2

Preliminaries

In this chapter, I first give definitions of voting, some common voting rules, and some desired properties. In the end of this chapter, I will give a brief introduction to some other major topics in Computational Social Choice not covered in this dissertation.

Let $\mathcal{C} = \{c_1, \ldots, c_m\}$ denote the set of *alternatives* (or *candidates*). Each voter uses a *linear order* on $\mathcal{C}$ to represent his/her preferences. A linear order is a transitive, antisymmetric, and total relation on $\mathcal{C}$. The set of all linear orders on $\mathcal{C}$ is denoted by $L(\mathcal{C})$. For any natural number n, an n-voter profile P on $\mathcal{C}$ is a vector consisting of n linear orders on $\mathcal{C}$, one from each voter. That is, $P = (V_1, \ldots, V_n)$, where for every $j \leqslant n$, $V_j \in L(\mathcal{C})$. The set of all n-profiles is denoted by $\mathcal{F}_n$. Throughout the dissertation, we let n denote the number of voters, and let m denote the number of alternatives.

For any linear order $V \in L(\mathcal{C})$ and any $i \leqslant m$, we let $\mathrm{Alt}(V, i)$ denote the alternative that is ranked in the ith position in V. A *voting rule* r is a function that maps any profile on $\mathcal{C}$ to a unique winning alternative (the winner), that is, $r : \mathcal{F}_1 \cup \mathcal{F}_2 \cup \ldots \to \mathcal{C}$. A *voting correspondence* r^c can select more than one winner, that is, $r^c : \mathcal{F}_1 \cup \mathcal{F}_2 \cup \ldots \to 2^{\mathcal{C}} \backslash \{\varnothing\}$. Mathematically, a voting rule is a special voting

correspondence that always selects a unique winner.

2.1 Common Voting Rules

In this section we give definitions of some common voting rules. In fact, most of them are defined to be the maximizer/minimizer of some type of "scores".[1] Therefore, these voting rules are actually defined to be voting correspondences plus some tie-breaking mechanisms. In this paper, if not mentioned specifically, ties are broken in the fixed order $c_1 \succ c_2 \succ \cdots \succ c_m$.[2] Below is a list of common voting rules that will be studied in this thesis.

• **(Positional) scoring rules**: Given a *scoring vector* $\vec{s}_m = (\vec{s}_m(1), \ldots, \vec{s}_m(m))$ of m integers, for any vote $V \in L(\mathcal{C})$ and any $c \in \mathcal{C}$, let $\vec{s}_m(V, c) = \vec{s}_m(j)$, where j is the rank of c in V. For any profile $P = (V_1, \ldots, V_n)$, let $\vec{s}_m(P, c) = \sum_{j=1}^{n} \vec{s}_m(V_j, c)$. The rule will select $c \in \mathcal{C}$ so that $\vec{s}_m(P, c)$ is maximized. We assume scores are integers and nonincreasing. Some examples of positional scoring rules are *Borda*, for which the scoring vector is $(m-1, m-2, \ldots, 0)$; *plurality*, for which the scoring vector is $(1, 0, \ldots, 0)$; and *veto*, for which the scoring vector is $(1, \ldots, 1, 0)$. When there are only two alternatives, Borda, plurality, and veto (as well as all other voting rules introduced below) are called *majority*.

The definition of positional scoring rules naturally extends to the case in which voters are weighted; the weights are represented by a vector $\vec{w} = (w_1, \ldots, w_n) \in \mathbb{R}_+^n$, where for any $i \leqslant n$, w_i is the weight of voter i. In particular, we let

$$\vec{s}_m(P, \vec{w}, c') = \sum_{i=1}^{n} w_i \cdot \vec{s}_m(V_i, c'),$$

and again, the rule will select $c \in \mathcal{C}$ so that $\vec{s}_m(P, c)$ is maximized.

[1] This idea will be generalized to define a class of voting rules called *generalized scoring rules*. See Section 5.1.

