them. Actions 2 and 4 both appear in the definition of the neutrality axiom.

**Fact 5.1.13** (See, for example, in [6], Ex. 1.3.3)**.** *Let $H < S_m$, and $S_m$ acting on left cosets of $H$ in the way described above. The kernel of the mapping $\rho : S_m \to Perm(S_m/H)$ is the largest normal subgroup of $S_m$ contained in $H$.*

The notion of action brings forth another natural notion: that of orbits.

**Definition 5.1.14** (Orbit)**.** *Given an action of $G$ on $X$ we say that the **orbit** of $x \in X$ is the set $G_x = \{gx \in X : g \in G\}$.*

In simple words, the orbit of $x$ is the set of elements to which $x$ can be moved by the elements of $G$. It is a simple fact in group theory that the set of orbits generated by an action constitutes a partition of $X$, so we can define an equivalence relation $\sim$ on $X$, where $x \sim y \Leftrightarrow G_x = G_y \Leftrightarrow x = gy$ for some $g \in G$.

**Definition 5.1.15** (Stabilizer)**.** *Given $G$ acting on $X$, we call the set $stab_G(x) = \{g \in G : gx = x\}$ the* ***stabilizer of*** *$x$.*

**Fact 5.1.16.** *A conjugate of a stabilizer is a stabilizer.*

**Fact 5.1.17.** *Given a group $G$ acting on a set $X$, $stab_G(x)$ is a subgroup of $G$ for any $x \in X$.*

**Definition 5.1.18** (Faithful action)**.** *An action of $G$ on $X$ is* ***faithful*** *if the only element $g \in G$ satisfying $gx = x$ for all $x \in X$ is the unit element $e$.*

Remark: Given a group $G$ acting on $X$, an element $g \in G$ satisfies $gx = x$ for every $x \in X$ iff we have $\rho(g) = e_X$, where $\rho : G \to Perm(X)$ is the homomorphism associated with the action and $e_X$ is the identity permutation on $X$. In light of this and the last definition we can say that an action of $G$ on $X$ is faithful iff the kernel of the associated homomorphism $\rho : G \to Perm(X)$ is trivial (i.e., is equal to the unit element of $G$).

The following definitions relate more specifically to the symmetric group and to permutations:

**Definition 5.1.19** (Cycles)**.** *A permutation $\sigma$ of a set $X$ is called a* ***cycle*** *if there are $x_1, \ldots, x_k \in X$ such that $\sigma(x_i) = x_{i+1}$ for $1 \leq i < k$, $\sigma(x_k) = x_1$ and $\sigma$ fixes all other elements of $X$. We then use cycle notation to write $\sigma$ in the form $\sigma = (x_1 x_2 \cdots x_k)$. A cycle of length two is called a* ***transposition****. The empty cycle $()$ denotes the identity permutation.*

**Fact 5.1.20.** *Every permutation can be written down as a product of disjoint cycles.*

**Definition 5.1.21.** *A permutation is* ***even*** *(****odd****) if it can be written down in cycle notation as the product of an even (odd) number of transpositions.*

**Fact 5.1.22.** *Every permutation is either even or odd (exclusively).*

**Fact 5.1.23.** *The set of all even permutations in $S_m$ constitutes a subgroup, denoted $A_m$ and called the alternating group.*

**Fact 5.1.24.** *For any $m \neq 4$, the only proper non-trivial normal subgroup of $S_m$ is $A_m$. For $m = 4$ the only proper non-trivial normal subgroups of $S_4$ are $A_4$ and the normal Klein four-group, which is the subgroup*

$$\{(), (12)(34), (13)(24), (14)(23)\}$$

*of index* 6.

We are now ready to begin constructing the correspondence.

## 5.2 Paving the Way for the Correspondence

We start by building a basic correspondence between permutations and linear orders, which we will then "lift" into the correspondence we are aiming at, between partitions induced by NA-voting rules and subgroups of $S_m$.

**Proposition 5.2.1.** *There is a one-to-one correspondence between linear orders over $m$ elements and permutations of $m$ elements.*

*Proof.* First we choose a specific linear order $l_0 \in \mathcal{L}(X)$, $l_0 = \langle 1 > 2 > \cdots > m \rangle$, and now we define the correspondence $\eta_{l_0} : S_m \to \mathcal{L}(X)$ in the following way: Let $\sigma \in S_m$ be some permutation of $X$, then $\eta_{l_0}(\sigma) = \sigma(l_0) =$, where $\sigma(l_0)$ is the action on linear orders defined in Example 3 in the previous section. It is simple to verify that this is a bijection. □

This means that we can speak of linear orders as permutations, and sets of linear orders as sets of permutations (and vice versa). This is of course all with respect to our choice of $l_0$. Any order can be chosen for the correspondence, but once one has been chosen, it defines $\eta_{l_0}$, that defines which permutation goes to which linear order. Specifically, the identity element $e$ of $S_m$ corresponds to $l_0$.

The following lemma is rather technical, and its purpose is basically to show that the two diagrams presented in Figure 5.1 are commutative, for any permutation $\sigma$ and for $\eta_{l_0}$, the correspondence function.

$$\begin{array}{ccc} \mathcal{P}(\mathcal{L}(X)) & \xrightarrow{\eta_{l_0}^{-1}} & \mathcal{P}(S_m) \\ \downarrow \sigma & & \downarrow \sigma \\ \mathcal{P}(\mathcal{L}(X)) & \xrightarrow{\eta_{l_0}^{-1}} & \mathcal{P}(S_m) \end{array} \qquad \begin{array}{ccc} \mathcal{P}(S_m) & \xrightarrow{\eta_{l_0}} & \mathcal{P}(\mathcal{L}(X)) \\ \downarrow \sigma & & \downarrow \sigma \\ \mathcal{P}(S_m) & \xrightarrow{\eta_{l_0}} & \mathcal{P}(\mathcal{L}(X)) \end{array}$$

Figure 5.1: Commutativity of permutations and $\eta_{l_0}$.

**Lemma 5.2.2.** *Let $\eta_{l_0}$ be the correspondence function defined in the proof of Proposition 5.2.1. The following holds for any $\sigma \in S_m$:*

1. *For any $S \subseteq S_m$ we have $\sigma(\eta_{l_0}[S]) = \eta_{l_0}[\sigma(S)]$.*
2. *For any $T \subseteq \mathcal{L}(X)$ we have $\sigma(\eta_{l_0}^{-1}[T]) = \eta_{l_0}^{-1}[\sigma(T)]$*

*Proof.* We prove the first part, and then we show how the second follows from that.

1. First we show that for any permutations $\rho, \sigma \in S_m$ it holds that $\eta_{l_0}(\sigma(\rho)) = \sigma(\eta_{l_0}(\rho))$. The LHS gives us $\eta_{l_0}(\sigma(\rho)) = \eta_{l_0}(\sigma \circ \rho) = \sigma \circ \rho(l_0)$ while the RHS gives $\sigma(\eta_{l_0}(\rho)) = \sigma(\rho(l_0))$. This equality is that required of an action, and we have seen that it holds when we defined this action.

Now, given a set $S \subseteq S_m$, let $l \in \eta_{l_0}[\sigma(S)]$, then $l$ is of the form $\eta_{l_0}(\sigma(\rho))$ for some $\rho \in S$, and since $\eta_{l_0}(\sigma(\rho)) = \sigma(\eta_{l_0}(\rho)) \in \sigma(\eta_{l_0}[S])$ we get $\eta_{l_0}[\sigma(S)] \subseteq \sigma(\eta_{l_0}[S])$.

On the other direction, if $l \in \sigma(\eta_{l_0}[S])$, then $l = \sigma(l')$ for some $l' \in \eta_{l_0}[S]$, so there is $\rho \in S$ such that $\eta_{l_0}(\rho) = l'$. We get $l = \sigma(l') = \sigma(\eta_{l_0}(\rho)) = \eta_{l_0}(\sigma(\rho)) \in \eta_{l_0}[\sigma(S)]$, so we get $\sigma(\eta_{l_0}[S]) \subseteq \eta_{l_0}[\sigma(S)]$, which gives us the equality.

2. Let $T \subseteq \mathcal{L}(X)$. We need to show $\sigma(\eta_{l_0}^{-1}[T]) = \eta_{l_0}^{-1}[\sigma(T)]$. Since $\eta_{l_0}$ is a bijection, there is a set $S \subseteq S_m$ such that $\eta_{l_0}[S] = T, \eta_{l_0}^{-1}[T] = S$. From the first part we get:

$$\sigma(T) = \sigma(\eta_{l_0}[S]) = \eta_{l_0}[\sigma(S)] = \eta_{l_0}[\sigma(\eta_{l_0}^{-1}[T])]$$

By operating with $\eta_{l_0}^{-1}$ on both sides ($\eta_{l_0}$ is a bijection so the equality is preserved) we get:

$$\eta_{l_0}^{-1}[\sigma(T)] = \eta_{l_0}^{-1}[\eta_{l_0}[\sigma(\eta_{l_0}^{-1}[T])]] = \sigma(\eta_{l_0}^{-1}[T])$$

□

We now define the set of all partitions of $\mathcal{L}(X)$ induced by neutral and anonymous voting rules for $m$ alternatives. Let

$$B_m = \{\mathcal{M}_{\mathcal{F}_{m,n}} : n \in \mathbb{N}, \mathcal{F}_{m,n} \text{ is some NA-voting rule}\}$$

We say that the elements of $B_m$ are *partitions induced by voting rules*. Recall that for anonymous voting rules, the partitions are identical for all voters (Fact 3.2.4), so the omission of the subscript $i$ is justified. We also define the following set of partitions of $\mathcal{L}(X)$:

$$C_m = \{\mathcal{M} : \mathcal{M} \text{ is a partition of } \mathcal{L}(X) \text{ s.t. for all } M \in \mathcal{M}, \sigma \in S_m \quad \sigma(M) \in \mathcal{M}\}$$

We say that the elements of $C_m$ are *partitions respected by permutations*.

From Fact 5.1.1 we know that $B_m \subseteq C_m$. Our goal in the next section is to establish a connection between $C_m$ and the subgroups of $S_m$, and to characterize the inclusion $B_m \subseteq C_m$.

## 5.3 Subgroups and NA-Rules

We start with the following theorem, which is the seminal theorem of the correspondence:

**Theorem 5.3.1.** *There is a mapping $\varphi : C_m \to \{H \leq S_m\}$ which is one-to-one and onto, and for any $\mathcal{M} \in C_m$ satisfies $\mathcal{M} = \{\eta_{l_0}[C] : C \in S_m/\varphi(\mathcal{M})\}$.*

In words, the theorem says that for any partition of ballots $\mathcal{M} \in C_m$ we can find a subgroup $H$ which induces a partition of permutations (into the set of cosets $S_m/H$), so that the image of $S_m/H$ under $\eta_{l_0}$ is simply $\mathcal{M}$, and that correspondence is a bijection. After we have proved this, we will be able to say that the set $C_m$ is the set of partitions induced by subgroups of $S_m$ (up to $\eta_{l_0}$).

*Proof.* Let $\mathcal{M} \in C_m$ be some partition of $\mathcal{L}(X)$. Let $M \in \mathcal{M}$ be the (unique) equivalence class in this partition which contains $l_0$. We show that $H := \eta_{l_0}^{-1}[M]$ is a subgroup of $S_m$. For this we need to show that $H$ contains the unit element and is closed under the operations of composition and inversion:

- The unit element $e$ is in $H$ simply since we chose $M$ to be such that $l_0 \in M$, and since $\eta_{l_0}(e) = e(l_0) = l_0$, we get that $e \in H$.
- Let $\sigma_1, \sigma_2 \in H$, we need to show that $\sigma_1 \circ \sigma_2 \in H$. We show that in general, for any element $\sigma \in H$ we have $\sigma H = H$. From this it follows immediately that $\sigma_1\sigma_2 \in \sigma_1 H = H$, which is what we want.

