Winner Determination under Common Voting Rules using Truncated Ballots

Manel Ayadi

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à Paris-Dauphine et ISG-Tunis

Winner Determination under Common Voting Rules using Truncated Ballots

Soutenue par
Manel Ayadi
Le **18/10/2019**

École doctorale n°543
ED de Dauphine

Spécialité
Informatique

Composition du jury :

İpek Özkal Sanver Professeur, Université Bilgi d'Istanbul	*Rapporteur*
Nicolas MAUDET Professeur, Université Sorbonne	*Rapporteur*
Zied ELOUEDI Professeur, Institut Supérieur de Gestion-Tunis	*Examinateur*
Umberto GRANDI Maitre de conférences, Université de Toulouse	*Examinateur*
Tristan CAZENAVE Professeur, Université Paris Dauphine	*Président*
Vincent MERLIN Directeur de recherche, Université de Caen Basse Normandie	*Examinateur*
Nahla BEN AMOR Professeur, Institut Supérieur de Gestion-Tunis	*Directeur de thèse*
Jérôme LANG Directeur de recherche, Université Paris Dauphine	*Directeur de thèse*

Dauphine | PSL

Abstract

This dissertation focuses on voting with partial preferences in the form of *top-k* ballots. We study ways to minimize the amount of communication required to approximate or to output the correct winner from truncated ballots with respect to different voting rules and we propose and analyze different methods allowing a compromise between the accuracy of the result and the amount of communication required; some require only one round of communication, while others are interactive.

First, we assume that voters give, in a single shot, their *top-k* alternatives (for a fixed k); we define a version of voting rules (Borda, Harmonic, Copeland, Maximin, ranked pairs and STV) that works for such k-truncated votes. We consider two measures of the quality of the approximation: the probability of selecting the same winner as the original rule, and the score ratio. We do a worst-case study (for the latter measure only), and for both measures, an average-case study and a study from real data sets. Among the voting rules studied, we pay a particular attention to the *single transferable vote* (STV) voting rule. We start by testing empirically to which (quantitative) extent clone-proofness, which is a key property of (STV) and *ranked pairs* (RP), is preserved when replacing complete ballots by k-truncated ballots. Then, we study the closeness of the k-truncated version of STV to plurality with runoff. Finally, we investigate the computational complexity of the possible winner problem for *top-k* ballots under STV.

Second, we consider interactive communication protocols, that can ask voters for more information in an adaptive fashion. We study possible ways to elicit voters' preferences in an efficient manner so that the winner is determined. We consider exact methods and heuristics to guide the choice of the next voter to query. For the former we focus on STV voting rule; building on an existing protocol of Conitzer and Sandholm [25], we propose a new protocol and we show how we can reduce the amount of communication required in practice. For the latter, we explore two heuristic methods: (1) we adapt Monte Carlo Tree Search (MCTS) approach to vote elicitation in order to select the most prominent voter to ask in each round based on its evaluation function Upper Confidence Bounds for Trees (UCT), and (2) we propose an alternative search heuristic able to select the voter (under certain conditions) for whom we want

to reveal more information.

Key words: partial preferences, *top-k* ballots, approximations, preference elicitation.

Résumé

Cette thèse porte sur le vote avec des préférences partielles sous forme de top-k votes. Nous étudions des moyens pour réduire la quantité de communication requise pour déterminer le résultat de l'élection (de manière exacte ou approchée) à partir de bulletins tronqués selon différentes règles de vote et nous proposons et analysons différentes méthodes permettant un compromis entre précision du résultat et quantité de communication requise; certaines ne requièrent qu'une seule phase de communication, alors que d'autres sont dynamiques.

Premièrement, nous supposons que les électeurs donnent, d'un seul coup, leurs *top-k* préférences (pour un k fixe); nous définissons une version des règles de vote (Borda, Harmonique, Copeland, Maximin, ranked pairs et STV) qui fonctionne pour ces k-votes tronqués. Nous considérons deux mesures de la qualité de l'approximation : la probabilité de sélectionner le même gagnant que la règle originale et le ratio de score. Nous faisons une étude du pire des cas (pour cette dernière mesure seulement), et pour les deux mesures, une étude à partir de données synthétiques et réelles. Nous accordons une attention particulière à la règle *vote simple transférable* (STV). Nous commençons par tester empiriquement jusqu'à quel point la résistance aux clones, qui est une propriété clé de STV, est préservée lorsque on remplace des bulletins complets par des bulletins tronqués. Ensuite, nous étudions la proximité de la version k-tronquée de STV à la *pluralité à deux tours*. Enfin, nous étudions la complexité du problème des gagnants possibles pour les k-votes tronqués pour STV.

Deuxièmement, nous envisageons des protocoles de communication qui demandent aux électeurs de compléter leurs votes de manière dynamique. Nous étudions des moyens pour éliciter les préférences des électeurs de manière efficace afin de déterminer le gagnant. Nous considérons des méthodes exactes et des heuristiques pour guider le choix du prochain électeur à interroger. Pour le premier cas, nous nous concentrons sur STV ; en nous appuyant sur un protocole existant [25], nous proposons un nouveau protocole et nous montrons comment nous pouvons réduire la quantité de communication requise en pratique. Pour le deuxième cas, nous explorons deux méthodes heuristiques : (1) nous adaptons l'approche de la *recherche dans les arbres de Monte-Carlo* (MCTS)

au processus d'élicitation de vote, et (2) nous proposons une heuristique capable de déterminer l'électeur pour qui nous voulons révéler plus d'informations.

Mots clés : préférences partielles, top-k votes, approximations, élicitation de préférence.

Contents

List of Figures

Introduction (English Version)

Social choice theory concerns the design and analysis of methods for aggregating individual preferences towards a collective decision making. Central questions are: How can we choose a winning outcome from the often conflicting individuals' preferences? What are the properties of different voting systems? etc. Questions on social choice have captivated the interest of many researchers from the 18^{th} century with Nicolas de Condorcet and took off in 20^{th} century with Kenneth Arrow.

Voting theory, which is an important part of social choice theory, is one of the most popular ways of reaching common decisions using voting methods. Voting methods are applied to high-stakes settings such as political elections, middle-stakes including hiring committee and elections in local organizations, and low-stake such as scheduling a meeting date using an online survey. In this context, a set of individual opinions (preferences) is aggregated into a collective decision according to a predefined voting rule. Any electoral system (political or non-political) is based on a voting rule. There are a large number of variations in electoral systems, but the most common systems are plurality voting, the two-round (runoff) system or preferential voting such as single transferable vote (STV).

Classical Social choice focuses on normative and game-theoretic issues that includes an extensive study of the axiomatic method and proofs of several characterization and impossibility theorems. The most authoritative reference on classical social choice theory is the two-volume *Handbook of Social Choice and Welfare* [2, 3]. *Computational* social choice, by comparison, is an interdisciplinary field of study at the interface of social choice theory and theoretical computer science. It involves the application of techniques developed in computer science to the study of social choice mechanisms, such as the development of algorithms for aggregating individual preferences towards collective decisions.

One prominent research stream within computational social choice deals with communication aspects of voting. Chapter 10 of the *Handbook of Computational Social*

Choice edited by Boutilier and Rosenschein [13] provides background on incomplete information and communication in voting. This line of research focuses on quantifying the amount of information about preferences that is needed to accurately decide the election's outcome. Clearly a high communication burden is a drawback that can hinder the practical applicability of a voting rule. For example, in many legislatures, such as for the Irish presidential elections which employ STV electoral system; voters are allowed to only submit a partial ranking: they rank a subset of the candidates, and leave the rest unranked. This is the main motivation to study the problem of social choice under incomplete knowledge which comes down to partial preferences. This in turn raises a central question: *How elicit agents' preferences in an efficient way in order to minimize the amount of information required to determine a winning outcome ?* Topics covered in this area include: incomplete knowledge, preference elicitation (elicit the partial information in one shot and output a winner using only the information at hand), communication protocols (ask agents to expand their partial ballots in several shots, until the outcome is determined), online voting and vote streams.