[2] Tie-breaking can have important impact on the properties of voting rules, e.g, the computational complexity of manipulation (Obraztsova et al., 2011; Obraztsova and Elkind, 2011).

• **Copeland$_\alpha$** ($0 \leqslant \alpha \leqslant 1$): For any two alternatives c_i and c_j, we can simulate a *pairwise election* between them, by seeing how many votes prefer c_i to c_j, and how many prefer c_j to c_i; the winner of the pairwise election is the one preferred more often. Then, an alternative receives one point for each win in a pairwise election, α points for each tie, and zero point for each loss. The winner is an alternative that maximizes the score.

• **Maximin**: Let $D_P(c_i, c_j)$ denote the number of votes that rank c_i ahead of c_j minus the number of votes that rank c_j ahead of c_i in the profile P. The winner is the alternative c that maximizes $\min\{D_P(c, c') : c' \in \mathcal{C}, c' \neq c\}$.

• **Ranked pairs**: This rule first creates an entire ranking of all the alternatives. In each step, we will consider a pair of alternatives c_i, c_j that we have not previously considered; specifically, we choose the remaining pair with the highest $D_P(c_i, c_j)$. We then fix the order $c_i \succ c_j$, unless this contradicts previous orders that we fixed (that is, it violates transitivity). We continue until we have considered all pairs of alternatives (hence we have a full ranking). The alternative at the top of the ranking wins.[3]

• **Voting trees**: A voting tree is a binary tree with m leaves, where each leaf is associated with an alternative. In each round, there is a pairwise election between an alternative c_i and its sibling c_j; if the majority of voters prefer c_i to c_j, then c_j is eliminated, and c_i is associated with the parent of these two nodes. The alternative that is associated with the root of the tree (i.e., wins all its rounds) is the winner.

• **Bucklin**: The Bucklin score of an alternative c, denoted by $B_P(c)$, is the smallest number t such that more than half of the votes rank c somewhere in the top

[3] We note that at any stage there could be two or more edges whose weights are the highest. In this dissertation, we first use *parallel-universe tie-breaking* (Conitzer et al., 2009b) to select multiple winners, that is, an alternative is a winner if there exists a way to break ties among the edges such that the alternative is ranked in the top position in the ranking created by ranked pairs. After obtaining all "parallel-universe" winners, we use a fixed-order tie-breaking mechanism to select a unique winner from them.

t positions. A Bucklin winner minimizes the lowest Bucklin score, and ties are broken by the number of times that the alternative is ranked within top $B_P(c)$ positions.

• **Plurality with runoff**: The rule has two steps. In the first step, all alternatives except the two that are ranked in the top position the most often are eliminated; in the second round, the plurality rule (a.k.a. *majority* rule in case of two alternatives) is used to select the winner.

• **Single transferable vote (STV), a.k.a. instant-runoff or alternative vote**: The election has m rounds. In each round, the alternative that gets the lowest plurality score (the number of times that the alternative is ranked in the top position) drops out, and is removed from all of the votes (so that votes for this alternative transfer to another alternative in the next round). The last-remaining alternative is the winner.[4]

• **Baldwin's rule:** This is a multi-round voting rule similar to STV. The election has m rounds. In each round, the alternative that gets the lowest Borda score drops out. The last-remaining alternative is the winner.

• **Nanson's rule:** This is another multi-round voting rule similar to STV. The election has multiple rounds. In each round, all alternatives with less than the average Borda score are eliminated. This process then repeated with the reduced set of alternatives until there is a single alternative left. Nanson's rule and Baldwin's rule are closely related, and indeed are sometimes confused (Niou, 1987).