  Let $\sigma \in H$. From Lemma 5.2.2 we know that $\eta_{l_0}[\sigma H] = \sigma(\eta_{l_0}[H]) = \sigma M \in \mathcal{M}$, and we know that $e \in H$ and $\sigma(e) = \sigma$, hence $\eta_{l_0}[\sigma H] \cap \eta_{l_0}[H]$ is non empty, and since these are both elements in a partition, they are either disjoint or identical, and therefore $\eta_{l_0}[\sigma H] = \eta_{l_0}[H]$, and since $\eta_{l_0}$ is a bijection, we get that $\sigma H = H$.
- We show that $H$ is closed under inversion. Let $\sigma \in H$. We have shown that $\sigma H = H$. We know that $e \in H$, hence there is $\sigma' \in H$ such that $\sigma\sigma' = e$, and that can only be for $\sigma' = \sigma^{-1}$, and so $H$ is closed under inverse elements.

We can now properly define the mapping $\varphi : C_m \to \{H \leq S_m\}$, given by $\varphi(\mathcal{M}) = \eta_{l_0}^{-1}[M]$ for $\mathcal{M} \in C_m$, where $M \in \mathcal{M}$ is the unique element in partition $\mathcal{M}$ which satisfies $l_0 \in M$.

We verify that $\mathcal{M} = \{\eta_{l_0}[C] : C \in S_m/\varphi(\mathcal{M})\}$. Take $\mathcal{M} \in C_m$ and let $M \in \mathcal{M}$ be the equivalence class containing $l_0$. Take $M' \in \mathcal{M}$. If $l \in M'$ is some linear order, then there is a permutation $\sigma \in S_m$ such that $\sigma(l) = l_0$, and from that it follows that $\sigma M' \cap M \neq \emptyset$, and so $\sigma M' = M$. We get that

$$H = \varphi(\mathcal{M}) = \eta_{l_0}^{-1}[M] = \eta_{l_0}^{-1}[\sigma M'] = \sigma\eta_{l_0}^{-1}[M']$$

(where we used Lemma 5.2.2) and by acting with $\sigma^{-1}$ on the left on both sides we get $\eta_{l_0}^{-1}[M'] = \sigma^{-1}H$, so $M'$ is the image of a coset of $\varphi(\mathcal{M})$, as we claimed.

We show that $\varphi$ is one to one. Let $\mathcal{M}, \mathcal{M}' \in C_m$ be two partitions, and say that $\varphi(\mathcal{M}) = \varphi(\mathcal{M}')$. But then we get

$$\mathcal{M} = \{\eta_{l_0}[C] : C \in S_m/\varphi(\mathcal{M})\} = \{\eta_{l_0}[C] : C \in S_m/\varphi(\mathcal{M}')\} = \mathcal{M}'$$

and so $\varphi$ is injective.

Finally, we show that $\varphi$ is onto. Let $H \leq S_m$. We define $\mathcal{M} = \{\sigma(\eta_{l_0}[H]) : \sigma \in S_m\}$. Lemma 5.2.2 which states that $\sigma(\eta_{l_0}[H]) = \eta_{l_0}[\sigma H]$, combined with

Fact 5.1.5, gives us that this is indeed a partition of $\mathcal{M}$, which is respected by permutations from the definition, and so $\mathcal{M} \in C_m$. Let $M = e(\eta_{l_0}[H]) = \eta_{l_0}[H]$, then $\varphi(\mathcal{M}) = H$, and so $\varphi$ is onto. □

We already saw that $B_m \subseteq C_m$, and now we have established a correspondence between $C_m$ and subgroups of $S_m$ (in fact, we saw that the elements of $C_m$ are partitions induced by subgroups, up to $\eta_{l_0}$). That means that instead of talking of a partition induced by an NA-voting rule, we can also talk about the corresponding subgroup, or the corresponding partition induced by the subgroup. We can do this, even though the first partition is a partition of ballots and the second one is of permutations, since we saw that these sets are isomorphic, and our operations on them commute. Indeed, in the rest of the chapter we will not always explicitly mention the function $\eta_{l_0}$, even when it is used, in the hope that the reader can make out the translation from the context.

We remark that while Theorem 5.3.1 implies that for every NA-voting rule there is a subgroup which induces that partition, we do not yet know if for any subgroup there is an NA-voting rule which induces the same partition as the subgroup. The answer to that question, in fact, will be no (in terms of the sets we defined, $B_m \subsetneq C_m$). The next two sections are devoted to charactering which subgroups induce partitions corresponding to NA-rules, and which do not. That characterization will give us an exact description of the partitions that can be induced by voting rules – a description that we will use in the last section, to prove Sato's claim.

Before we continue to the characterization, however, we give an example which explains why not every (partition induced by a) subgroup can correspond to (a partition induced by) some NA-voting rule.

We take the case $m = 4$. We show that there is a subgroup of $S_4$ that the partition it induces cannot be induced by an NA-voting rule. The subgroup we look at is the alternating group, denoted $A_4$, which consists of all the permutations of 4 elements which contain an even number of cycles (the subgroup has index 2, i.e., it has only one coset which is not the subgroup itself). Table 5.1 shows the subgroup and the corresponding orders, along with its other coset and the orders corresponding to it.

Assume by contraposition that there is such an NA-rule $\mathcal{F}_{4,n}$, which induces the same partition as $A_4$. That is, $\mathcal{M}_{\mathcal{F}_{4,n}}$ has two sets: $M_1 = \eta_{l_0}[A_4]$ and $M_2 = \eta_{l_0}[S_4 \setminus A_4]$. $\mathcal{F}_{4,n}$ is not constant,[3] and so there is a profile $P = \langle V_1, \ldots, V_n \rangle$ such that $\mathcal{F}(P) = Y \neq X$. Since $A_4$ is symmetric with respect to all four elements of $X$, we can assume w.l.o.g that $1 \in Y$ and $2 \notin Y$.

Take the permutation $\sigma = (12)(34) \in A_4$. We get that $\sigma Y \neq Y$. However, since $\sigma \in A_4$, we claim that for any ballot $V_i$, $\sigma V_i \sim V_i$. If $V_i \in \eta_{l_0}[\rho H]$, where $\rho H$ represents either $A_4$ or its complement, we get that $\sigma V_i \in \sigma(\eta_{l_0}[\rho H]) = \eta_{l_0}[\sigma(\rho H)]$. If $\rho H = A_4$, then since $\sigma \in A_4$, we also get $\sigma(\rho H) = A_4$. If $\rho H$ is

[3] There is a unique partition induced by constant rules: the partition into one equivalence class (i.e., $\mathcal{M} = \{M\}$ where $M = \mathcal{L}(X)$), and that partition corresponds to the to the partition induced by $S_m$ itself as a subgroup of $S_m$, which is not the case here. In addition, we are considering only proper subgroups/non-constant rules.

| # | elements of $A_4$ | $\eta_{l_0}[A_4]$ | elements of $S_4 \setminus A_4$ | $\eta_{l_0}[S_4 \setminus A_4]$ |
|---|---|---|---|---|
| 1 | () | 1234 | (12) | 2134 |
| 2 | (12)(34) | 2143 | (13) | 3214 |
| 3 | (13)(24) | 3412 | (14) | 4231 |
| 4 | (14)(23) | 4321 | (23) | 1324 |
| 5 | (123) | 2314 | (24) | 1432 |
| 6 | (132) | 3124 | (34) | 1243 |
| 7 | (134) | 3241 | (1234) | 2341 |
| 8 | (143) | 4213 | (1432) | 4123 |
| 9 | (124) | 2431 | (1243) | 2413 |
| 10 | (142) | 4132 | (1342) | 3142 |
| 11 | (234) | 1342 | (1423) | 4312 |
| 12 | (243) | 1423 | (1324) | 3421 |

Table 5.1: The alternating group $A_4$, a subgroup of $S_4$, induces a partition of the linear orders on $\{1, 2, 3, 4\}$ into two equivalence classes (for cycle notation, see 5.1.19)

the complement, which is all odd permutations, then multiplying by $\sigma$, which is even, will keep us in the same coset of odd permutations. One way or another we get $\sigma V_i \in \eta_{l_0}[\rho H]$, just like $V_i$. This means that both ballots are in the same equivalence class, and therefore equivalent, as we have claimed.

We now use Proposition 3.2.1 to get that $P \sim \sigma P$, but this is a direct contradiction to neutrality, which requires that for any permutation, and especially for our $\sigma$, we have $\sigma \mathcal{F}_{4,n}(P) = \mathcal{F}_{4,n}(\sigma P)$, but the RHS should give us $Y$, since $P \sim \sigma P$, and we saw that the LHS is $\sigma Y \neq Y$, so neutrality is violated, and there can be no such rule $\mathcal{F}_{4,n}$.

In the next section we show that no subgroups of index less than $m$ can correspond to the partition of an NA-voting rule (though, as it will follow, there are very few such groups), and in the following section we will show that all subgroups of index greater or equal $m$ can correspond to partitions of NA-voting rule, thus we will have the complete characterization of partitions induced by NA-voting rules.

### 5.3.1 Subgroups of Index $< m$

As was mentioned above, in this section we show that no subgroup $H < S_m$ of index $< m$ can induce a partition which corresponds to that of an NA-voting rule. We will first show that for any $m \geq 5$ the only proper non-trivial subgroup of index less than $m$ is $A_m$, the alternating group, which was mentioned before, and whose example we have encountered in the case $m = 4$. We will then generalize the proof for the case $A_4$, to show that $A_m$ never corresponds to a partition induced by an NA-voting rule.

The other cases, where $m \leq 4$, will be handled by "brute force", since these groups are small, and the general proof does not work for them, whether it is

because they have more subgroups of small indices (as in the case of $S_4$) or because they do not have enough elements, as required by the general proof (as in the case of $S_3$).

The following lemma could be taken as a theorem of group theory, yet since some of the principles used to prove it are used in other places in the chapter, we bring the proof.

**Lemma 5.3.1.1.** *For $m \geq 5$, the only proper subgroup $H < S_m$ of index $< m$ is $A_m$, the alternating group.*

*Proof.* Let $H < S_m$ be a subgroup of index $k < m$, $m \geq 5$. We look at the action of $S_m$ on left cosets of $H$, which we described before ($\sigma(\sigma' H) = (\sigma\sigma')H$). Let $\rho : S_m \to Perm(S_m/H)$ be the homomorphism associated with the action. We know that $|S_m/H| = k < m$, hence $\rho$ is a function from a domain of size $m!$ to a codomain of size $k! < m!$. This means, from the pigeonhole principle, that $\rho$ cannot be one-to-one, so there are two different permutations, $\sigma_1, \sigma_2 \in S_m$ such that $\sigma_1(\sigma H) = \sigma_2(\sigma H)$ for every $\sigma \in S_m$. We multiply both sides of the equation by $\sigma_1^{-1}$ to the left, and we get: $\sigma H = \sigma_1^{-1}\sigma_2(\sigma H)$ for any $\sigma \in S_m$.

But this means, from the definition, that $\sigma_1^{-1}\sigma_2$ is in the kernel of $\rho$, and since $\sigma_1 \neq \sigma_2$, we also get that $\sigma_1^{-1}\sigma_2 \neq e$, so the kernel is non-trivial. We use Fact 5.1.13 to conclude that $H$ contains some non-trivial normal subgroup of $S_m$, but according to Fact 5.1.24 that subgroup can only be $A_m$ or $S_m$ itself. If it is $S_m$, then $H$ is not proper, in contradiction to our assumption, so $A_m \subseteq H$. This inclusion actually implies $A_m = H$, since the order of a subgroup must be a divisor of the order of the group (Fact 5.1.6). □

We now use this to prove the claim for the correspondence:

**Theorem 5.3.1.2.** *If $H < S_m$ is of index $< m$, then there is no NA-voting rule $\mathcal{F}$ such that $\mathcal{M}_{\mathcal{F}} = \{\eta_{l_0}[C] : C \in S_m/H\}$*

*Proof.* We divide the proof into two cases: the case $m < 5$, where we prove it based on the known subgroup structure of those small groups, and the case $m \geq 5$, where we present the general proof.