In this thesis we tackle the problem of incomplete preferences in voting where we are interested in determining the winning outcome given partial preferences. As input we consider top-k votes by asking voters to list their k preferred candidates (instead of all candidates). Then the question is: *Given a voting rule, how can we determine the winner when we know only top-k votes?* We consider two ways of proceeding:

- For the first context, we assume that we know nothing more than these top-k votes. We define approximations of voting rules based on ordinal input (including Borda, Harmonic, Copeland, Maximin, ranked pairs and STV) using top-k votes instead of complete ones. To measure the quality of the proposed approximations, we consider two measures: (i) an empirical one by measuring the frequency with which the approximation outputs the *true* winner based on randomly generated profiles and on real-world data, and (ii) a theoretical one which is the score ratio between the score of the true winner and of the winner of the approximate rule. We pay particular attention to the STV voting rule by studying theoretically and empirically different aspects when using truncated ballots.

- For the second context, we are interested in incremental elicitation protocols by asking some voters to complete their votes until the winner is known. We focus on top-k queries where one voter will be asked at a time and i.e. if she is queried for the k^{th} time, she will be asked to provide her k^{th} preferred candidate. First, we propose exact methods for interactive vote elicitation where we elicit preferences from voters in rounds until the winner is known. Second, we explore heuristic

methods, based on evaluation functions that will guide the choice of the next voter to query.

Manuscript organization

This thesis is organized as follows.

Chapter 1 presents basic notions and concepts in voting theory. This includes the definition of the most important families of voting rules, such as: positional scoring rules, pairwise comparison rules and elimination based rules. Also, we investigate the use of several distributional preference models used to generate voters' preferences based on a prior distribution.

Chapter 2 focuses on determining the winner with partial preferences. We describe the basic notations of voting with partial preferences. Then, we present existing solutions to determine the winners with partial voters' preferences. Finally, we tackle the problem of preference elicitation where we present existing work on the two types of vote elicitation, namely: interactive elicitation and one-shot elicitation.

In Chapter 3, given top-k ballots, we propose "top-k approximations" of rules, which take only into account the top-k candidates of each ballot. Here, voters are assumed to report their top-k alternatives in a single shot. We consider two measures of the quality of the approximation: the probability of selecting the same winner as the original rule, and the score ratio.

Chapter 4 focuses on Single Transferable Vote (STV) voting rule. First, we empirically study to which (quantitative) extent the key property of STV, namely *clone-proofness*, is preserved when using truncated ballots. Second, we study the closeness of the k-truncated version of STV (namely STV_k) to plurality with runoff. Also, we investigate the computational complexity of the possible winner problem for top-k ballots. Finally, we propose ways to minimize the amount of communication required to use single-winner STV. Building on an existing protocol, we show how we can reduce the amount of communication required in practice. We then study empirically the average communication complexity of these protocols.

Chapter 5 proposes an incremental vote elicitation process using heuristic methods to guide the choice of the next voter to ask. First we adapt Monte Carlo Tree Search (MCTS) approach to vote elicitation in order to select the most prominent voter to ask in each round. Second, we propose an alternative search heuristic able to select

the voter (under certain conditions) for whom we want to reveal more information. Then, we empirically study the average number of questions asked to voters in order to determine the winner when using the two proposed heuristics.

Introduction (Version Française)

La théorie du choix social concerne la conception et l'analyse de méthodes d'agrégation des préférences individuelles en vue d'une prise de décision collective. Les questions centrales sont : Comment pouvons-nous choisir un gagnant parmi les préférences souvent contradictoires des individus ? Quelles sont les propriétés des différents systèmes de vote ? etc. Les questions sur le choix social ont captivé l'intérêt de nombreux chercheurs comme Nicolas de Condorcet, Jean-Charles de Borda and Kenneth Arrow.

La théorie du vote, qui est un élément important de la théorie du choix social, est l'une des moyens populaires pour parvenir à une décision commune en utilisant les méthodes de vote. Ces méthodes de vote s'appliquent dans un contexte politique (élection d'un président, d'une assemblée) ou non (conseil de classe, d'administration etc.). Dans ce contexte, un problème de vote consiste en un ensemble de candidats et un ensemble de votants, chque votant a des préférences sur les candidats. Le but d'une règle de vote est de déterminer un candidat préféré collectivement. Il existe un grand nombre de variantes dans les systèmes électoraux, mais les systèmes les plus courants sont les suivants : la pluralité, la pluralité à deux tours (appelée aussi « scrutin majoritaire à deux tours ») ou le vote préférentiel tel que le vote simple transférable (STV).

Le choix social *classique* met l'accent sur les questions normatives qui se basent sur une étude approfondie de la méthode axiomatique et des preuves de plusieurs caractérisations et des théorèmes d'impossibilité. La référence la plus fiable en matière de choix social classique est le *Handbook of Social Choice and Welfare* [2, 3]. Le choix social *computationnel*, par comparaison, est un domaine d'études interdisciplinaire à l'interface de la théorie du choix social et de l'informatique. Il s'agit de l'utilisation de notions et techniques venant de l'informatique (en particulier l'intelligence artificielle, la recherche opérationnelle, l'algorithmique) qui permet non seulement de résoudre des problèmes classiques de choix social, mais aussi de les voir sous un angle différent, et au-delà, de prendre conscience de nouveaux problèmes.

L'un des principaux axes de recherche dans le domaine du choix social computationnel

porte sur les aspects communicationnels du vote. Le chapitre 10 du *Handbook of Computational Social Choice* publié par Boutilier et Rosenschein [13] fournit l'historique et le contexte sur l'information et la communication incomplètes dans le vote. Cet axe de recherche vise à quantifier la quantité d'information nécessaire sur les préférences des votants pour décider avec précision du résultat de l'élection. Il est clair qu'une charge de communication élevée est un inconvénient qui peut entraver l'applicabilité pratique d'une règle de vote. Par exemple, dans de nombreuses assemblées législatives, comme lors des élections présidentielles irlandaises qui utilisent le système électoral STV, les électeurs ne peuvent soumettre qu'un ordre partiel : ils classent un sous-ensemble des candidats et laissent le reste sans classement. C'est la principale motivation pour étudier le problème du choix social dans le cadre d'une connaissance incomplète qui se résume à des préférences partielles. Cela soulève à son tour une question centrale : *Comment éliciter les préférences des agents de façon efficace afin de minimiser la quantité d'information requise pour déterminer un résultat gagnant* ? Parmi les sujets abordés dans ce domaine, mentionnons : les connaissances incomplètes, l'élicitation des préférences (éliciter les préférences partielles en un seul coup et désigner un gagnant en utilisant seulement l'information disponible), les protocoles de communication (demander aux votants de compléter leurs bulletins de vote partiels en plusieurs coups jusqu'à ce que le résultat soit déterminé), le vote en ligne et les flux de vote.

Dans cette thèse, nous abordons le problème des préférences incomplètes dans le vote en suggérant de demander aux électeurs de ne classer que leurs k candidats préférés (où k peut varier selon les électeurs et/ou au cours du processus). On dit que de tels votes sont k-tronqués. L'avantage est que non seulement cela permet d'économiser des efforts de communication, mais il est aussi souvent plus facile pour l'électeur de connaître la partie supérieure de son classement (composée de ses k candidats préférés) que le reste du classement. Ensuite, la question est la suivante : *étant donné une règle de vote, comment pouvons-nous déterminer le vainqueur alors que nous ne connaissons que les top-k votes des votants* ? Pour répondre à cette question, nous étudions la quantité d'information nécessaire pour déterminer le résultat de l'élection (de manière exacte ou approchée) à partir des bulletins tronqués selon différentes règles de vote et nous proposons et analysons différentes méthodes permettant un compromis entre précision du résultat et quantité de communication requise; certaines ne requièrent qu'une seule phase de communication, alors que d'autres sont dynamiques.