2.2 Axiomatic Properties for Voting Rules

As we discussed in the introduction, since in the voting setting the voters' preferences are ordinal, it is hard to measure how "good" an alternative is to all voters. Therefore, it does not seem to be obvious how can we argue that a voting rule is "good" or

[4] In this dissertation we use fixed-order tie-breaking at all stages. Conitzer et al. (2009b) investigated the STV rule using parallel-universe tie-breaking.

not. To overcome this difficulty, economists have proposed some desired properties (or, *axioms*) that a good voting rule should satisfy, and have investigated how to characterize voting rules by which properties they satisfy. Below, we include a list of such properties. We say a voting rule r satisfies:

- **anonymity**, if the output of the rule is insensitive to the names of the voters;
- **neutrality**, if the output of the rule is insensitive to the names of the alternatives;
- **homogeneity**, if for any profile P and any $n \in \mathbb{N}$, $n > 0$, $r(P) = r(nP)$, where nP is the profile composed of n copies of P;
- **non-imposition**, if any alternative is the winner under *some* profile. That is, for any alternative c and any $n \in \mathbb{N}$, there exists an n-profile P such that that $r(P) = c$;
- **unanimity**, if $\mathrm{Alt}(V, 1) = c$ for all $V \in P$ implies $r(P) = c$;
- **(strong) monotonicity**, if for any profile $P = (V_1, \ldots, V_n)$ and another profile $P' = (V_1', \ldots, V_n')$ such that each V_i' is obtained from V_i by raising only $r(P)$, we have $r(P') = r(P)$;
- **consistency**, if, whenever we have two disjoint profiles P_1, P_2 with $r(P_1) = r(P_2)$, we must have $r(P_1 \cup P_2) = r(P_1) = r(P_2)$;
- **participation**, if for any profile P and any vote V, $r(P \cup \{V\}) \succeq_V r(P)$;
- **Pareto efficiency**, if for any profile P, there is no alternative c that is preferred to $r(P)$ by all the voters;

- **the Condorcet criterion**, if, whenever there exists a *Condorcet winner* in a voting profile P, we must have that $r(P)$ is the Condorcet winner. Here a Condorcet winner is the alternative that wins each pairwise elections;

- **the majority criterion**, if, whenever the majority of voters rank an alternative in the top position, that alternative must be the winner under r.

Table 2.1 summarizes whether some common voting rules mentioned above satisfy these axiomatic properties. The Wikipidea entry for "voting system" (`http://en.wikipedia.org/wiki/Voting_system`) is a good place for the definitions of more voting rules and axiomatic properties.

Table 2.1: Some common voting rules and their axiomatic properties.

	Pos. scoring	Copeland	Maximin	Ranked pairs	STV	Bucklin	Plurality w/ runoff
Anonymity Neutrality Homogeneity Pareto efficiency	Y	Y	Y	Y	Y	Y	Y
Monotonicity	Y	Y	Y	Y	N	Y	N
Consistency Participation	Y	N	N	N	N	N	N
Condorcet	N	Y	Y	Y	Y	N	N
Majority	N	Y	Y	Y	Y	Y	Y

Each of these axiomatic properties evaluates voting rules from a specific viewpoint. For example, anonymity measures how "fair" a voting rule is to the voters, while neutrality measures how "fair" a voting rule is to the alternatives. We next consider some other important concepts in voting.

Definition 2.2.1. *For any profile P, we let* $\mathrm{WMG}(P)$ *denote the* weighted majority graph *of P, defined as follows.* $\mathrm{WMG}(P)$ *is a directed graph whose vertices are the alternatives. For $i \neq j$, if $D_P(c_i, c_j) \geqslant 0$, then there is an edge (c_i, c_j) with weight $w_{ij} = D_P(c_i, c_j)$.*

Example 2.2.2. *Let P denote the profile defined in Example 1.2.1. The weighted majority graph of P is illustrated in Figure 2.1.*

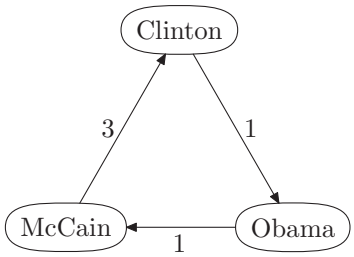

FIGURE 2.1: The weighted majority graph of the profile define in Example 1.2.1.