**The case $m < 5$:** Since we are only considering cases where we actually have several different alternatives, we need only consider the cases $2 \leq m \leq 4$. For the case $m = 2$, if $H < S_2$ with index $< m$, then the index of $H$ is 1, i.e., $H = S_2$, which corresponds to the constant rule, which we are not considering. We base the proof for the cases $m = 3$ and $m = 4$ on facts which are known about the subgroup structure of $S_3$ and $S_4$.

The case $m = 3$ is simple, since there are only six possible orders, and there is only one subgroup of index 2, $A_3$, so any partition into two equivalence classes must correspond to the partition induced by $A_3$, containing the permutations $(), (123), (132)$. Assume we have some NA-rule $\mathcal{F}_{3,n}$ inducing the partition of $A_3$ and $S_3 \setminus A_3$, and a profile $P = \langle V_1, \ldots, V_n \rangle$ such that $\mathcal{F}_{3,n}(P) = Y \neq \{1, 2, 3\}$ (otherwise the rule is constant). We

can assume w.l.o.g that $1 \in Y, 2 \notin Y$ (the subgroup and corresponding partitions are symmetric with respect to the three elements), but then we see that the permutation $(123)Y \neq Y$, and the permutation $(123)$ is in $A_3$, so like we saw in the case of $A_4$, $V \sim (123)V$ for any ballot $V$, and then we get $\mathcal{F}_{3,n}((123)P) = \mathcal{F}_{3,n}(P) = Y \neq (123)Y = (123)\mathcal{F}_{3,n}(P)$, and that is in contradiction to neutrality.

For the case $m = 4$, we have already covered the partition corresponding to $A_4$ in the example. However there are still three more conjugate subgroups of index 3 (8 elements in each subgroup), which are isomorphic to the Dihedral group (the group of all symmetries of a square), and are referred to as $D8$ in $S_4$. Each of these subgroups contains the normal Klein four-group, which is the additional normal subgroup of $S_4$ that we have mentioned in Fact 5.1.24. Since, as we saw, the elements of this subgroup fix all the equivalence classes, we get that the same proof for $A_4$ also works in this case (the Klein-four group is a subset of $A_4$), and so no NA-rule corresponds to subgroups of index 3, which concludes the case $m = 4$.

**The case $m \geq 5$:** From Lemma 5.3.1.1 we know that our $H$ is $A_m$. We show that $A_m$ cannot correspond to a partition induced by an NA-voting rule. The proof is of the same form as that of $A_4$ in the example, but we prove it fully here and in a slightly more formal way.

Assume by contraposition that the partition induced by $H$ does correspond to a partition of some NA-rule $\mathcal{F}_{m,n}$, then there is some profile $P = \langle V_1, \ldots, V_n \rangle$ such that $\mathcal{F}_{m,n}(P) \neq X$. Let $x \in \mathcal{F}(P)$ and $y \notin \mathcal{F}(P)$. We can take the permutation $\sigma = (xy)(wz)$ where $w, z \in X$ are two distinct elements, pairwise different from $x, y$ (for any $m \geq 4$ we can find such $w, z$). This permutation is expressed as a product of two transpositions, and is therefore an element of $A_m$. Let us see what happens when we act with $\sigma$ on the profile $P$. If a ballot $V_i$ corresponds to a permutation $\rho \in A_m$, then $\sigma V_i = \sigma\eta_{l_0}(\rho) = \eta_{l_0}(\sigma\rho)$, and since $\sigma\rho \in A_m$ (multiplying two even permutations), we get $V_i \sim \sigma V_i$. If, on the other hand, $V_i$ corresponds to $\rho \in S_m \setminus A_m$, then $\sigma\rho$ also belongs to the complement (multiplying an even permutation with an odd one), and the resulting ballot is still equivalent to the original ballot $V_i$. We found that for each $V_i$ we have $\sigma V_i \sim V_i$, which means the entire profile $P$ is equivalent to $\sigma P$, which in turn means (Proposition 3.2.1) $\mathcal{F}(P) = \mathcal{F}(\sigma P)$.

However, from the way we defined $\sigma$ we know that $\sigma$ swaps a winning alternative (namely $x$) with a losing alternative (namely $y$), so $\mathcal{F}(P) \neq \sigma(\mathcal{F}(P))$. Combining this with the previous result, we get $\mathcal{F}(\sigma P) \neq \sigma(\mathcal{F}(P))$, in contradiction to $\mathcal{F}$ being neutral, so $H$ has no corresponding (partition induced by an) NA-voting rule $\mathcal{F}$. □

We now proceed to some more "positive" results, with subgroups of index greater or equal to $m$.

### 5.3.2 Subgroups of Index $\geq m$

Unlike the previous section, where we saw that none of the aforementioned subgroups correspond to a partition of an NA-voting rule, we will see in this section that *all* subgroups $H < S_m$ of index $\geq m$ correspond to partitions of voting rules. We do this by taking an arbitrary subgroup $H$ with index $\geq m$ and constructing a voting rule that induces the same partition as the $H$. At first we construct just a neutral voting rule, where the partition of all the voters is identical to that of the subgroup, and then we show how we can use that rule to generate a new rule which is both neutral and anonymous, and induces the same partition.

While the rules we construct will not necessarily make any "sense", when comparing to standard rules in voting theory, we remember that we are looking at the entire set of non-constant NA-rules. The requirement of "sense" is usually expressed by additional axiomatic requirements such as monotonicity, Pareto etc.

We start by first constructing a rule which is just neutral:

**Lemma 5.3.2.1.** *Every subgroup of $S_m$ of index $\geq m$ corresponds to a partition induced by some N-voting rule, where all the voters have the same partition.*

*Proof.* We prove this by taking an arbitrary subgroup $H \leq S_m$ of index $k \geq m$, and constructing a voting rule which induces for all voters a partition of $\mathcal{L}(X)$ which corresponds to the partition given by $S_m/H$.

Let $H \leq S_m$ of index $k \geq m$, let $C_1, \ldots, C_k$ be an enumeration of the cosets of $H$, and let $D_1, \ldots, D_k$ be the respective sets of linear orders (that is $D_i = \eta_{l_0}[C_i]$). We define a voting rule $\mathcal{F}_{m,k}$ such that $\mathcal{M}_{i,\mathcal{F}_{m,k}} = \{D_1, \ldots, D_k\}$ for every voter $1 \leq i \leq k$.

We need to define $\mathcal{F}_{m,k}(P)$ for any profile $P = \langle V_1, \ldots, V_k \rangle$. By defining the rule not for every specific ballots $V_1, \ldots, V_k$, but rather for arbitrary representatives of the sets $D_1, \ldots, D_k$, we guarantee that the rule "cannot tell between ballots in the same coset", and the partition induced by the rule will be at least as coarse as that of $\{D_1, \ldots, D_k\}$.

Slightly more formal, let $V_1, \ldots, V_k$ be representatives of the $\{D_1, \ldots, D_k\}$, such that $V_i \in D_i$. We define the rule for all and only the profiles composed of these ballots (possibly with repetitions), and for any profile $P = \langle V'_1, \ldots, V'_l \rangle$ we set $\mathcal{F}_{m,k}(P) = \mathcal{F}_{m,k}(\langle V_{i_1}, \ldots, V_{i_k} \rangle)$, where $V'_j \in D_{i_j}$. This gives us a well defined rule $\mathcal{F}_{m,k}$, and it guarantees that $V, V' \in D_i$ implies $V \sim_j V'$ for any $1 \leq i, j \leq k$.

So we just have to define the rule in a way that would make it neutral, and would guarantee that the partition it induces is as fine as $D_1, \ldots, D_k$. We define the rule in stages, so that the rule we end up with satisfies these requirements.

Let $A_1 = \{P_1, \ldots, P_{k!}\}$ be the set of all the profiles that contain *all* the ballots $V_1, \ldots, V_k$ in different orders (there are $k!$ such profiles), and assume these profiles are enumerated according to some order. Let $i_1 = 1$. We first define $\mathcal{F}_{m,k}(P_{i_1}) = 1$. Next, for any $\sigma \in S_m$ we define $\mathcal{F}_{m,k}(\sigma P_{i_1}) = \sigma(1)$. We note that due to the fact that $m \leq k$, from Fact 5.1.13 and from what we have

seen in the previous section about the existence of normal subgroups in $S_m$, we get that the action is faithful, which means that $\sigma_1 P_{i_1} = \sigma_2 P_{i_1}$ iff $\sigma_1 = \sigma_2$ (so this is well defined). By now we have defined $\mathcal{F}_{m,k}$ for a set of size $m!$ of profiles in $A_1$.

We next define $A_2 = A_1 \setminus \{\sigma P_{i_1} : \sigma \in S_m\}$. We now take the first profile in $A_2$, say $P_{i_2}$, and again we define $\mathcal{F}_{m,k}(P_{i_2}) = 1$, and for any $\sigma \in S_m$: $\mathcal{F}_{m,k}(\sigma P) = \sigma(1)$. Again this gives us a set of new $m!$ profiles, after which we define $A_3$ etc. In general, in the $j+1$th stage we define $A_{j+1} = A_j \setminus \{\sigma P_{i_j} : \sigma \in S_m\}$, we choose $P_{i_{j+1}}$ to be the first profile in $A_{j+1}$, we define $\mathcal{F}_{m,k}(P_{i_{j+1}}) = 1$ and $\mathcal{F}_{m,k}(\sigma P_{i_{j+1}}) = \sigma(1)$ for any $\sigma \in S_m$. The process is bound to halt after we exhaust our set of profiles, which will happen after $\frac{k!}{m!}$ steps.[4]

For any profile $P$ that wasn't in the original set $A_1$ (we notice that these are all the profiles where there is at least one repetition of one ballot) we define $\mathcal{F}_{m,k}(P) = X$. This defines our rule for all the profiles composed of the representatives of $D_1, \ldots, D_k$, and so for all profiles.

We show that this rule is neutral: Let $P = \langle V_1, \ldots, V_k \rangle$ be a profile, $\sigma$ a permutation, we need to show that $\mathcal{F}_{m,k}(\sigma P) = \sigma \mathcal{F}_{m,k}(P)$. There are two possible cases:

- If $P \notin A_1$, then there are ballots $V_{j_1}, V_{j_2}$ in $P$ such that $V_{j_1}, V_{j_2} \in D_l$ for some $l$. For such a profile we have defined $\mathcal{F}_{m,k}(P) = X$. We can see that in $\sigma P$ we also have $\sigma V_{j_1}, \sigma V_{j_2} \in \sigma D_l$, so we get $\mathcal{F}_{m,k}(\sigma P) = X$, and so $\mathcal{F}_{m,k}(\sigma P) = X = \sigma X = \sigma \mathcal{F}_{m,k}(P)$.

- If $P \in A_1$, then there is some profile $P_{i_j}$ (one of the profiles selected during the definition of $\mathcal{F}_{m,k}$) and a permutation $\rho$ such that $P = \rho P_{i_j}$. We then get
$$\mathcal{F}_{m,k}(\sigma P) = \mathcal{F}_{m,k}(\sigma(\rho P_{i_j})) = \mathcal{F}_{m,k}((\sigma\rho) P_{i_j}) = (\sigma\rho)(1)$$
And also
$$\sigma \mathcal{F}_{m,k}(P) = \sigma \mathcal{F}_{m,k}(\rho P_{i_j}) = \sigma(\rho 1) = (\sigma\rho)(1)$$
Where in both cases we used the definition of the rule as well as the associativity of action, and again we have neutrality.