Afin de déterminer le résultat de l'élection à partir des bulletins tronqués, nous considérons deux façons de procéder :

1- Des protocoles non-dynamiques

Des protocoles à une seule phase de communication où toutes les préférences des votants doivent être recueillies en même temps. Dans ce modèle, le centre de vote recueille immédiatement les meilleurs k bulletins, pour une valeur fixe de k, et délivre un vainqueur à partir de cette information partielle sans exiger aux électeurs de fournir des informations supplémentaires. Dans ce cas, si l'on veut calculer avec certitude le vainqueur d'une règle de vote donnée, il faut demander aux électeurs de déclarer l'ensemble de leurs préférences, c'est-à-dire de déclarer leur ordre linéaire. Relaxons l'objectif de certitude et visons plutôt à calculer le gagnant avec une probabilité assez élevée. En échange, nous demandons moins d'information : Les électeurs donnent leurs top-k meilleurs candidats pour un $k > 1$ fixe, c'est-à-dire qu'ils rapportent une liste de classement des k candidats qu'ils préfèrent le plus.

Nous généralisons la définition d'une règle de vote de telle sorte qu'elle utilise des bulletins tronqués comme entrée : nous définissons des approximations des règles de vote qui prennent comme entrée les top-k candidats de chaque bulletin. La question est alors : *Ces approximations sont-elles de bons prédicteurs de la règle originale* ? Nous répondons à cette question en considérant deux mesures :

- la probabilité que la règle approximative sélectionne le 'vrai' gagnant. Pour ce faire, nous générons des profils aléatoires avec des ordres de préférence complets et tronquons ces ordres de manière à ce que les électeurs ne classent que leurs meilleurs choix. Nous calculons ensuite la probabilité que le gagnant des top-k bulletins de vote est le même que le gagnant des bulletins complets pour différentes règles votes.
- le ratio entre les scores (pour la règle originale) du vrai vainqueur et celui du vainqueur de la règle approximative.

Pour cette dernière mesure, nous présentons une analyse théorique du pire des cas. Pour les deux mesures, nous présentons une étude empirique, basée sur des profils générés aléatoirement (en utilisant le modèle Mallows ϕ [56] et une mixtures de Mallows [53]) et sur des données du monde réel.

Nous nous concentrons sur les règles de vote avec des préférences ordinales comme entrée où les électeurs soumettent un ordre complet sur les candidats. [1] Nous définissons plusieurs top-k règles qui correspondent aux règles de vote standard. Nous considérons trois types de règles : (1) les règles à base de score (PSR), à savoir : Borda, Harmonique ; (2) les règles à base de comparaison par paires, à savoir : Copeland, Maximin et ranked pairs, et (3) une règle basée sur l'élimination à savoir le v*ote simple transférable* (STV).

Pour la première mesure et afin d'évaluer la capacité des approximations à prédire le vrai gagnant, nous proposons maintenant une approche empirique. Puisque nous voulons mesurer expérimentalement la probabilité que le gagnant de la top-k approximation d'une règle coïncide avec le vrai gagnant, nous avons besoin de connaître ce vrai gagnant, et donc d'avoir accès au profil vrai. Pour ce faire, nous considérons le processus suivant :

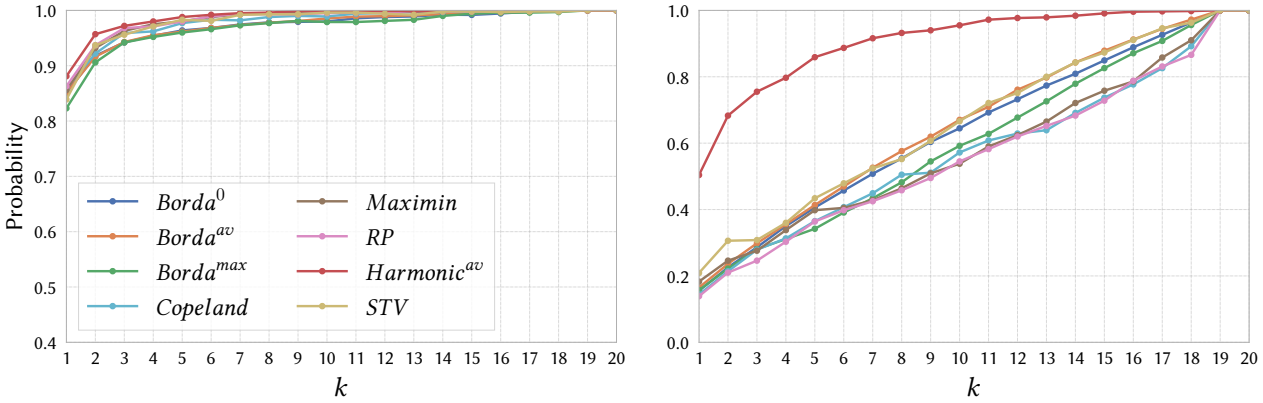
1) prendre un profil complet (avec n votants et m candidats) ;

2) pour $k = 1$ à $m - 2$: comparer le gagnant approximatif au gagnant correct

L'étape 1 varie selon que nous sommes dans le cadre de la génération aléatoire ou dans le cadre des données du monde réel. Pour le premier, nous générons un profil en fonction d'une distribution donnée. Pour le dernier, nous générons un profil en sélectionnant n^* ($n^* < n$) votes de manière uniforme et aléatoire parmi les votes disponible dans l'ensemble des données.

En appliquant cette approche, nos résultats expérimentaux sur des profils générés aléatoirement en utilisant le modèle Mallows suggèrent qu'une très petite valeur de k fonctionne très bien dans la pratique. Par exemple, la Figure 1 montre la probabilité que la règle approxmative donne le même vainqueur que la règle originale lorsque nous considèrons $m = 20$ et $n = 200$ et nous varions k et ϕ. Les résultats suggèrent que dans les larges élections et même si ϕ est très élevé ($\phi = 0,9$), les règles top-k arrivent à identifier le vrai vainqueur lorsque $k = 8$ (resp. $k = 14$) pour Harmonique (resp. les autres règles) sur $m = 20$. Pour $\phi = 1$, toutes les préférences des électeurs sont nécessaires pour déterminer le vainqueur sauf pour Harmonique qui ne nécessite que $k = 15$ préférences des électeurs sur 20 disponibles.

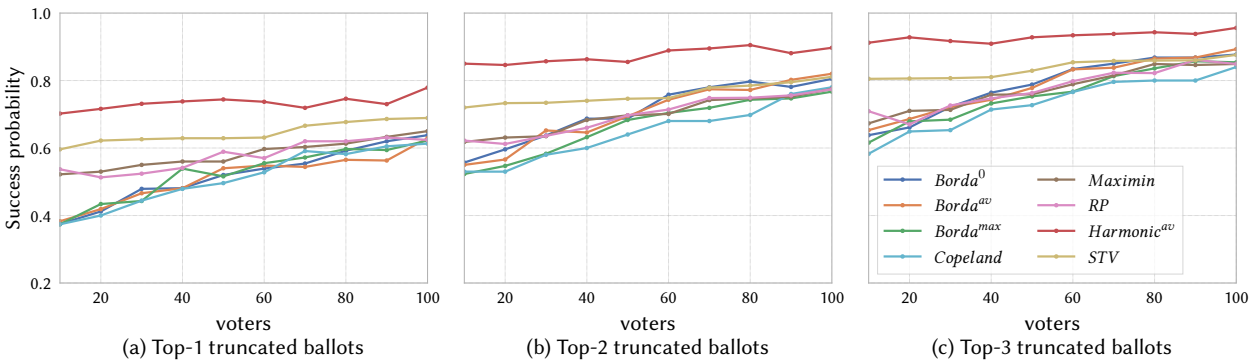
[1] Il y a d'autres règles qui ne sont pas basées sur des ordres mais sur des sous-ensembles de candidats tels que la règle d'approbation où chaque électeur peut approuver n'importe quel nombre de candidats et où le gagnant est le candidat le plus approuvé. Dans cette thèse, nous considérons des règles à entrée ordinale parce que nous nous concentrons sur les bulletins tronqués et dans d'autres cas (règles basées sur des sous-ensembles de candidats) la notion de bulletins tronqués n'a pas de sens.