We say that a voting rule r is based on the *weighted majority graph (WMG)*, if the winner for r only depends on the weighted majority graph of the input profile. More precisely, for any pair of profiles P_1, P_2 such that $\mathrm{WMG}(P_1) = \mathrm{WMG}(P_2)$, we have $r(P_1) = r(P_2)$.

The following lemma will be frequently used in this dissertation. Informally, the lemma states that for any weighted directed graph G where the weights have the same parity, there exists a polynomially large profile whose WMG is G. This lemma allows us to focus on constructing a WMG that satisfies some desired properties, rather than constructing the profile directly. The lemma was first proved by McGarvey (1953), and there is also some subsequent work studying how to use as few votes as possible to obtain the desired WMG (Erdös and Moser, 1964). In this dissertation, we only need the polynomiality guaranteed by McGarvey's original result.

Lemma 2.2.3. *(McGarvey, 1953) Given a function $F : \mathcal{C} \times \mathcal{C} \to \mathbb{Z}$ such that*

1. *for all $c_1, c_2 \in \mathcal{C}$, $c_1 \neq c_2$, $F(c_1, c_2) = -F(c_2, c_1)$, and*

2. *either for all pairs of candidates $c_1, c_2 \in \mathcal{C}$ (with $c_1 \neq c_2$), $F(c_1, c_2)$ is even, or for all pairs of candidates $c_1, c_2 \in \mathcal{C}$ (with $c_1 \neq c_2$), $F(c_1, c_2)$ is odd,*

there exists a profile P such that for all $c_1, c_2 \in \mathcal{C}$, $c_1 \neq c_2$, $D_P(c_1, c_2) = F(c_1, c_2)$ and

$$|P| \leqslant \frac{1}{2} \sum_{c_1, c_2 :\ c_1 \neq c_2} |F(c_1, c_2) - F(c_2, c_1)| \quad .$$

2.3 A Brief Overview of Computational Social Choice

In this section, I will give a more detailed overview of some major topics in Computational Social Choice, which is an emerging interdisciplinary area at the intersection of Computer Science and Economics. Despite being young, Computational Social Choice has already found its place as a major topic in a number of Ph.D. dissertations since 2006, for example, Conitzer (2006a), Estivie (2007), Pini (2007), Altman (2007), Bouveret (2007), LeGrand (2008), Procaccia (2008), Faliszewski (2008), Aziz (2009), Uckelman (2009), Guo (2010), and Betzler (2010). An ever-increasing list of Ph.D. dissertations related to Computational Social Choice can be found at `http://www.illc.uva.nl/COMSOC/theses.html`. The Computational Social Choice workshop (COMSOC) has been held every other year since 2006. Computational Voting Theory is by far the most active research direction in Computational Social Choice. Below I will describe some major research topics in Computational Voting Theory, followed by some other research topics in Computational Social Choice.

2.3.1 Major Topics in Computational Voting Theory

Researchers in Computational Voting Theory have extensively studied the following topics.

- **How can we compute the winner or ranking more efficiently?** In traditional Social Choice Theory, voting rules are designed for aggregating voters' preferences over a generally small set of alternatives, where determining the winner is not a significant computational issue. In fact, computing the winner for many

common voting rules can be done in polynomial time. However, for some voting rules that have a long history, it has been shown that computing the winner is hard. For example, computing the winner for Kemeny's rule was shown to be NP-hard by Bartholdi et al. (1989b) and was later shown to be complete for parallel access to NP (Hemaspaandra et al., 2005); similar results have been obtained for Dodgson's rule—computing the winner for Dodgson's rule is NP-hard (Bartholdi et al., 1989b) and is also complete for parallel access to NP (Hemaspaandra et al., 1997). A third example is Slater's rule, for which computing the winner is NP-hard (Ailon et al., 2005; Alon, 2006; Conitzer, 2006b). For these voting rules, efficient approximation/heuristic algorithms have been proposed (Ailon et al., 2005; Conitzer, 2006b; Conitzer et al., 2006; Charon and Hudry, 2000; Hudry, 2006; Betzler et al., 2009a; Caragiannis et al., 2009, 2010). However, if the voters' preferences are restricted to be single-peaked, then a *Condorcet winner* always exists, which means that computing winners for both Kemeny's and Dodgson's rules are in P (Brandt et al., 2010a).