We now need to show that the partition induced by $\mathcal{F}_{m,k}$ is as fine as $\{D_1, \ldots, D_k\}$ for all the voters. W.l.o.g we show that for the first voter, given $V \in D_1, V' \in D_2$, there is some profile $P = \langle V, U_2, \ldots, U_k \rangle$, such that $\mathcal{F}_{m,k}(P) \neq \mathcal{F}_{m,k}(\langle V', U_2, \ldots, U_k \rangle)$. The proof for the other sets of the partition and the other voters is identical.

We take the profile $P = \langle V, V_2, \ldots, V_k \rangle$, where $V_i \in D_i$ for $2 \leq i \leq k$. We know that $P \in A_1$ (up to representatives), since there are no two ballots of the same equivalence set $D_i$, so $\mathcal{F}_{m,k}(P)$ is a singleton. On the other hand, $P' = \langle V', V_2, \ldots, V_k \rangle$ is not in $A_1$, since both $V', V_2$ are of the same equivalence set ($D_2$), and so $\mathcal{F}_{m,k}(P') = X$, and so we have $\mathcal{F}_{m,k}(P) \neq \mathcal{F}_{m,k}(\langle V', U_2, \ldots, U_k \rangle)$, as we required. □

[4] In group-theoretic terms, on every step we define $\mathcal{F}_{m,k}$ for a new orbit.

We strengthen the previous result by showing that the rule above can actually be made to be anonymous. We notice that while from Fact 3.2.4 we know that for any anonymous rule, the partition of $\mathcal{L}(X)$ is identical for all the voters, the converse does not necessarily hold, i.e., we could have a rule which induces the same partition of $\mathcal{L}(X)$ for all voters, yet it is not anonymous. However, the correspondence we are constructing is for partitions of NA-rules, and the rule we built in Lemma 5.3.2.1 is not necessarily anonymous. To overcome this problem we present the following lemma:

**Lemma 5.3.2.2.** *For every N-voting rule $\mathcal{F}_{m,n}$, if there is a partition $\mathcal{M}$ such that for every voters $i$ we have $\mathcal{M}_{i,\mathcal{F}_{m,n}} = \mathcal{M}$, then there exists an NA-voting rule $\mathcal{F}_{m,2^n-1}$ which also induces the partition $\mathcal{M}$.*

*Proof.* The idea of the proof is to "code" the $i$th voter using $2^{i-1}$ anonymous voters. Let $\mathcal{M} = \{M_1, \ldots, M_k\}$ be the partition induced by $\mathcal{F}_{m,n}$. We define a function $\Phi : \mathcal{L}(X)^{2^n-1} \to \mathcal{L}(X)^n \cup \{\emptyset\}$, i.e., a function from profiles of $2^n - 1$ voters to profiles of $n$ voters (or the empty set). The function will work like this: If for a profile $P$ of $2^n - 1$ voters we can find a profile $P'$ of $n$ voters such that $2^{i-1}$ distinct voters in the big profile voted like the $i$th voter in the small profile, then $\Phi(P) = P'$, otherwise the image is the empty set.

More formally: Given a profile $P = \langle V_1, \ldots, V_{2^n-1}\rangle$, if there are $M_{j_1}, \ldots, M_{j_n} \in \{M_1, \ldots, M_k\}$ (not necessarily distinct) sets, and a partition of $\{V_1, \ldots, V_{2^n-1}\}$ into sets $A_1, \ldots, A_n$, such that for every $1 \leq i \leq n$, $|A_i| = 2^{i-1}$ and all the ballots in $A_i$ belong to $M_{j_i}$, then $\Phi(P) = \langle U_1, \ldots, U_n\rangle$ where $U_i \in M_{j_i}$. Otherwise $\Phi(P) = \emptyset$. We note that this partition of $\{V_1, \ldots, V_{2^n-1}\}$, if it exists, is unique (since we can uniquely determine how many voters voted for a specific equivalence class, and since any number has a unique representation in binary).

We now define the rule: For any $P = \langle V_1, \ldots, V_{2^n-1}\rangle$ we define $\mathcal{F}_{m,2^n-1}(P) = \mathcal{F}_{m,n}(\Phi(P))$ if $\Phi(P) \neq \emptyset$, and when $\Phi(P) = \emptyset$ we define $\mathcal{F}_{m,2^n-1}(P) = X$.

We need to show that this rule is anonymous, neutral, and induces the same partition $\mathcal{M}$. Anonymity is immediate from the way the rule is defined. Regarding neutrality, let $\sigma \in S_m$ be a permutation (acting on both profiles of $2^n - 1$ voters and $n$ voters), and $P$ a profile. We notice that if $\Phi(P) = \emptyset$, then also $\Phi(\sigma P) = \emptyset$ (as the partition $\mathcal{M}$ is respected by permutations), so in such a case neutrality holds.

If, on the other hand, $\Phi(P) \neq \emptyset$ for some $n$ voter profile, then also $\Phi(\sigma P) \neq \emptyset$ (again, $\mathcal{M}$ is respected by permutations ), it is enough to show that $\Phi(\sigma P) = \sigma\Phi(P)$, and then from the neutrality of $\mathcal{F}_{m,n}$ we will get:

$$\mathcal{F}_{m,2^n-1}(\sigma P) = \mathcal{F}_{m,n}(\Phi(\sigma P)) = \mathcal{F}_{m,n}(\sigma\Phi(P)) = \sigma\mathcal{F}_{m,n}(\Phi(P)) = \sigma\mathcal{F}_{m,2^n-1}(P)$$

And $\Phi(\sigma P) = \sigma\Phi(P)$ follows again from the fact that $\mathcal{M}$ is respected by permutations, so if $\Phi(P) = \langle U_1, \ldots, U_n\rangle$ where $U_i \in M_{j_i}$, then $\Phi(\sigma P) = \langle U'_1, \ldots, U'_n\rangle$ where $U_i \in \sigma M_{j_i}$, which is just $\sigma\Phi(P)$.

Last, we have to show that $\mathcal{M}_{\mathcal{F}_{m,2^n-1}} = \mathcal{M}$. From the same arguments we have seen before, the partition is at least as coarse as $\mathcal{M}$. We show that voter 1 can tell between the sets $M_1, M_2$ (the proof is identical for all voters

and sets). Since $\mathcal{F}_{m,n}$ tells the sets apart with respect to any voter, there is an $n$ voter profile $P_1 = \langle V_1, \ldots, V_n \rangle$, $V_1 \in M_1$ and a ballot $V_1' \in M_2$ such that $\mathcal{F}_{m,n}(P_1) \neq \mathcal{F}_{m,n}(P_1')$ where $P_1' = \langle V_1', V_2, \ldots, V_n \rangle$. We take the $2^n - 1$ voter profile $P_2 = \langle V_1, V_2, V_2, V_3, V_3, V_3, V_3, V_4, \ldots, V_n \rangle$, where for each $V_i$ there are $2^{i-1}$ voters choosing that ballot. We get that $\Phi(P_2) = P_1$. Now, if voter 1 switches to $V_1'$ we get a new $2^n - 1$ voter profile $P_2'$, for which $\Phi(P_2') = P_1'$, and so $\mathcal{F}_{m,2^n-1}(P_2) \neq \mathcal{F}_{m,2^n-1}(P_2')$. □

We now combine our two lemma to get the theorem we wanted for our correspondence:

**Theorem 5.3.2.3.** *Every subgroup of $S_m$ of index $\geq m$ corresponds to a partition induced by some NA-voting rule.*

*Proof.* Given $H \leq S_m$ of index $k \geq m$, we take the neutral rule defined in Lemma 5.3.2.1, $\mathcal{F}_{m,k}$ (where the partition is $S_m/H$ for all voters), and transform it into an anonymous rule $\mathcal{F}_{m,2^k-1}$, giving us the required result. □

## 5.4 Using the Correspondence

In his paper, Sato [14] shows how when a neutral and anonymous voting rule is operating on minimal informational requirements, then it must have informational size $mn$, and in addition it "fixes" one of the coordinates in ballots, i.e.:

$$\exists 1 \leq r \leq m \text{ s.t. } \forall V_1, V_2 \quad V_1 \sim V_2 \Leftrightarrow V_1^r = V_2^r \tag{5.1}$$

(Lemma 4.3 in the paper). However, the proof in the paper contains an small error [15].[5]

Sato solved this problem by adding an additional restriction on the space of voting rules: We consider only voting rules that, given a profile where all the voters submit the same ballot, do not elect the entire set of alternatives ($\mathcal{F}(P) \neq X$ for $P = \langle V, V, \ldots, V \rangle$). While this move is indeed enough to prove this claim [16], the restriction is sufficient but not necessary, as we see in the following example:

We give an example of an NA-voting rule which operates on minimal informational requirements, and obeys the conditions of the lemma, though it does not satisfy the additional restriction. Let $\mathcal{F}$ be a rule which is identical to the plurality rule for any profile $P$, except where $P$ is composed only of identical ballots, in which case $\mathcal{F}(P) = X$. This rule is neutral, anonymous and it operates on minimal informational requirements, though it does not satisfy the restriction (indeed, it was built to violate it).

In the following section we prove the claim without the additional restriction (for any number of alternatives $m$ except for $m = 6$, where we show it to be

[5] In Claim 2 in his proof it is assumed that for any two profiles of $n$ voters, there exists a permutation $\sigma$ of $X$ which can be used to get from the one profile to the other. While this is true for any ballot, it is not always the case for profiles.

wrong), by using the correspondence between partitions induced by NA-voting rules (for $m$ alternatives) and subgroups of $S_m$.

As we have seen, we can identify the partition of ballots $\mathcal{M}_{\mathcal{F}_{m,n}}$ induced by an NA-rule $\mathcal{F}_{m,n}$ with the partition of $S_m$ induced by a subgroup $H < S_m$ (Theorem 5.3.1). It follows that the informational size of $\mathcal{F}_{m,n}$ is $n|\mathcal{M}_{\mathcal{F}_{m,n}}| = n|S_m/H|$, where $|S_m/H|$ is simply the index of $H$ in $S_m$. That means that if we know which subgroup corresponds to the partition of the rule, we know the informational size of the rule, and so claims about informational size can be translated to claims about the existence of subgroups of a certain index, which is precisely what we will do.

The critical claim, based on Sato's paper, is the following:

**Theorem 5.4.1.** *(adapted from [14, Theorem 3.1]) For any neutral and anonymous voting rule operating on minimal informational requirements, the following holds:*

1. *The rule has informational size $nm$.*
2. *In addition, whenever the number of alternatives $m$ satisfies $m \neq 6$, there exists $1 \leq r \leq m$ such that any two ballots $V_1, V_2$ are equivalent iff $V_1^r = V_2^r$.*

The first part of the theorem will follow more or less directly from our correspondence. For the second part, however, we have to understand what that claim means in group-theoretic terms. Recall the definition of a stabilizer from the beginning of the chapter. We claim that part 2 of Theorem 5.4.1 can be reduced to the requirement that the partition induced by the rule corresponds to that induced by a stabilizer of some $x \in X$. We formalize this in the following lemma:

**Lemma 5.4.2.** *Let $\mathcal{F}_{m,n}$ be an NA-rule operating on minimal informational requirements, $m \neq 6$. $\mathcal{F}_{m,n}$ satisfies condition 2 of Theorem 5.4.1 iff the partition induced by $\mathcal{F}_{m,n}$ corresponds to that of $stab_{S_m}(x)$, the stabilizer subgroup of $x \in X$, where $x$ is the $r$th element in $l_0$.*

*Proof.* ($\Rightarrow$) Let $H$ be the subgroup corresponding to the partition induced by a rule satisfying part 2 of the theorem, we need to find an element of $X$ which is fixed by $H$. Take the $r$ from part 2 in the theorem, and take $x$ to be $l_0^r$. Any two ballots $V_1, V_2$ are equivalent iff $V_1^r = V_2^r$. Specifically, any ballot $V$ is equivalent to $l_0$ (the order on which we based the correspondence in 5.2.1) iff $V^r = l_0^r = x$. Any such $V$ corresponds to a permutation $\sigma$ satisfying $\sigma x = x$, and since $H = \{\eta_{l_0}^{-1}(V) : V^r \sim l_0^r\}$ ($\eta_{l_0}$ being the correspondence function from Proposition 5.2.1), we get that $H = \{\sigma \in S_m : \sigma x = x\}$, which is exactly $stab_{S_m}(x)$.