(a) $\phi = 0.9$, $m = 20$ (b) $\phi = 1$, $m = 20$

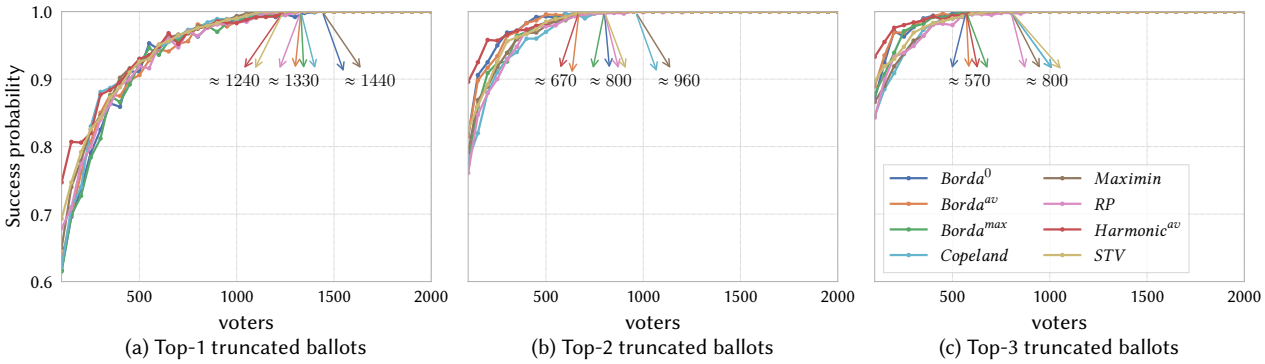
Figure 1: Taux de réussite, $n = 2000$, $m = 20$: Mallows ϕ lorsque $\phi \in \{0.9, 1\}$ et en variant k.

Nous considérons maintenant les données réelles de Preflib [58] : la base de données Dublin-Nord avec 12 candidats et 3662 électeurs. Les données de Dublin contiennent quelques bulletins avec un ordre partiels. Comme nous avons besoin de bulletins de vote complets pour mener nos expériences, nous avons exclu tous les bulletins partiels. Nous sommes intéressés à prédire le résultat pour des petites et larges élections. Pour ce faire, nous considérons des données avec des échantillons de n^* électeurs parmi n ($n^* < n$), en commençant par $n^* = 10$ et en incrémentant n^* par pas de 10. Dans chaque expérience, 1000 profils aléatoires sont construits avec n^* votants ; ensuite, nous considérons les k meilleurs bulletins obtenus à partir de ces profils, avec $k \in \{1, 2, 3\}$, et nous calculons la fréquence avec laquelle nous choisissons le vrai gagnant. La Figure 2 (resp. Figure 3) présente les résultats de Dublin pour les petites (resp. larges) élections $n^* \in \{10, ..., 100\}$ (resp. $n^* \in \{100, ..., 2000\}$).



(a) Top-1 truncated ballots (b) Top-2 truncated ballots (c) Top-3 truncated ballots

Figure 2: Taux de réussite, Dublin North: variant $k \in \{1, 2, 3\}$ et $n^* = \{10, \ldots, 100\}$.



(a) Top-1 truncated ballots (b) Top-2 truncated ballots (c) Top-3 truncated ballots

Figure 3: Taux de réussite, Dublin Nord: variant $k \in \{1, 2, 3\}$ et $n^* = \{100, \ldots, 2000\}$.

Nos résultats suggèrent que prédire le bon vainqueur pour les petites élections (Figure 2) échoue assez souvent lorsque la valeur de k est trop petite ($k \leqslant \frac{1}{4}m$). Nous constatons également que la probabilité augmente avec le nombre d'électeurs. Avec un petit nombre d'électeurs, Harmonique donne la meilleure performance suivie par STV, puis les autres règles, par exemple pour Harmonique (resp. STV), une précision de 92% (resp. 82%) est atteinte avec $k = 3$ sur 12 candidats et $n^* = 50$ contre environ 75% pour les autres règles. Pour les larges élections (Figure 3), la performance augmente avec n^* et k. Nos résultats montrent que demander $k = \frac{1}{4}m$ de candidats sur 12 pour chaque électeur est suffisant pour prédire le vainqueur correct lorsque $n^* > 570$ (resp. $n^* > 800$) pour Borda et Harmonique (pour les autres règles). En général, les différentes approximations nécessitent un nombre suffisant d'électeurs pour converger vers la correcte prédiction.

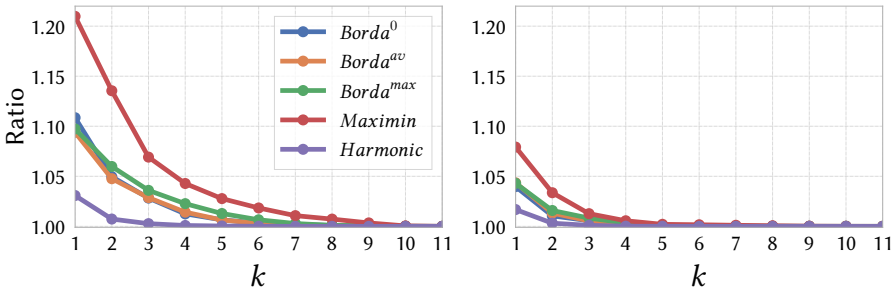
Pour la deuxième mesure (calculer le ratio entre le score du vrai vainqueur et celui du vainqueur de la règle approximative) nous avons considéré les règles dont la définition est basée sur la maximisation de score (Borda, Harmonique, Copeland et Maximin). Notez que comme la règle STV et *ranked pairs* ne sont pas basées sur la maximisation de score (pour une discussion, voir [23]), elles ne seront pas considérées dans notre étude. Les ratios de score dans le pire des cas sont particulièrement pertinents si le score d'un candidat est significatif au-delà de son utilisation pour déterminer le vainqueur. C'est certainement le cas de Borda, car la règle de Borda est souvent considéré comme une mesure du bien-être social (voir [28]). Ce ratio de score du pire des cas est appelé le prix de la *top-k* troncation.

Definition 0.1. *Soit f une règle de vote définie par la maximisation d'un score S, et f_k une approximation top-k de f. Le prix de la top-k troncation pour f, f_k, m et k, est défini comme suit*

$$R(f, f_k, m, k) = \max_{P \in \mathcal{P}_m} \frac{S(f(P))}{S(f_k(P_k))} \tag{1}$$

Les limites obtenues dans le pire des cas sont plutôt négatives : très négatives pour Copeland (∞) et Maximin ($m-k$), moins négatives pour Borda ($\Theta(\frac{m}{k})$), et encore moins pour Harmonique ($\Theta(\frac{m}{k \log k})$). Bien que les limites théoriques sont, au mieux, modérément encourageantes, nos expériences montrent qu'en pratique le ratio d'approximation est bien meilleur que dans le pire des cas : nos résultats sur l'évaluation du ratio sur des profils générés en utilisant le modèle Mallows ainsi que des données réelles suggèrent qu'une très petite valeur de k fonctionne très bien en pratique ce qui contraste largement les limites théoriques.

Par example, si nous considérons les données 2002 de Dublin Nord ($m = 12, n = 3662$) avec des échantillons de n^* électeurs parmi n ($n^* < n$) où $n^* = \{15, 100\}$ (voir Figure 4). Dans chaque expérience, 1000 profils aléatoires sont construits avec n^* électeurs ; ensuite, nous considérons les k meilleurs bulletins obtenus à partir de ces profils avec $k = \{1, ..., 11\}$. Là encore, les résultats sont très positifs, surtout avec un grand nombre d'électeurs.