Kemeny's, Dodgson's, and Slater's rules are all defined by first computing the (weighted) majority graph, then applying a *tournament solution* (also called *choice set function* in Chapter 9) to the graph to select the winner. The computational complexity of computing some important tournament solutions has been investigated (Brandt et al., 2009, 2010b, 2011).

• **How can we communicate the voters' preferences more efficiently?** When the number of alternatives is extremely large, it is computationally inefficient for the agents to communicate their full preferences to the center. *Preference elicitation* studies how to query the agents iteratively to elicit enough information for computing the winner (Conitzer and Sandholm, 2002). The lowest number of bits of communication required to compute the winner, called *communication complexity*, was investigated for some common voting rules (Conitzer and Sandholm, 2005b). Eliciting single-peaked preferences were studied in Conitzer (2009) and Farfel and

Conitzer (2011)

Communication complexity provides a worst-case guarantee about the information that must be transmitted in order to compute the winner. However, it is quite likely that in practice, the elicitation process can usually end earlier. As I mentioned in Section 1.6, in these situations one important problem is how to compute the possible/necessary winners (Konczak and Lang, 2005). Besides the works discussed in Section 1.6 (i.e., Chevaleyre et al. (2010b), Chevaleyre et al. (2010a), Xia and Conitzer (2011a), and Xia et al. (2011)), there are a number of other works studying computing possible/necessary winners in different settings (Pini et al., 2007; Walsh, 2007; Betzler et al., 2009b; Betzler and Dorn, 2010; Baumeister and Rothe, 2010; Bachrach et al., 2010; Baumeister and Rothe, 2010; Baumeister et al., 2011).

• **How can we prevent voters from misreporting their preferences?** In this line of research, we investigate the possibility of using computational complexity to prevent manipulation. Therefore, we need computational problem (for a manipulator to compute a manipulation) to be as hard as possible. This topic will be discussed in Section 3.1.

• **How can we use computational complexity to protect elections from bribery and control?** Bribery and control are two ways for someone (not necessarily a voter) to influence the outcome of the election. In general, bribery is the behavior where the briber makes her favored alternative win by paying money to some voters to change their votes. Control in voting is more complicated than bribery in some sense—there are many different types of control, for example, introducing new voters/alternatives, removing existing voters/alternatives, or partition the voters/alternatives. Bartholdi et al. (1992) first studied several types of control in voting systems. Recently, more computational complexity results for bribery and control problems have been obtained. In the bribery problem setting of Faliszewski et al. (2009a), every voter has a cost, and we are asked whether there is a way to

bribe some voters to make a given candidate win, subject to a total budget constraint. Elkind et al. (2009b) considered an even finer setting called *swap-bribery*, where the voters are paid to "swap" adjacent alternatives in their votes. Faliszewski et al. (2009b) showed that the Copeland rules (for different α parameters) broadly resist known types of bribery and control. We (Conitzer et al., 2009a) studied the computational complexity of agenda control in sequential voting systems. A special type of control that introduces clones of alternatives was studied by Elkind et al. (2010a). The setting where the briber can use multiple types of bribery/control simultaneously was studied by Faliszewski et al. (2011a). On the negative side, Faliszewski et al. (2011b) showed that if the voters' preferences are single-peaked, then for many common voting rules the manipulation and control problems are in P.

• **How can we analyze voters' incentives and strategic behavior?** This topic will be discussed in Section 3.2.

• **How can we design novel voting rules when the set of alternatives has a combinatorial structure, and is exponentially large?** This topic will be discussed in Chapter 8.

2.3.2 Other Major Topics in Computational Social Choice

Besides Computational Voting Theory, researchers in Computational Social Choice have also studied the following topics.