($\Leftarrow$) On the other hand, assume a rule $\mathcal{F}_{m,n}$ satisfies the conditions of 5.4.1, and its induced partition corresponds to a subgroup $stab_{S_m}(x)$ for some $x \in X$. Let $r$ be the number such that $l_0^r = x$. We show that two ballots $V_1, V_2$ are equivalent iff $V_1^r = V_2^r$.

First assume they are equivalent, then they belong in the same coset $\sigma stab_{S_m}(x)$. Every element $\rho \in \sigma stab_{S_m}(x)$ is of the form $\sigma\rho'$ for some $\rho' \in stab_{S_m}(x)$, so $\rho x = \sigma\rho' x = \sigma x$. Let $\sigma_1, \sigma_2 \in \sigma stab_{S_m}(x)$ be the permutations corresponding to $V_1, V_2$ respectively, then $\sigma_1 x = \sigma x = \sigma_2 x$, and so the $r$th element in both $V_1, V_2$ is equivalent to $\sigma x$, and we have $V_1^r = V_2^r$.

In the other direction, assume $V_1^r = V_2^r$, we need to show that the profiles are equivalent. Let $\sigma_1, \sigma_2$ be the permutations corresponding to $V_1, V_2$. It is enough to show that $\sigma_2 \in \sigma_1 stab_{S_m}(x)$. We know that $\sigma_1 x = \sigma_2 x = y$ for some $y \in X$, so we need a permutation $\rho \in stab_{S_m}(x)$ such that $\sigma_1\rho = \sigma_2$. We multiply by $\sigma_1^{-1}$ on the left on both sides of the equation, and we get $\rho = \sigma_1^{-1}\sigma_2$, and we can see that $\sigma_1^{-1}\sigma_2 x = x$, since $\sigma_2 x = y$ and $\sigma_1^{-1} y = x$, so $\rho \in stab_{S_m}(x)$ like we required, so $\sigma_2 \in \sigma_1 stab_{S_m}(x)$, $V_1$ is equivalent to $V_2$. □

We now proceed to the proof of Theorem 5.4.1.

*Proof.* We start by proving part 1 in the theorem for every possible $m$, and then prove the second part for $m \neq 6$.

1. Let $\mathcal{F}$ be an NA-rule of minimal informational requirements, then we know from Theorem 5.3.1 that there is some subgroup $H < S_m$ which corresponds with $\mathcal{F}$, and which induces the same partition (up to $\eta_{l_0}$).

   As we saw, the informational size of $\mathcal{F}$ is $nk$, where $k$ is the index of $H$. However, from Theorem 5.3.1.2 we know that $k$ cannot be less than $m$, so the informational size has to be $\geq nm$. Since there are rules of informational size $nm$ (e.g., plurality, veto), it follows that it is really equal to $nm$. This proves the first part of the theorem.

2. To complete the claim from the paper we need to show that for any partition $\mathcal{M} \in B_m$ which divides $\mathcal{L}(X)$ into $m$ equivalence classes, there exists $1 \leq r \leq m$ such that any two ballots $V_1, V_2$ are equivalent iff $V_1^r = V_2^r$. As we have seen in Lemma 5.4.2, it is enough to require that the partition induced by $\mathcal{F}_{m,n}$ corresponds to that of $stab_{S_m}(x)$ for some $x \in X$. This will be a direct result of our correspondence, according to which each partition of an NA-voting rule corresponds to some partition, combined with a theorem from group theory describing the subgroups of index $m$ in $S_m$. The theorem we need is:

   **Theorem 5.4.3** (Adapted from Theorem 3.2.19 in [17])**.** *If $X$ is the set $\{1, \ldots, m\}$, $H < S_m$ of index $m$, the following holds:*

   *(a) If $m \neq 6$, then $H$ is conjugate to $stab_{S_m}(1)$, and it is the stabilizer of some $x \in X$.*

   *(b) If $m = 6$, then there are 6 conjugate subgroups of index 6 which are stabilizers of elements in $X$, and 6 conjugate subgroups of index 6 which do not stabilize any element.*

We know that an NA-rule operating on minimal requirements has informational size $nm$, so a rule $\mathcal{F}$ operating on minimal informational requirements corresponds to $H < S_m$ of index $m$. From the theorem above we know that when $m \neq 6$, every subgroup of index $m$ is of the form $stab_{S_m}(x)$ for some $x \in X$, and we know from Lemma 5.4.2 that this is equivalent to part 2 in 5.4.1. This concludes the proof. □

Finally, we show that part 2 is indeed false for the case $m = 6$. This is also a direct result of our correspondence (this time using Theorem 5.3.2.3), along with Theorem 5.4.3, which says that for the case $m = 6$ we actually have subgroups of index 6 which are not stabilizers of any point.

Indeed, take a subgroup $H < S_6$ of index 6 which does not fix any $x \in X$ (see [10] for a description of such a subgroup). From Theorem 5.3.2.3 there is an NA-voting rule, $\mathcal{F}_{6,63}$ which induces the same partition as that of $H$ and its cosets. This voting rule operates on minimal informational requirements, yet it violates part 2 in Theorem 5.4.1.

# Conclusions

Let us start by summarizing what we have done in the thesis. Our starting point was the search for informative representations for voting rules. The way we chose to search for such representations was by looking at how voting rules deal with information. For that we presented in Chapters 2 and 3 three different notions from the literature, that are related to this issue: compilation complexity, communication complexity and informational size. We showed that informational size constitutes an upper bound for both other measures, though a comparison between the two complexities showed us that for many rules they behave quite differently, which meant that we cannot link them directly.

In Chapter 4 we focused more on descriptions related to compilation complexity. We defined a notion of representation of preferences for a voting rule (which we showed to be linked to compilation functions) and of a sufficient and necessary representations. We gave examples of several representations which are sufficient for many common rules, and also necessary for many of those. We discussed the properties of some of those representations, such as WSM, where we gave a profile-independent characterization, and used it to compute an upper bound for STV. Later we saw that we can build hierarchical relations between the different representations, and so place rules in different nodes along the hierarchy.

After that we introduced GSR, which give a relatively informative representation that captures many rules. We showed that they can be seen as composed of two parts: one which is about the aggregation of information into some representation, and one which performs a computation on that representation, where the first part is related to compilation complexity and the second to communication complexity.

Finally, in Chapter 5 we used informational size to find an algebraic representation for partitions induced by voting rules. We showed that all the partitions induced by neutral and anonymous voting rules are in one-to-one correspondence with the subgroups of the symmetric group (of large enough index). We used that correspondence to prove a result regarding rules with minimal informational requirements.

We now discuss our results.

## Discussion

We start from our last chapter. Group theory is a very rich field in mathematics, and so we wonder if the correspondence can give us more results which can be used in voting theory. The answer to that depends, at least to some level, on the richness of informational size. The correspondence is based on the partitions of the ballot space which result from the informational size definitions. In order

to have more interesting results, we need to be able to differentiate between interesting rules.

Unfortunately, it seems that in many cases the definitions of informational size are not fine-grained enough to characterize many interesting rules. When we compared compilation and communication complexity, we saw that most rules have a maximal informational size (this means that the partition they induce corresponds to the trivial subgroup). It can also be shown, for example, that all Condorcet consistent rules have maximal informational size, which would place them in the same group with STV, Borda and other very different rules.

This sounds a bit grim, but it doesn't mean we cannot get more results from group theory at least in order to understand the limits of informational size. Considering that informational size is similar to compilation complexity, only in the level of the ballots, it is also possible to think of extensions of the correspondence that would give us more results. We give some suggestions for future work in this direction in the last section.

As for communication and compilation complexities, we saw that each of them has its limitations, placing many different rules in the same level. That is also why our hierarchy, which is basically representations used in compilation complexity, also relates many different rules to the same representation. So one of those alone cannot give us our informative representation. Could the two of them together be the key for that?

We saw that generalized scoring rules can describe many voting rules (though not all), and that they have elements related to both compilation complexity and communication complexity. In order to know if the formalism of representations we defined can be embedded in GSR, we need to understand under which conditions a representation satisfies additivity for $f$ and compatibility for $g$. These questions don't seem simple, as they are quite abstract. It might be that many of the concepts in Chapter 4 can benefit from being treated using another abstract field such as category theory (many notions we defined there might be characterized in terms of commutativity, for example). It remains to be seen if that can help answering any of the questions raised.

Finally, if we take compilation and communication complexity to be two different halves of the same problem of representation, then we have mostly dealt with compilation complexity, which can be taken as the "informational half". Communication complexity, which is more about computation, was not formalized to a similar level in this thesis. Coming up with something similar to the representations hierarchy, only for communication complexity, could help us in having better formalization in the "computational half" as well.

# Future Work

It seems as though our list of questions has only grown. There are many interesting directions to be followed from this work, involving different fields and intriguing questions. We present those that seem most important:

- The limitations of informational size – Just how limited is informational size? A possible line of investigation, to try and understand those limits, could be adding simple, reasonable axioms (such as monotonicity) and see how that restricts our set of partitions/subgroups. It might be possible to use the correspondence to show that with such axioms there are few subgroups left for our set of rules. We conjecture that with few natural axioms, we can narrow the set of subgroups to just the stabilizer groups of the first $k$ elements in $l_0$ or the last $k$ elements of $l_0$, for some $1 \leq k \leq m$.

- Lifting the correspondence – A natural question to ask is whether we can extend our algebraic correspondence for profiles and equivalences of profiles, using some group-theoretic notion. The natural structure to correspond with profiles is the product group $S_m^n$. Though some lemmas can be carried to this level (for example, if $P \sim P'$ then $\sigma P \sim \sigma P'$ for neutral rules), it does not seem like we can find an intuitive notion such as a group to correspond to equivalence of profiles. The main reason for this is that while the action of permutations on ballots was transitive (that is, for any two ballots $V, V'$ there is a permutation $\sigma$ such that $\sigma V = V'$), that is not the case for profiles, and so such a correspondence would have to include more complex notions.

  It is worth mentioning, though, that if such a correspondence is found, it might prove to be more interesting and richer than the correspondence we constructed here, as it would, for one thing, help in answering many compilation complexity questions.

- Sharpening the hierarchy of representations – We mentioned that our representations are not meant to cover all voting rules. However, there is still much work that can be done inside the hierarchy. We focused mostly on ABOVE and WSM, not discussing much about WMG or PSR, but perhaps those representations can also be given some characterization not in terms of profiles. In addition, one could also find new representations which should be part of that hierarchy.

  Another way to go is to try and estimate the size of one representation based on the size of another. For example, a technique that wasn't attempted here for estimating the size of ABOVE could be to look at the set AP (anonymous profiles), whose size is known, and see how many elements of ABOVE correspond uniquely to some element of AP, how many elements correspond to two anonymous profiles, and so on. This technique can be used between any two levels, where the size of one is known and the size of the other is not.

  Another thing that can be done is to connect more voting rules to necessary and sufficient representations. That can be done both by defining necessary and sufficient representations for specific voting rules, but also by checking if some axioms can limit us to a set of representations (as anonymity takes us from the set of all representations to the set of AP and those below it).

- Determining the relations between generalized scoring rules and compilation functions – This question, while having some importance in our ability to formalize rules better under the GSR formulation, is interesting mostly because it raises purely abstract questions about computations. For example, how do requirements of additivity (as that for $f$ in GSR) and compatibility (as for $g$ in GSR) affect our ability to compute different functions? This is a question we ask in the context of voting rules, but it can also be asked in a broader context. Can be find a representation that is not additive, cannot be made into an additive representation, and whose size is smaller than any additive representation?