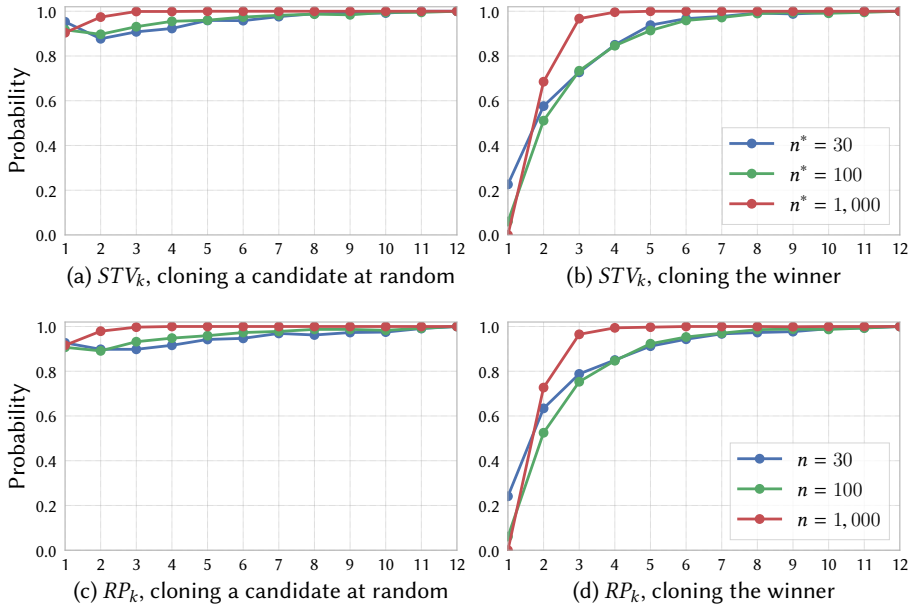
(a) $n = 15$ (b) $n = 100$

Figure 4: Ratio approximatif avec l'ensemble de données de Dublin Nord pour différentes règles de vote tronquées.

Dans le même contexte (i.e. top-k communiqué en un seul coup), nous avons accordé une attention particulière à la règle STV qui présente un intérêt particulier dans le vote car elle est difficile à manipuler et possède une propriété normative très importante : la résistance aux clones. En fait, plusieurs questions se posent : jusqu'à quel point la résistance au clonage est préservée lors du remplacement des bulletins complets par des bulletins k-tronqués ? Quel est le lien entre la règle STV_k (la version STV avec k bulletins tronqués) et la règle de la *pluralité à deux tours* ? Sachant qu'il est NP-complet de décider si un candidat est *vainqueur possible* pour STV étant donnée un profil partiel ; quelle est la complexité de ce problème lorsqu'on considère les "bulletins tronqués uniformes" (k fixe) ? Nos conclusions sur ces questions sont les suivantes, respectivement, comme suit :

- Pour la question de la résistance au clonage, les résultats obtenus à l'aide de

profils générés au hasard (sur le modèle de Mallows en variant ϕ) montrent que la résistance au clonage augmente rapidement avec k et diminue avec ϕ. De plus, elle augmente considérablement avec le nombre d'électeurs. Avec des données réelles, la résistance au clonage augmente rapidement avec k, voire plus rapidement qu'avec des profils générés au hasard. Le même résultat s'applique à la version k-tronquée de la règle *ranked pairs* (RP), à savoir RP_k (voir Figure 5).



(a) STV_k, cloning a candidate at random (b) STV_k, cloning the winner

(c) RP_k, cloning a candidate at random (d) RP_k, cloning the winner

Figure 5: Résistance de STV_k et RP_k au clonage en utilisant les données de Dublin (12 candidats et 3662 électeurs) avec des échantillons de n^* électeurs parmi n ($n^* < n$) où $n^* = \{30, 100, 1000\}$ puis nous clonons : (i) un candidat au hasard (Figure 5(a) et (c)) et (ii) le gagnant de STV_k et RP_k (Figure 5(b) et (d)). Dans chaque expérience, 1000 profils aléatoires sont construits avec des électeurs n^*.

- Concernant le lien entre STV_k et la *pluralité à deux tours*, nous avons étudié empiriquement la question (sur le modèle de Mallows en variant ϕ) et nous concluons que STV_k pour $k \approx \frac{2}{5}m$ (resp. $k = 1$) coïncide avec la *pluralité à deux tours* lorsque $\phi = 0,8$ (resp. $\phi = 0.7$). Pour $\phi = 1$, STV est très loin de la *pluralité à deux tours*. Le Tableau 1 résume les résultats de la probabilité moyenne (pour différentes valeurs de n) pour laquelle : la *pluralité à deux tours* et la pluralité ($=STV_1$) coïncident, et la *pluralité à deux tours* et STV coïncident.

	Plu. à/ 2 tours et Plu. coïncident	Plu. à/ 2 tours et STV coïncident
$\phi = 0.7$	0.97	1
$\phi = 0.8$	0.90	0.99
$\phi = 0.9$	0.75	0.89
$\phi = 1$	0.61	0.65

Table 1: Probabilité moyenne pour $n \in \{100, \ldots, 500\}$ pour laquelle : pluralité à deux tours et pluralité coïncident et pluralité à deux tours et STV coïncident lorsque $\phi \in \{0.7, 0.8, 0, 0.9, 1\}$.

- Les résultats sur la complexité sont très intéressants, en fait nous avons prouvé qu'avec $k = 1$ déterminer le gagnant possible pour STV peut être résolu en temps polynomial, mais le problème reste NP-complet quand $k \geqslant 2$ principalement en raison du nombre de complétions possible. Bien que le problème soit difficile dans le pire des cas, nous avons montré qu'en pratique, il existe un algorithme simple ($PW_{top\text{-}k}$ Algorithme 4.1) qui permet de classer les candidats en trois classes : gagnants possibles, perdants nécessaires et candidats sur lesquels nous ne pouvons pas décider.

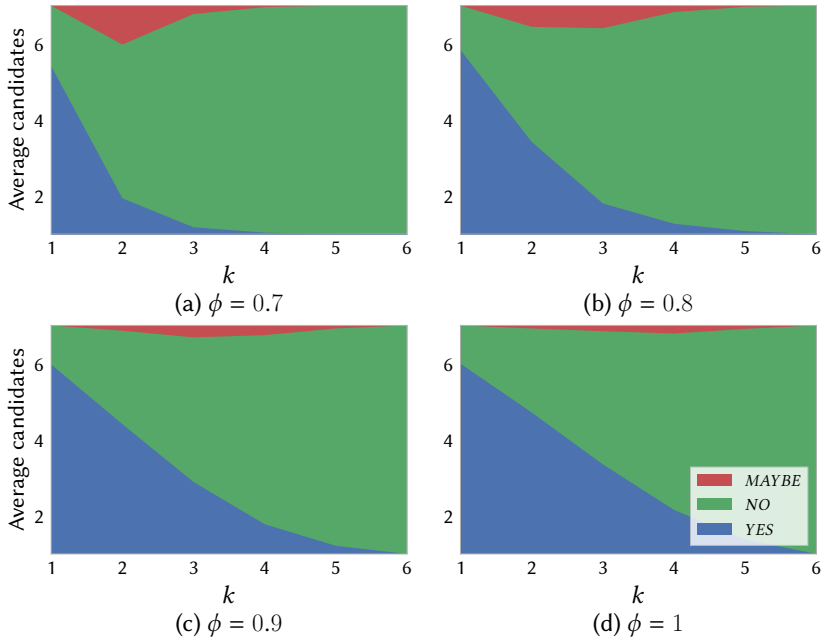


Figure 6: Identifier les gagnants possibles (YES), les perdants nécessaires (NO) et les candidats sur lesquels nous ne pouvons pas décider (MAYBE), quand $m = 7$, $n = 100$ et $\phi \in \{0.7, 0.8, 0.9, 1\}$: modèle Mallows ϕ.