• **Fair division**, a.k.a. *cake cutting* (Steinhaus, 1948), aims at providing a good allocation of resources that satisfies some desired properties. The most desired property is *envy-freeness*, which states that in the allocation, no agent would prefer the resources allocated to any other agent. Fair division is closely related to Multiagent Resource Allocation (Chevaleyre et al., 2006). For indivisible goods, Lipton et al. (2004) obtained approximability and inapproximability results for envy-free allocations. For one heterogeneous divisible good, Procaccia (2009) proved a lower bound

on the number of steps that must be used in any envy-free cake-cutting algorithm. Chen et al. (2010), and Mossel and Tamuz (2010) introduced truthfulness in cake-cutting, and proposed several cake-cutting algorithms that are truthful, envy-free, and also satisfy some other desired properties. Caragiannis et al. (2011) studied the cake-cutting setting where agents' valuation functions are *piecewise uniform with minimum length*. In such settings envy-freeness does not imply *proportionality*, and Caragiannis et al. proposed allocation algorithms that approximate the proportionality. Cohler et al. (2011) proposed an algorithm that computes the "optimal" envy-free allocation, that is, an envy-free allocation that has the highest social welfare among all envy-free allocations.

• **Judgement aggregation.** In judgement aggregation, a group of agents (judges) need to aggregate their opinions over several interrelated binary propositions. Recently, judgement aggregation has attracted much attention in Economics and Political Science (List and Puppe, 2009). It differs from combinatorial voting in the sense that in judgement aggregation, the aggregated values of the propositions must be consistent, while in combinatorial voting there is no such requirement. The best-known motivating example for the study of judgement aggregation is called the "doctrinal paradox" (Chapman, 1998), which is illustrated in the following example.

Example 2.3.1. *Suppose there are three judges who want to decide whether a defendant is liable. They use the majority rule to aggregate their opinions over three binary propositions: (1) $\mathcal{P}$, which is true if and only if the defendant did the action X, (2) $\mathcal{Q}$, which is true if and only if the defendant intended to do X, and finally (3) $\mathcal{R}$, which is true if and only if the judge thinks that the defendant is liable. Suppose all judges agree that the defendant is liable if and only if he did X and intended to do X (that is, $\mathcal{R} \leftrightarrow (\mathcal{P} \wedge \mathcal{Q})$). Consider the following three judges' judgments and the majority aggregation for each proposition.*

Table 2.2: The doctrinal paradox.

	$\mathcal{P}$	$\mathcal{Q}$	$\mathcal{R} \leftrightarrow (\mathcal{P} \wedge \mathcal{Q})$	$\mathcal{R}$
Judge 1	T	T	T	T
Judge 2	T	F	T	F
Judge 3	F	T	T	F
Majority	T	T	T	F

All judges' valuation over these propositions are consistent. However, the proposition-wise aggregations of the judges are not consistent ($\mathcal{P} = T$, $\mathcal{Q} = T$, and $\mathcal{R} \leftrightarrow \mathcal{P} \wedge \mathcal{Q} = T$ imply that $\mathcal{R} = T$). The message behind the example is that: under the majority rule, the judges come to the conclusion that the defendant did X and intended to do X, but they also agree that he is not liable, which contradicts the rule "anyone who did X and intended to do X is liable".

One of the first papers that considers computational aspects of judgement aggregation is Endriss et al. (2010a). They investigated the computational complexity of checking whether the set of propositions (called the *agenda*) satisfies some axioms, and showed that if these axioms are satisfied, then any judgement aggregation rule that satisfies some desired properties will never produce an inconsistent outcome. Later, Endriss et al. (2010b) investigated the computational complexity of computing the collective judgement as well as deciding whether an agent can influence the outcome by misreporting her valuation of the propositions. The latest work in this line of research is by Grandi and Endriss (2011), who proposed a general framework for aggregating binary variables under constraints, and obtained a general definition of paradoxes in this framework. This framework includes combinatorial voting and judgement aggregation as special cases.

2.4 Summary

In this Chapter we recalled the voting setting, notation that is used throughout this dissertation, and definitions of some common voting rules and axiomatic properties. We also gave a very brief introduction to some other major research directions in Computational Social Choice.