  Of course, these questions need to be put in more formal terms, but they seem to be related to interesting topics in the theory of computation.

- Going after the computational half – Finally, much of this thesis was devoted for what we called "the informational half" of voting rules. The "computational half", which we associated with $g$ for GSR and with communication complexity, received somewhat less attention. It would be interesting to try and find a more rigorous formalization for that side of voting rules, so that perhaps the two formalizations can complement each other, to give a complete formal and informative way of describing voting rules.

# Appendix A

# List of Common Voting Rules

Here follows a comprehensive list of all the common voting rules mentioned in the thesis. Some of the rules might be defined differently is some places, but those differences are usually related to the use of tie breaking rules, and we will not be concerned with those. All rules are defined for $m$ candidates and $n$ voters (arbitrary $m, n$). For notation, see Definition 1.1.1

**Positional Scoring Rules** :

A positional scoring rule is any rule based on a scoring vector $\langle v_1, \ldots, v_m \rangle$, $v_i \in \mathbb{R}^+ \cup \{0\}$, where we associate the vector with "points" given to the candidates by the voter. When voter $i$ submits a ballot $V_i$, they give the candidate in the $j$th place $v_j$ points. Given a profile $P$, the score of a candidate $x$ is $\sum_{1 \leq i \leq m} N(x, i, P) \cdot v_i$ (where $N(x, j, P)$ is the number of voters who ranked candidate $x$ in the $j$th place in $P$).

Given a profile $P$ and a positional scoring rule $\mathcal{F}$, $\mathcal{F}(P)$ is the set of candidates with a maximal score. All the rules below are positional scoring rules, associated with different scoring vectors.

**Plurality** – Denoted $\mathcal{F}_P$, plurality is perhaps the most common voting rule. It is associated with the vector $\langle 1, 0, \ldots, 0 \rangle$ (the score of each candidate is the number of times they were voted first).

**Veto** – Also called anti-plurality at times, $\mathcal{F}_V$ is associated with $\langle 1, \ldots, 1, 0 \rangle$ (every candidate receives $n$ minus the number of times they were voted last), so the candidates who were voted last fewest times are the winners.

**Borda** – Associated with $\langle m-1, m-2, \ldots, 1, 0 \rangle$.

**$k$-Approval** – For any $1 \leq k < m$, $k$-approval is associated with the vector defined by $v_i = \begin{cases} 1 & 1 \leq i \leq l \\ 0 & \text{otherwise} \end{cases}$

**Condorcet Consistent Rules** :

$x \in X$ is a Condorcet winner if for every $y \neq x$ we have $N(x, y, P) > N(y, x, P)$ (most voters prefer $x$ to $y$). A Condorcet consistent rule is a rule that alway elects the Condorcet winner as the winning candidate, if the Condorcet winner exists (that is not always the case). All the rules below are Condorcet consistent:

**Copeland** – For each candidate $x$, let $W(x) = \{y \in X : N(x, y, P) > N(y, x, P)\}$ and $L(x) = \{y \in X : N(x, y, P) < N(y, x, P)\}$, the number of candidates $x$ wins over in pairwise elections and the number of candidates to which $x$ loses, respectively. The Copeland score of $x$ is $W(x) - L(x)$, and the winning candidates are those with the highest Copeland score.

**Kemeny** – Given a profile $P$ and a linear order of candidates $V$, the Kemeny Score of $V$ is:

$$\sum_{1 \leq i < m} N(V^i, V^{i+1}, P)$$

The candidate which is ranked first in the order which maximizes the Kemeny score wins.

**Dodgson** – The Dodgson score of a candidate $x$ is the minimal number of swaps required to make $x$ a Condorcet winner, when a swap is a change in a single ballot $V_i$, where we swap the order of two adjacent candidates in the ballots (that is, $V_i \rightsquigarrow V_i'$ where $V_i'^j = V_i^{j+1}, V_i'^{j+1} = V_i^j$ for some $1 \leq j < m$). The candidates with the lowest Dodgson score win.

**Cup** – Take a balanced binary tree (balanced here means that the difference in levels between two leaves is at most 1) with $m$ leaves. We associate each node with a candidate, starting from the leaves. First we assign each leaf with a single candidate (any assignment will do, though it inserts a non-neutral component to the rule). Now by recursion, for any node whose children are $x, y$, define the node to be associated with $x$ if $N(x, y, P) > N(y, x, P)$ and $y$ if $N(x, y, P) < N(y, x, P)$ (possibly using some tie breaking rule). The winner is the candidate associated with the root.

**Maximin** – The maximin score of $x$ in $P$ is $\min\{N(x, y, P) : y \neq x\}$. The winner is the candidate with the maximal score.

**Ranked Pairs** – We construct a linear order in steps. At every step we choose the two candidates $x, y$ which maximize $N(x, y, P)$, and which have not been chosen so far (possibly using a tie breaking rule). We set $x > y$ in our order, unless that contradicts the order we have constructed until now. In the end of the process we have a linear order. The candidate ranked first in that order is the winner.

**Weighted Majority Graphs (WMG) rules** :

Given a profile $P$ we can construct a graph $G(P)$ whose nodes are the candidates, and between two candidates $x, y$ there is a directed edge from $x$ to $y$ if $N(x, y, P) > N(y, x, P)$ ($x$ is preferred to $y$ by most voters) and the edge is labeled by $N(x, y, P)$. A rule $\mathcal{F}$ is called a WMG rule if for any two profiles $P, P'$ such that $G(P) = G(P')$ we have $\mathcal{F}(P) = \mathcal{F}(P')$.

This means that $\mathcal{F}$ can be determined entirely by knowing $N(x, y, P)$ for the different candidates. All the rules introduced as Condorcet consistent, with the exception of Dodgson, are WMG rules. We also note that WMG rules are not Condorcet consistent by definition, though they usually satisfy that requirement.

**Other Common Rules** :

**Plurality with Runoff** – Under plurality with runoff, we take the two candidates with the highest plurality score, call them $x$ and $y$, and the winner among those is $x$ if $N(x, y, P) > N(y, x, P)$ and $y$ if $N(x, y, P) < N(y, x, P)$. This rule might require the use of some tie breaking rule, which will make the rule non-neutral.

**STV** – Single Transferable Vote – STV – works the following way: If there is a candidate who was voted first by more than half the voters, that candidate wins. Otherwise, the candidate with the lowest plurality score is eliminated (we could either have all the candidates with minimal score removed, or one candidate based on the tie breaking rule), and we look at the profile as a profile of $m - 1$ candidates, where all the candidates that were ranked after the eliminated candidate in some ballot "shift" one place higher in that ballot. We continue with this process until some candidate is ranked first by more than half the voters.

**Bucklin** – The Bucklin score of a candidate $x$ is the minimal $k$ such that $\sum_{1 \leq i \leq k} N(x, i, P) > \frac{n}{2}$ (more than half the voters ranked $x$ in the $k$th place or higher). The candidates with the lowest Bucklin score win.

# Appendix B

# Proof of Lemma 4.1.6

First we note that while this claim seems immediate, it seems like a standard greedy algorithm will have problems with it. Consider the following example, where a greedy algorithm starts on the first row, trying to allocate the first cell that is positive and whose column was not previously chosen.

For the WSM-square presented below, of order 7 and sum 5, the algorithm will define the pairs $(i, \sigma(i))$ for the first 6 rows in the following way: $(1,1), (2,2), (3,3), (4,4), (5,5), (6,6)$, but then on the 7th row, there is no cell to choose from, and the algorithm is stuck.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 2 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 3 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 4 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 5 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 6 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |

The proof below is based on ideas of flow networks from computer science. It was pointed out by van Emde Boas that this lemma can also be proven in a simpler way as a result of the Birkhoff-von Neumann Theorem [18]. We bring this alternative proof in the end of this appendix. For historical reasons, though, we start by the flow networks proof, for which we must first define some notions.

**Definition B.1** (Flow Networks)**.** *A flow network is a tuple $N = (V, E, c, s, t)$, where $V$ is a set of vertices, $E$ a set of directed edges between the vertices (i.e., a subset of $V \times V$), $c$, called the capacity function, is a function $c : E \to \mathbb{R}^+$, and $s, t$ are two distinguished members of $V$, called 'source' and 'target' respectively.*

*A legal flow in the flow network $N$ is a function $f : E \to \mathbb{R}^+$ satisfying:*

- *Capacity constraint:* $\forall \langle u, v\rangle \in E \quad f(\langle u, v\rangle) \leq c(\langle u, v\rangle)$
- *Conservation of matter:* $\forall v \in V, v \neq s, t \quad \sum_{\langle u,v\rangle \in E} f(\langle u, v\rangle) = \sum_{\langle v,u\rangle \in E} f(\langle v, u\rangle)$

Put in words, the first constraint on a legal flow says we cannot have more flow through an edge than our capacity allows, and the seconds says that for any vertex which is not the source or the target, the total incoming flow has to equal the total outgoing flow.

**Definition B.2** (Cut)**.** *A cut in a flow network $N$ is any partition of $V$ into two disjoint sets $S, T$ such that $s \in S, t \in T$.*

*The capacity of the cut is defined as* $\sum_{\substack{\langle u,v\rangle \in E,\\ u\in S, v\in T}} c(\langle u, v\rangle)$.

**Definition B.3** (Flux)**.** *For a flow $f$ in a network $N$ we define the flux of the $f$:* $|f| = \sum_{\langle s,v\rangle \in E} f(\langle s, v\rangle) - \sum_{\langle u,s\rangle \in E} f(\langle u, s\rangle)$.

We will also be using the following well-known theorem from the theory of flow networks:

**Theorem B.4** (Min-Cut Max-Flow)**.** *For every flow network, the flux $|f|$ of an optimal flow $f$ is equal to the capacity of the minimal cut (i.e., the cut whose capacity is minimal among all possible cuts).*

More detailed definitions, as well as the proof of theorem, can be found in [5, Chapter 26]. We now proceed to the proof of Lemma 4.1.6.

*Proof.* We define a flow network based on the WSM-square $r$, show that the capacity of the minimal cut on the network is $m$, and so from the Min-Cut Max-Flow Theorem we find that there is an optimal flow $f$ of flux $m$ for the flow network. We then use the flow $f$ to define the required $\sigma$ for $r$. We start by describing the flow network.

The idea behind the network is that we will have $m$ nodes for the rows, $R_1, \ldots, R_m$, called row vertices, and $m$ nodes for the columns, $C_1, \ldots, C_m$, called column vertices. We will connect the row vertex $R_i$ with the column vertex $C_j$ based on the value of $r(i, j)$. We are interested in sending 1 unit of flow through each row, and getting it out through some column. The edges through which it flows will give us the positive elements we need. We give the formal construction:

We define $N = (V, E, c, s, t)$ to be the following:

$$V = \{s, t\} \bigcup \{R_i\}_{1\le i\le m} \bigcup \{C_i\}_{1\le i\le m}$$

$$\begin{aligned} E = \quad & \{(s, R_i) : 1 \le i \le m\} \\ \bigcup \quad & \{(R_i, C_j) : 1 \le i, j \le m, r(i, j) > 0\} \\ \bigcup \quad & \{(C_i, t) : 1 \le i \le m\} \end{aligned}$$

$$c(u, v) = \begin{cases} 1 & \text{if } u = s \text{ or } v = t \\ r(i, j) & \text{if } u = R_i, v = C_j \end{cases}$$

Figure B.1 illustrates the network created from $r$.

We show that the minimal cut of $N$ is $m$. Any cut $(S, T)$ satisfies $s \in S, t \in T$. If $T = \{t\}$ then clearly the capacity of the cut is $m$ (there are $m$ column vertices connected to $t$ with capacity 1 for each), and similarly the capacity is $m$ if $S = \{s\}$.