Les résultats sur des profils générés aléatoirement à partir du modèle Mallows (voir Figure 6) montrent que l'algorithme proposé réduit significativement l'ensemble des candidats sur lesquels nous ne pouvons pas décider. Les résultats en variant ϕ montrent que lorsque $\phi = 0.7$ un seul candidat sur 7 n'est ni gagnant possible ni perdant nécessaire compte tenu des deux meilleurs votes. Pour $\phi = 0.8$, l'ensemble des candidats sur lesquels nous ne pouvons pas décider n'est présent qu'avec une probabilité de 2.7% et il diminue à 1.6% (resp. 1.08%) lorsque $\phi = 0.9$ (resp. $\phi = 1$).

2- Des protocoles dynamiques

Maintenant, nous voulions calculer le vrai gagnant, plutôt qu'un gagnant approximatif, en permettant une communication interactive (dynamique). Un protocole dynamique ou interactif demande aux électeurs d'élargir progressivement leurs votes partiels, jusqu'à ce que le résultat du vote soit déterminé. Nous avons étudié les moyens possibles de réduire au minimum la quantité de communication requise pour déterminer le résultat de l'élection à partir de bulletins tronqués. Plus précisément, la question ici est : *Quel est le nombre suffisant de bits (en moyenne) qui doit être communiqué par les électeurs au centre de vote pour que le gagnant soit déterminé* ? Pour répondre à cette question, nous avons d'abord proposé des méthodes exactes pour choisir l'électeur à interroger à chaque tour jusqu'à ce que le gagnant soit connu. Ensuite, nous avons exploré des heuristiques.

- Pour les méthodes exactes, nous nous sommes concentrés sur la règle STV. En nous basant sur un protocole proposé par Conitzer et Sandholm [25] que nous avons appelé P_1, nous avons proposé un nouveau protocole (P_2) capable de réduire la quantité de communication requise en pratique pour STV. Ensuite, nous avons étudié empiriquement le coût moyen de communication de P_1 et P_2, à partir de profils générés aléatoirement et de données réelles. Notre objectif est de déterminer la complexité de la communication, c'est-à-dire le nombre de questions aux quelles les électeurs doivent répondre en moyenne. Pour les profils générés aléatoirement, nous présentons les résultats de simulation avec $m = 7$ et $n = 100$, et laissons ϕ varier. Nous calculons le nombre de questions posées par P_1 et P_2 lorsque $\phi \in \{0.7, 0.8, 0.9, 1\}$. La Figure 7 montre le coût moyen de communication de P_1, P_2 en le comparant à P_{Worst}, le coût du pire cas de P_1. Les résultats suggèrent qu'en pratique, P_1 et P_2 posent moins de questions que dans le pire des cas (P_{Worst}). P_2 pose moins de questions que P_1 pour toutes les valeurs de ϕ, mais la différence est plus significative pour les ϕ inférieurs.

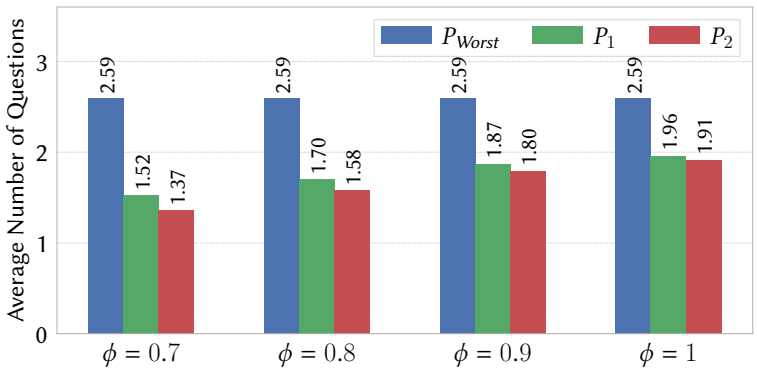


Figure 7: Coût moyen de communication, Mallows.

Afin d'évaluer nos protocoles avec des ensembles de données réelles, la Figure 8 montre le nombre moyen de questions posées aux électeurs par les protocoles sur les ensembles de données réelles. P_2 donne de meilleurs résultats que P_1 en posant 5 à 10% moins de questions.

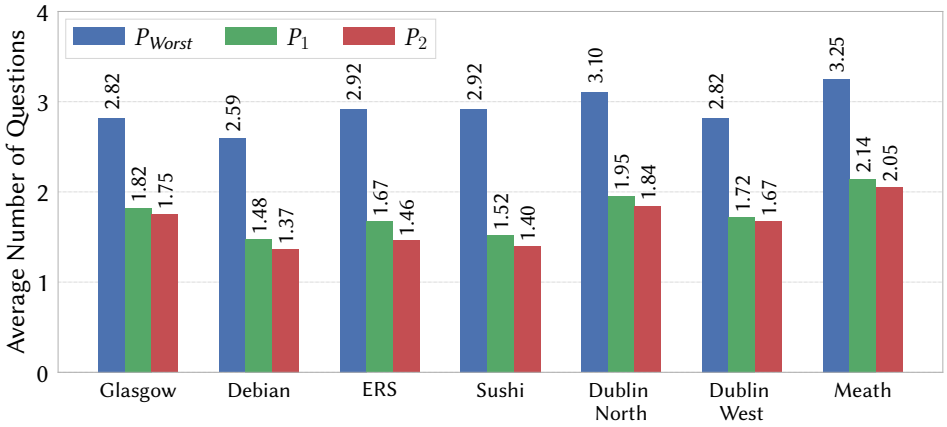


Figure 8: Coût moyen de communication, données réelles

- Pour les méthodes heuristiques, premièrement, nous proposons un processus incrémentiel d'élicitation de vote en explorant les méthodes heuristiques pour guider le choix du prochain électeur à intérroger. Compte tenu des préférences top-k, ces heuristiques seront utilisées pour choisir à chaque tour l'électeur à qui il est le plus pertinent de demander de compléter son k-vote en donnant son prochain candidat (ainsi classé k+1). À cette fin, nous proposons d'abord d'utiliser l'approche de recherche dans l'arbre de Monte-Carlo (MCTS) dans un contexte d'élicitation de vote pour sélectionner l'électeur le plus en vue à demander à chaque tour de compleler son vote en se basant sur la fonction d'évaluation *Upper Confidence Bounds for Trees* (UCT). Nous avons appelé ce protocole P_{MCTS}. Deuxièmement, nous

proposons une fonction d'évaluation heuristique simple qui se déroule en deux étapes principales : (1) nous identifions deux candidats clés pour lesquels nous voulons recueillir plus d'information, et ensuite (2) s'il y a au moins un électeur dont le bulletin actuel ne contient aucun de ces deux candidats, interroger cet électeur ; sinon, interroger un électeur dont le bulletin actuel contient un de ces deux candidats. Nous avons appelé ce protocole P_{Search}.

Nous avons étudié empiriquement le coût moyen de communication de P_{MCTS} et P_{Search} à partir de profils générés aléatoirement (le modèle Mallows ϕ pour différentes valeurs de ϕ) et des données réelles. Notre objectif est de déterminer le nombre de questions auxquelles les électeurs doivent répondre, en moyenne, afin de déterminer le gagnant selon les règles de vote Borda et Harmonique. Nous présentons les résultats de simulation avec $m = 7$ et $n = 10$ en variant la valeur $\phi \in \{0.7, 0.8, 0, 0.9, 1\}$. Figure 9 montre les résultats obtenus pour Borda avec P_{MCTS}, P_{Search} et P_{Rounds}, et nous montrons aussi P_{Worst} pour comparaison. Les résultats montrent que P_{MCTS} et P_{Search} sont efficaces pour réduire le nombre de questions posées aux électeurs sous les règles Borda et Harmonic. D'après les résultats, P_{MCTS} a un faible coût de communication selon les règles Borda et Harmonic.

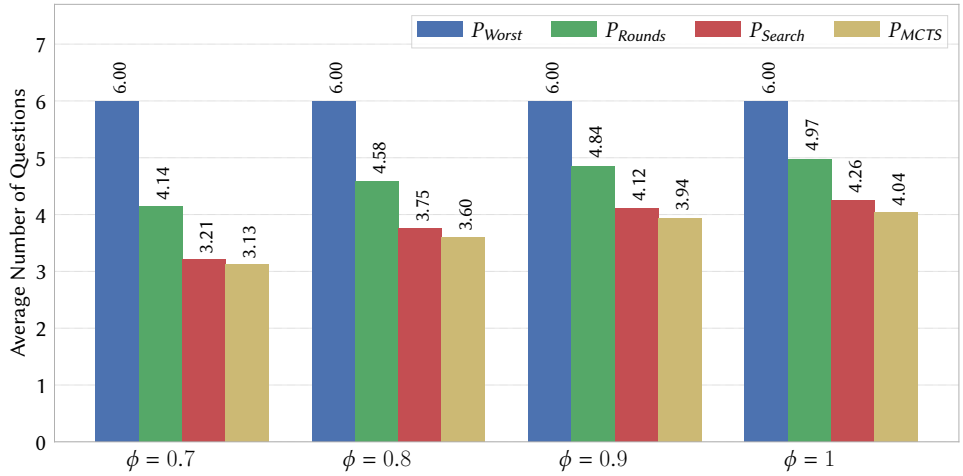


Figure 9: Coût moyen de communication de la règle Borda avec le modèle Mallows ϕ lorsque : $m = 7$, $n = 10$ and $\phi \in \{.7, .8, .9, 1\}$.

Organisation du manuscrit

Cette thèse est organisée comme suit:

Le chapitre 1 présente les notions et concepts de base de la théorie du vote. Ceci inclut la définition des familles de règles de vote les plus importantes, telles que : les règles à base de score, les règles de comparaison par paires et les règles basées sur l'élimination. De plus, nous examinons l'utilisation de plusieurs modèles de préférences utilisés pour générer les préférences des électeurs en fonction d'une distribution antérieure.

Le chapitre 2 se focalise sur la détermination du gagnant avec des préférences partielles. Nous décrivons les notations de base du vote avec des préférences partielles. Ensuite, nous présentons les solutions existantes pour déterminer le gagnant lorsque les préférences des électeurs sont incomplètes. Enfin, nous abordons le problème d'élicitation de préférence où nous présentons les travaux existants sur les deux types d'élicitation de vote, à savoir : l'élicitation interactive (dynamique) et l'élicitation non-interactive (non-dynamique).

Dans le chapitre 3, nous proposons des approximations des règles qui ne prennent en compte que les top-k meilleurs candidats (pour un $k > 1$ fixe) de chaque électeur. Dans ce cas, les électeurs rapportent leurs top-k préférences en un seul coup. Nous considérons deux mesures de la qualité de l'approximation : la probabilité de sélectionner le même gagnant que la règle originale et le ratio de score. Nous proposons une analyse théorique du pire des cas (pour cette dernière mesure seulement), et pour les deux mesures, une étude de cas moyen et une étude à partir d'ensembles de données réelles.

Le chapitre 4 se concentre sur la règle de *vote simple transférable* (STV). Tout d'abord, nous étudions empiriquement jusqu'à quel point la propriété clé de STV, à savoir la résistance aux clones, est préservée lorsque nous considérons des bulletins tronqués. Deuxièmement, nous étudions la proximité de la version k-tronquée de STV (à savoir STV_k) à la pluralité à deux tours. De plus, nous étudions la complexité du problème du gagnant possible pour les top-k vote. Enfin, nous proposons des moyens pour réduire au minimum la quantité de communication requise pour utiliser STV. En nous basant sur un protocole existant, nous montrons comment nous pouvons réduire la quantité de communication requise dans la pratique. Nous étudions ensuite empiriquement le coût moyen de communication de ces protocoles.

Le chapitre 5 propose un processus incrémentiel d'élicitation de vote en utilisant des méthodes heuristiques pour guider le choix du prochain électeur. Tout d'abord,

nous adaptons l'approche de recherche dans l'arbre de Monte-Carlo (MCTS) dans un contexte d'élicitation de vote pour sélectionner l'électeur le plus en vue à interroger dans chaque tour. Deuxièmement, nous proposons une heuristique de recherche alternative capable de sélectionner l'électeur (sous certaines conditions) pour lequel nous voulons révéler plus d'information. Ensuite, nous étudions empiriquement le nombre moyen de questions posées aux électeurs afin de déterminer le gagnant en utilisant les deux heuristiques proposées.

Part I

Background

Part I presents the background of this dissertation. It provides an overview of social choice theory. Chapter 1 introduces voting theory and the most important voting rules used to select the winner based on the voters' preferences. Chapter 2 deals with partial preferences. It provides an overview of a number of models and techniques developed to make decision with incomplete information about voters' preferences.

Chapter 1

Voting

Contents

This chapter provides an overview of the basic notations and concepts in voting theory. This includes the definition of the most important families of voting rules with ordinal preferences as input, such as: positional scoring rules, pairwise comparison rules and elimination based rules. Also, we investigate the use of several distributional preference models used to generate voters' preferences based on a prior distribution such as Impartial Culture (IC), Impartial Anonymous Culture (IAC), Mallows model, Mixture of Mallows and Single Peaked Impartial Culture (SP-IC).

1.1 Introduction

Social choice is an area of research concerned with the design of methods for collective decision making. The main question investigated in this field is: *How best to aggregate the agents' preferences over a given set of alternatives so as to determine the best alternative for the group?* For instance, in political elections, voters submit their individual preferences over a set of available candidates standing for election. In this context, given the collective preferences of the voters, various voting rules are used to determine which candidate should win the election.

We focus on voting rules with ordinal preferences as input where voters submit a collection of complete rankings/orders over candidates. For example, consider the situation in which a consumer is asked to compare different products (X, Y and Z), a natural given response is: '*I prefer product X to product Y to product Z*'. There are other rules that are not based on orders but on subsets of candidates such as approval rule where each voter may approve any number of candidates and the winner is the most-approved candidate. In this thesis we consider rules with ordinal input because we focus on truncated ballots and in other cases (rules based on subsets of candidates) the notion of truncated ballots is meaningless.

The analysis of methods for collective decision has captivated the interest of many researchers who were mainly interested in the design and theoretical evaluation of voting rules. Three results are most prominent and associated with the development of social choice theory:

1) *Condorcet's paradox*: presented by Marquis de Condorcet [60] who investigated majority voting and observed that the majoritarian preference relation can be problematic and irrational because it contains cycles. Suppose that a majority of voters prefers a to b, an other majority prefers b to c and yet a final one prefers c to a. Clearly, the pairwise majority relation is cyclic and violates transitivity.

2) *Arrow's impossibility theorem*: stated by Kenneth Arrow [1] who dealt with a class of possible aggregation methods which he called *social welfare functions*. *Arrow's impossibility theorem* states that a particular combination of axioms are only satisfied by dictatorial rules (the election outcome depends only on the opinion of one voter) when considering three or more alternatives.

3) *Gibbard-Satterthwaite's theorem* (aka. GST): presented (independently) by Allan Gibbard and Mark Satterthwaite [39, 64] who stated that given at least 3 candidates, any resolute voting rule (outputs a unique winner) that is *surjective* (for

every candidate, there is some profile for which that candidate wins) and *strategy-proof* (no voter will ever have an incentive to misrepresent her preferences) is a *dictatorship*.

This chapter is organized as follows. In Section 1.2, we describe basic notations and concepts of the voting model used in preference aggregation. In Section 1.3 we give an overview of the most important families of voting rules such as positional scoring rules (plurality, Borda and Harmonic), pairwise comparison rules (Copeland, Maximin and Ranked Pairs) and elimination based rules (plurality with runoff, single transferable vote and Coombs). Finally, in Section 1.4 we describe several distributional preference models used to generate voters' preferences based on a prior distribution.