Now, if $T = \{t, C_{i_1}, \ldots, C_{i_k}\}$ for some $k \le m$ – that is to say, it contains the target and some subset of column vertices – then the capacity is necessarily

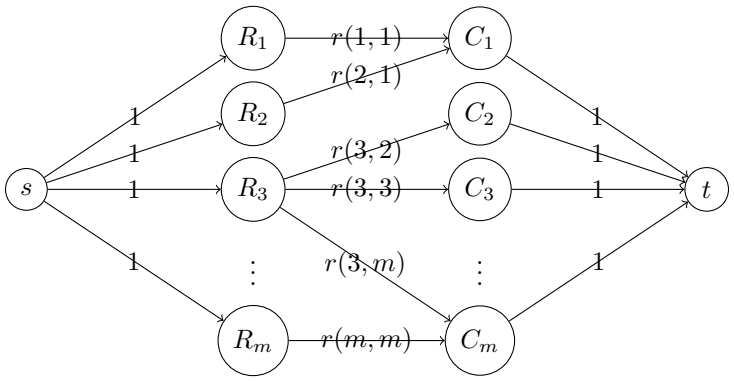


Figure B.1: An illustration of the flow network constructed from a WSM-square $r$. The labels on the edges denote the capacity.

$\geq m$, since there is an incoming capacity of $m-k$ from the column vertices not in $T$, and for each $C_i$ the incoming capacity is at least 1 (each column $i$ is accessibly at least one row $j$, where $r(i,j) > 0$, otherwise the sum on that row would be 0), so each of $C_{i_1}, \ldots, C_{i_k}$ adds at least 1 to the capacity, and in total we get cut capacity of at least $m$.

The last case we need to consider is when $T = \{t, C_{i_1}, \ldots, C_{i_k}, R_{j_1}, \ldots, R_{j_{k'}}\}$. Again, we have an incoming capacity of $m-k$ from all the column vertices not in $T$, and some additional incoming capacity for the column and row vertices in $T$. Look at the set $S_1$ of all the row vertices which are connected to column vertices in $T$ (i.e., all $R_j$s such $r(j, i_l) > 0$ for some $l \leq k$). We claim that $|S_1| \geq k$.

Assume that is not the case, and look at the sub-square of $r$ containing only the columns $C_{i_1}, \ldots, C_{i_k}$ and the rows in $S_1$. since $r(j, i_l) = 0$ for any $l \leq k$ and $j$ such that $R_j \notin S_1$, and since the sum on each of the columns is $n$, it follows that the total sum of all the entries of the sub-square is $nk$. Since we assumed there are less than $k$ rows, there has to be at least one row whose sum is greater or equal to $\lceil n\frac{k}{k-1} \rceil$ (there are at most $k-1$ rows, among which we have to divide a total sum of $nk$. The sum on the maximal row, which is minimal when the total sum is spread evenly on all the rows, equals to the ceiling of the average). That sum, however, is greater than $n$, in contradiction to $r$ being a WSM-square of sum $n$, and so we get that the number of vertices in $S_1$ is at least $k$.

Now, for any element in $S_1 \bigcap T$, that element contributes 1 into the capacity of the cut (each row vertex has an incoming capacity of 1), and for any element in $S_1 \setminus (S_1 \bigcap T)$, that element contributes at least 1 to the cut capacity, as it is connected to an element in $T$ with a positive outgoing capacity. So we get a capacity of at least $k$ from the set $S_1$, and combining this with the capacity $m-k$, which we've shown before is going into $t$, we get a cut capacity of at least $m$.

We have shown that the minimal cut capacity is $m$, and so there is an

optimal flow $f$ such that $|f| = m$. we use $f$ to construct the required $\sigma$. First we note that in order to have $|f| = m$, the flow $f(\langle s, R_i\rangle)$ has to be 1 for all $1 \leq i \leq m$, and the same for $f(\langle C_i, t\rangle)$. Let $A = \{(i,j) : f(\langle R_i, C_j\rangle) > 0\}$. From conservation of matter, and because of what we just mentioned, each $(i,j) \in A$ satisfies $f(\langle R_j, C_i\rangle) = 1$, and also for each $1 \leq i \leq m$ there is some (unique) $1 \leq j \leq m$ such that $(i,j) \in A$, and for each $1 \leq j \leq m$ there is some (unique) $1 \leq i \leq m$ such that $(i,j) \in A$. We can now define $\sigma(i) = j$ where $(i,j) \in A$.

From the previous paragraph it follows that $\sigma$ is well defined. To finish the proof we must show that $r(i, \sigma(i)) > 0$ for every $1 \leq i \leq m$. for some $1 \leq i \leq m$, we know that $f(\langle R_i, C_{\sigma(i)}\rangle) = 1$, hence $c(\langle R_i, C_{\sigma(i)}\rangle) \geq 1$, and from the definition of the capacity function it follows that $r(i, \sigma(i)) \geq 1$, as required. □

We remark that though the proof uses an existential argument ("there exists a flow"), there are actually algorithms for finding the optimal flow, and further yet, for cases where all the capacities are integer numbers – as it is in our case – they do it in polynomial time (e.g., see the Ford-Fulkerson algorithm), so in fact we have an effective way of finding $\sigma$ (and as a result, an effective way of constructing a profile $P$ which corresponds to $r$).

We now give the alternative proof using Birkhoff-von Neumann Theorem, mentioned earlier. For that we must first define:

**Definition B.5.** *A permutation matrix is a square matrix with a single 1 on every row and column and 0 everywhere else.*

In light of this definition, Lemma 4.1.6 claims there is a permutation matrix whose values are 1 exactly at the same cells where the corresponding WSM-square $r$ is positive.

**Definition B.6.** *A doubly stochastic matrix is a square matrix with non-negative real values where the sum on every row or column is 1.*

**Theorem B.7** (Birkhoff-von Neumann)**.** *For every doubly stochastic matrix $A$ of order $m$ there exist $k \in \mathbb{N}$, $k$ permutation matrices $\pi_1, \ldots, \pi_k$ of order $m$, and $k$ positive real values $\alpha_1, \ldots, \alpha_k$ such that $A = \sum_{1 \leq i \leq k} \alpha_i \pi_i$.*[1]

We now give the (much shorter) proof of Lemma 4.1.6, based on the theorem:

*Proof.* Let $r$ be a WSM-square of order $m$ and sum $n$, we show that there is a permutation matrix which takes positive values only in cells where $r$ is positive. Let $r/n$ denote the matrix we get by dividing every element of $r$ by $n$. $r/n$ is not necessarily a WSM-square, since the elements are not necessarily integers, but it is a doubly stochastic matrix, and therefore has a decomposition into permutation matrices.

Let $\pi$ be a permutation matrix in that decomposition, and let $\sigma$ be the permutation satisfying $\pi_{i,\sigma(i)} = 1$ for every $1 \leq i \leq m$. We immediately get that $r/n_{i,\sigma(i)} > 0$ for every $i$, and also $r(i, \sigma(i)) > 0$ for every $i$, as required. □

[1] More accurately, the theorem says that the set of doubly stochastic matrices is a convex hull of the permutation matrices of the same order.

# Appendix C

# Upper Bound on ABOVE and STV

In this appendix we use the definitions given for the ABOVE representation to calculate an upper bound for |ABOVE| (under certain assumptions), which is (when taken under the logarithm) also an upper bound for the compilation complexity of STV, as was shown by Chevaleyre et al. [3].

As we saw in Chapter 2, Chevaleyre et al. gave a lower bound for STV based on the number of combinations of $above(P, Z, x)$, when their bound was the number of possible combinations for a fixed candidate. In the terms of the set ABOVE, we may define an equivalence relation $\sim_x$ for $x \in X$, such that $f \sim_x g$ for $f, g \in$ ABOVE if $f(x, Z) = g(x, Z)$ for all $Z \subseteq X$. It is simple to verify that this is an equivalence relation. The result is a partition of ABOVE into equivalence classes, and if we denote the equivalence class of $f$ by $[f]$, then the set characterized by Chevaleyre et al. is $\{[f] : f \in \text{ABOVE}\}$.

For their upper bound, they assumed that these combinations are independent for all candidates (worst case scenario), and so the bound was $|\{[f] : f \in \text{ABOVE}\}|^m$. We take a somewhat different approach for an upper bound.

The main idea is that while Chevaleyre et al. fix a candidate, we keep the candidates free and fix instead all the sets of the same size, starting with sets of size 0 (only the empty set), then sets of size 1 (singletons) and so on. Due to the definition of ABOVE we get that when moving from sets of size $k$ to the sets of size $k + 1$, we only depend on the way we chose the distribution of sets of size $k$, while when you look at candidates, you cannot avoid having dependencies between all the different candidates.

We explain the approach in a slightly more formal way. We describe a tree structure such that its set of leaves is exactly the set ABOVE. For that we define the sets $S_i = \{Z \subseteq X : |Z| < i\}$ for $1 \leq i \leq m$, and for each $S_i$ we define $\text{ABOVE}_i = \{f : X \times S_i \to \mathbb{N} : f$ satisfies conditions I,II,III of ABOVE$\}$. We note that $\text{ABOVE}_m = \text{ABOVE}$.

Our tree will look like this: On the 0th level of the tree we have the root. On level one we have $\text{ABOVE}_1$, which are exactly the $\binom{n+m-1}{n}$ function which are different combinations of candidates ranked in the first place in some $n$ voter profile. Denote the nodes in the first level $f_{1,j}$ for $1 \leq j \leq \binom{n+m-1}{n}$. On the second level, for each node $f_{1,j}$ in the first level, we define its children to be $\{f \in \text{ABOVE}_2 : f|_{X \times S_1} = f_{1,j}\}$, that is, the functions which extend $f_{1,j}$ to the ranking on candidates in the second place. One can verify that for any

$f \in \text{ABOVE}_2$ there is some (unique) $f_{1,j}$ which is the parent of $f$, so the entire second level is $\text{ABOVE}_2$. The nodes in the second level are denoted $f_{2,j}$.

In general, on the $k+1$ level we define the children of $f_{k,j}$ (a node from level $k$) to be $\{f \in \text{ABOVE}_{k+1} : f|_{X \times S_k} = f_{k,j}\}$, and we get that the $k+1$ level $\text{ABOVE}_{k+1}$. As a result, we get that the leaves of the tree are $\text{ABOVE}_m = \text{ABOVE}$.

While it is not simple to count the sets $\text{ABOVE}_k$, except for the case $k = 1$ (otherwise we would get an immediate upper and lower bound for the compilation complexity of STV), we will see that it is simple to count the number of children of every node. In light of this, our strategy for calculating the upper bound is the following: For each node $f_{k,j}$, the branching factor of that node, $BF(f_{k,j})$ is the number of children of that node (i.e., $|\{f \in \text{ABOVE}_{k+1} : f|_{X \times S_k} = f_{k,j}\}|$). For every level $k$ we denote $BF_k = \max_j(BF(f_{k,j}))$, the maximal branching factor of that level. It follows that $\prod_{1 \leq k \leq m} BF_k$, the product of the maximal branching factors, gives us an upper bound on the number of leaves in the tree, which is the size of ABOVE.

The question remains, how do we find the maximal branching factor at each level. Let $f_{k,j}$ be a node that satisfies, for any $Z \subset X, |Z| = k$:

$$\sum_{x \in Z} f_{k,j}(x, Z \setminus \{x\}) = \frac{n}{\binom{m}{k}}$$

As there are $\binom{m}{k}$ sets of size $k$ in $X$, that last requirement means that if $P$ is a profile whose top $k$ ranks correspond to $f_{k,j}$ (we did not define this formally, but this is a natural extension of the natural correspondence of profiles to ABOVE functions), then the number of ballots in $P$ whose top $k$-tuple is composed of the set $Z$ is equal to the number of $k$-tuples composed of any other $k$-sized set $W$. In simpler words, $P$ distributes the votes evenly among all $k$ sets. Call such a node a uniform node.

We claim that the maximal branching factor of the $k$th level is given by $BF(f_{k,j})$, where $f_{k,j}$ is a uniform node. The intuition for this is that often in combinatorics and probability theory we get the highest number of combinations (in the case of combinatorics) or the highest entropy (for probability distributions) when we look at a uniform distribution.

We notice that there is a hidden assumption in this calculation, that $n$ is large enough for us to distribute it evenly among all different subsets of size $k$ (for any $k$). Formally this means that $n$ is at least the size of the least common multiple of $\{\binom{m}{k}\}_{1 \leq k \leq m}$.

We prove that claim, up to an assumption regarding the quality of Stirling's approximation.

**Claim C.0.4.** *Under the conditions described above, $BF_k = BF(f_{k,j})$, where $f_{k,j}$ is a uniform node.*

*Proof.* Let $f_{k,j}$ be a uniform node in the $k$th level. $f_{k,j}$ satisfies $\sum_{x \in Z} f_{k,j}(x, Z \setminus \{x\}) = \frac{n}{\binom{m}{k}}$ for any $Z \subset X, |Z| = k$. We can use this condition, and combine it

with condition III of the ABOVE functions to get a constraint on the children of $f_{k,j}$:

$$\sum_{1\le i\le m} f_{k+1,i}(x_i, Z) = \frac{n}{\binom{m}{k}} \quad \forall Z \subset X, |Z| = k$$

where $f_{k+1,i}$ is a child of $f_{k,j}$.

Let $f_{k,j'}$ be any other node in the $k$th level. We can identify $f_{k,j'}$ with a vector of length $\binom{k}{m}$, $\langle v_1, \ldots, v_{\binom{m}{k}}\rangle$, where $Z_1, \ldots, Z_{\binom{m}{k}}$ is an enumeration of all the sets of $k$ elements, and $v_i = \sum_{x\in Z_i} f_{k,j'}(x, Z_i \setminus \{x\})$. It is important to notice that the sum of the elements in the vector satisfies $\sum_{1\le i\le\binom{m}{k}} v_i = n$, as we have proved in the previous section.

We know that every child of $f_{k,j'}$ must satisfy condition III, which comes to:

$$\sum_{1\le i\le m} f_{k+1,i}(x_i, Z) = v_i$$

In both cases, we know how to calculate the number of children of each node. The important thing to notice is that $f(x, Z)$ is independent of $f(x, Y)$, as the only relevant conditions are II and III. Therefore, to calculate the number of children, we simply calculate the number of ways to expand each $k$-set $Z$ with the other $m - k$ candidates, and take the product for all $k$-sets.

In the case of the uniform node, the number of ways to expand a set $Z$ is the number of ways to distribute $\frac{n}{\binom{m}{k}}$ points among $m - k$ candidates, which is $\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}$. The total number of ways to expand the function is therefore

$$\binom{\frac{n}{\binom{m}{k}} + (m-k) - 1}{\frac{n}{\binom{m}{k}}}^{\binom{m}{k}}$$

For $f_{k,j'}$ we get a simpler formula: For any set $Z_i$, the number of ways to expand $Z_i$ is the number of ways to distribute $v_i$ points among $m-k$ candidates, i.e., $\binom{v_i+(m-k)-1}{v_i}$, and the total number of ways of expanding $f_{k,j'}$:

$$\prod_{1\le i\le\binom{m}{k}} \binom{v_i + (m-k) - 1}{v_i}$$

We need to show that the expression for the uniform node is greater or equal to that of the other node.

$$\prod_{1\le i\le\binom{m}{k}} \binom{v_i + (m-k) - 1}{v_i} \le \binom{\frac{n}{\binom{m}{k}} + (m-k) - 1}{\frac{n}{\binom{m}{k}}}^{\binom{m}{k}} \Leftrightarrow$$

$$\log[\prod_{1\le i\le\binom{m}{k}}\binom{v_i+(m-k)-1}{v_i}]\le\log[\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}^{\binom{m}{k}}]\Leftrightarrow$$

$$\sum_{1\le i\le\binom{m}{k}}\log\binom{v_i+(m-k)-1}{v_i}\le\binom{m}{k}\log\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}\Leftrightarrow$$

$$\frac{\sum_{1\le i\le\binom{m}{k}}\log\binom{v_i+(m-k)-1}{v_i}}{\binom{m}{k}}\le\log\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}$$

At this point we recall Jensen's inequality, which states that for any real concave function $\varphi$, values $x_1,\ldots,x_n$ in its domain and positive weights $a_1,\ldots,a_n$ we have:

$$\frac{\sum a_i\varphi(x_i)}{\sum a_i}\le\varphi(\frac{\sum a_ix_i}{\sum a_i})$$

If we take the index $n$ in Jensen's inequality to be $\binom{m}{k}$, and weights $a_i$ to be 1, and the values $x_i$ to be $v_i$, this inequality becomes:

$$\frac{\sum\varphi(v_i)}{\binom{m}{k}}\le\varphi(\frac{\sum v_i}{\binom{m}{k}})$$

We recall that $\sum v_i=n$, and we define $\varphi(l)=\log\binom{l+(m-k)-1}{l}$, which gives us

$$\frac{\sum\log\binom{v_i+(m-k)-1}{v_i}}{\binom{m}{k}}\le\log\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}$$

which is exactly the inequality we had. There is one problem with the use of Jensen's inequality here: The inequality is defined for real concave functions, while our function is discrete (only defined for natural numbers), and so the notion of concaveness does not even apply to it. At this point we use Stirling's approximation, which gives:

$$\log(n!)\approx n\log n-n+O(\log n)$$

and by manipulating the formulas, one gets:

$$\log\binom{l+m-1}{l}\approx m\log(\frac{l}{m}+1)+l\log(\frac{m}{l}+1)$$

We show that the function $\varphi'(l)=m\log(\frac{l}{m}+1)+l\log(\frac{m}{l}+1)$ is concave, which means that the inequality applies to it.

We know that a function is concave on an interval iff its derivative is monotonically decreasing. We calculate the derivative:

$$\frac{d\varphi'(l)}{dl} = (m-k)\frac{1}{\frac{l}{m-k}+1}\frac{1}{m-k} + \log(\frac{m-k}{l}+1) + l\frac{1}{\frac{m-k}{l}+1}(\frac{-(m-k)}{l^2}) =$$

$$\frac{m-k}{l+(m-k)} + \log(\frac{m-k}{l}+1) + \frac{l}{l+(m-k)}\frac{-(m-k)}{l} =$$

$$\log(\frac{m-k}{l}+1) + \frac{m-k}{l+(m-k)} - \frac{m-k}{l+(m-k)} = \log(\frac{m-k}{l}+1)$$

which is monotonically decreasing as $l$ grows, so the function is concave. A "wordy" explanation of why the function is concave: We have $\log f(x)$, and in order for that to be not concave, $f(x)$ has to grow faster than $e^x$, but $f$ is a binomial coefficient, and it is in fact less than $(v+(m-k)-1)^{(m-k)-1}$, which is of a fixed power.

This gives us, under the assumption that Stirling's approximation is good enough to not damage the inequality when we move from the binomial function to the approximation, that uniform nodes give us the maximal branching factor. □

Now we can calculate the upper bound, as the product of the maximal branching factors. We have already seen in the proof above that the branching factor for a uniform node is $\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}^{\binom{m}{k}}$, so the product gives us:

$$\prod_{1\le k\le m}\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}^{\binom{m}{k}}$$

We take the logarithm in order to calculate the compilation complexity, and we get

$$\log\prod_{1\le k\le m}\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}^{\binom{m}{k}} =$$

$$\sum_{1\le k\le m}\binom{m}{k}\log\binom{\frac{n}{\binom{m}{k}}+(m-k)-1}{\frac{n}{\binom{m}{k}}}$$

We want to evaluate the behavior of this bound. Denote $l = \frac{n}{\binom{m}{k}}$. We can present an element in the sum as

$$\frac{n}{l}\log\binom{l+(m-k)-1}{l} \approx n\log(1+\frac{m-k}{l}) + \frac{n(m-k)}{l}\log(1+\frac{l}{m-k})$$

For $k=0$ we get $l=n$, and this expression becomes

$$n\log(1+\frac{m}{n}) + m\log(1+\frac{n}{m})$$

For the case $k = \frac{m}{2}$ this expression is harder to evaluate, and the result depends on the interaction element $l$. The size of $l$ is also a result of the restrictions we have placed on $n$. Namely, that it is at least the least common multiple of $\left\{\binom{m}{k}\right\}_{1\leq k\leq m}$. Farhi [8] showed that

$$\frac{2^{m+1}}{m+1} \leq lcm\left\{\binom{m}{k}\right\}_{1\leq k\leq m} \leq \frac{3^{m+1}}{m+1}$$

If $n \approx \frac{2^{m+1}}{m+1}$ we get that the interaction element $l$ is almost constant, and our expression becomes

$$\frac{2^m}{\sqrt{m}} \log m$$

Which gives a slight improvement on the bound given by Chevaleyre et al.

# Bibliography


[1] J. Bartholdi, C. Tovey, and M. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6(3):157–165, 1989.

[2] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. In *Proc. 33rd Conference on Current Trends in Theory and Practice of Computer Science*, SOFSEM'07. Springer-Verlag, 2007.

[3] Y. Chevaleyre, J. Lang, N. Maudet, , and G. Ravailly-Abadie. Compiling the votes of a subelectorate. In *Proc. 21st International Joint Conference on Artificial Intelligence*, IJCAI'09, 2009.

[4] V. Conitzer and T. Sandholm. Communication complexity of common voting rules. In *Proc. 6th ACM Conference on Electronic Commerce*, EC'05, 2005.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, Cambridge, 2001.

[6] J.D. Dixon and B. Mortimer. *Permutation Groups*. Springer, 1996.

[7] J. Draisma, E. Kushilevitz, and E. Weinreb. Partition arguments in multiparty communication complexity. *CoRR*, abs/0909.5684, 2009.

[8] B. Farhi. An identity involving the least common multiple of binomial coefficients and its application. *The American Mathematical Monthly*, 116(9):836–839, 2009.

[9] W. Gaertner. *A primer in social choice theory*. LSE perspectives in economic analysis. Oxford University Press, 2009.

[10] G. Janusz and J. Rotman. Outer automorphisms of $S_6$. *The American Mathematical Monthly*, 89(6):407–410, 1982.

[11] E. Kushilevitz and N. Nisan. *Communiction Complexity*. Cambridge University Press, 1997.

[12] J. Lang. personal communication, June 2011.

[13] K. O. May. A set of independent necessary and sufficient conditions for simple majority decision. *Econometrica*, 20(4):680–684, 1952.

[14] S. Sato. Informational requirements of social choice rules. *Mathematical Social Sciences*, 57(2):188–198, 2009.

[15] S. Sato. personal communication, January 2011.

[16] S. Sato. Corrigendum to "informational requirements of social choice rules" [Math. Social Sci. 57 (2009) 188–198]. *Mathematical Social Sciences*, 62(1):77, 2011.

[17] M. Suzuki. *Group Theory I.* Springer, 1982.

[18] P. van Emde Boas. personal communication, September 2011.

[19] L. Xia and V. Conitzer. Generalized scoring rules and the frequency of coalitional manipulability. In *Proc. 9th ACM Conference on Electronic Commerce*, EC'08, 2008.

[20] L. Xia and V. Conitzer. Finite local consistency characterizes generalized scoring rules. In *Proc. 24th International Joint Conference on Artificial Intelligence*, IJCAI'09, 2009.

[21] L. Xia and V. Conitzer. Compilation complexity of common voting rules. In *Proc. 24th AAAI Conference on Artificial Intelligence*, AAAI'10, 2010.

[22] A. C. Yao. Some complexity questions related to distributive computing(preliminary report). In *Proc. 11th ACM Symposium on Theory of Computing*, STOC'79, pages 209–213, New York, NY, USA, 1979. ACM.