1.2 Voting Model

Voting consists in aggregating voters' preferences over a set of candidates in order to determine a consensus decision or recommendation where a winner is chosen from a set of candidates. In the following, we start by introducing some basic notations on voting. An election is a triple $E = (N, A, P)$ with:

- $N = \{1, ..., n\}$ is the set of *voters* (or agents) with $|N| = n$;
- $A = \{a, b, ...\}$ is the set of *candidates* (or alternatives), with $|A| = m$; and
- $P = (\succ_1, ..., \succ_n)$ is the *complete preference profile* of voters in N, which corresponds to a collection of complete rankings on A. For each voter $i \in N$, $\succ_i \in P$ denotes the preference relation of voter i which corresponds to a complete linear order over A. We will refer to this order either as a *preference order*, a *ballot*, or, a *vote*; we use these terms interchangeably.

For any $a, b \in A$, $a \succ_i b$ means that voter i prefers a to b. For the sake of simplicity, we may write $a \succ b$ by omitting the voter's number.

Example 1.1. *Let us consider an election with three candidates and three voters i.e.* $A = \{a, b, c\}$ *and* $n = 3$. *Table 1.1 presents the complete preference profile of the voters. The number in the left column corresponds to the number of voters, while the voters' preferences are in the right column.*

1	$a \succ b \succ c$
1	$b \succ c \succ a$
1	$c \succ a \succ b$

Table 1.1: Example of a complete preference profile.

Definition 1.1. ***Majority Graph*** *Given a complete profile P, let $N_P(a,b) = \#\{i, a \succ_i b\}$ be the number of voters who prefer a to b in P. The majority graph $M(P)$ is the graph whose set of vertices is the set of the candidates A and in which for all $a, b \in A$, there is a directed edge from a to b (denoted by $a \to b$) in $M(P)$ if a beats b in a pairwise election, that is, if a strict majority of voters prefer a to b i.e. $N_P(a,b) > \frac{n}{2}$.*

Definition 1.2. ***The Weighted Majority Graph*** *Given a complete profile P, the weighted majority graph associated with P is the graph $MW(P)$ whose set of vertices is the set of the candidates A and in which for all $a, b \in A$, there is a directed edge from a to b weighted by the number of voters who prefer a to b in P. Since the votes in P are linear orders, knowing $MW(P)$ is equivalent to knowing the pairwise majority matrix defined by: for all $a, b \in A$, $Score_P(a,b) = N_P(a,b) - N_P(b,a) = \#\{i \in N | a \succ_i b\} - \#\{i \in N | b \succ_i a\}$.*

Given an election $E = (N, A, P)$ as input, a *voting rule* is used to determine the winner over the set of candidates where:

- An *irresolute voting rule* (aka. voting correspondence or social choice correspondence) is a function $F : E \mapsto S$ which, for each election, outputs a non-empty subset S of A, whose elements are called the (co-)winners of the election E under F.
- A *resolute* voting rule (aka. social choice rule) is a function $f : E \mapsto A$, which for each election outputs a single winner. Resolute rules are typically obtained from composing an irresolute rule with a tie-breaking mechanism either by using a predefined priority relation on candidates or a predefined priority relation on voters.

In the remaining, we will consider the resolute version of voting rules and break ties using prespecified linear order $\rhd$ over the candidates in A, called tie-breaking priority. The voting rule is then denoted by $f^{\rhd}$. To alleviate notation we will simply write f, leaving $\rhd$ implicit.

1.3 Voting Rules

The use of voting rules to aggregate preferences has become a topic of intense study, and one of great importance in ranking, recommender systems, resource allocation, and other applications of social choice to computational systems. Voting rule is a very common way of resolving disagreements, determining common opinions and choosing public policies. It represents a procedure for making a choice from the set of candidates. Various voting rules exist in the literature, we focus on three important subclasses with ordinal preferences.

1) **Positional Scoring Rules (PSR)**: A positional scoring rule is defined by a vector $\vec{s} = (s_1, ..., s_m)$ such that $s_1 \geqslant ... \geqslant s_m$ and $s_1 > 0$. Each candidate receives s_j points from each voter who ranks her in the j^{th} position, and the score of a candidate is the total number of points she receives from all voters. The winner is the candidate with highest total score over all the votes. Let $S(x)$ for $x \in A$ denote the score of a candidate x. The most common positional scoring rules are:

   - *Plurality*: In plurality rule, each voter gives 1 point to the candidate she ranked first, and the winner is the candidate who receives the highest total number of points. The score vector for plurality rule is $\vec{s}_{Plurality} = (1, 0, \ldots, 0)$. Hence, the cumulative score of a candidate corresponds to the number of voters by which it is ranked first.
   - *Borda*: In Borda rule, each candidate receives $m-1$ points from each voter i who ranks her in the first position, $m-2$ points to the candidate ranked second, or in general $m-k$ points to the candidate ranked in the k^{th} position. The score vector for Borda rule is $\vec{s}_{Borda} = (m-1, m-2, \ldots, 1, 0)$. The winner is the candidate who obtains the highest total number of points. Borda's rule takes a special place within the class of scoring rules as it chooses those candidates with the highest average rank in individual rankings.
   - *Harmonic*: In Harmonic rule, each candidate receives 1 point from each voter i who ranks her in the first position, $\frac{1}{2}$ points to the candidate ranked second, or in general $\frac{1}{k}$ points to the candidate ranked in the k^{th} position. The score vector for Harmonic rule is $\vec{s}_{Harmonic} = (1, 1/2, \ldots, 1/m)$. The winner is the candidate who obtains the highest total number of points.

2) **Pairwise Comparison Rules**: For these rules, the output is determined from the majority graph or from the weighted majority graph. For any two alternatives

a and b, we can simulate a *pairwise comparison* between them, by seeing how many voters prefer a to b, and how many prefer b to a. The rules that we will refer to are: Copeland (based on the majority graph); Maximin and Ranked Pairs (based on the weighted majority graph).

- *Copeland:* The Copeland rule chooses the candidate who beats the highest number of other candidates in pairwise elections. An alternative receives one point for each win in a pairwise election, $\frac{1}{2}$ point for each draw, and 0 point for each loss. The Copeland score of a candidate x is the number of candidates y with an edge from x to y in $M(P)$, plus half the number of candidates $y \neq x$ with no edge between x and y in $M(P)$. The Copeland winners are the candidates with highest Copeland score.
- *Maximin:* The Maximin rule selects candidates for which the minimum pairwise majority margin is maximized, that is, alternatives x for which

$$S_m(x) = min_{x \in A(y \neq x)} \left(N_P\left(x, y\right)\right)$$

- *Ranked Pairs (RP):* The ranked pairs rule proceeds by ranking all pairs of candidates (x, y) by decreasing order according to $Score_P(x, y)$ (using tie-breaking when necessary); starting from an empty graph over A, it then considers all pairs in the described order and includes a pair in the graph if and only if it does not create a cycle in it (if $Score_P(x, y) = 0$ we ignore the produced edge). At the end of the process, the graph is a complete ranking, whose top element is the winner. This way of breaking ties in RP is called *immediate tie-breaking*. Another alternative is to consider all the possible ways of breaking ties whenever two pairs are tied, and then use the tie-breaking at the very end to choose between the winners. This is called *Parallel Universe Tie breaking* (PUT) [15] which will not be considered in this thesis.

3) **Elimination-Based Rules**: For these rules, two or more rounds are needed in order to determine the outcome of the election (depending on the voting rule used). Within this family of rules, candidates having the lowest support are sequentially eliminated. Ballots are then transferred from the eliminated candidate to the next/earlier preferred candidate indicated on the ballots. In the following we present the most common elimination based rules such as, plurality with runoff, single transferable vote and Coombs. Note that like RP, these rules are affected by how ties are broken in each round where two possible ways exist: either to consider the immediate tie-breaking and break ties whenever they occur, or to use the PUT version by considering all the possible ways of breaking ties [23]. In this work we consider immediate tie-breaking for elimination based rules.

- *Plurality with runoff*: Plurality with runoff proceeds in two rounds: the two candidates ranked first by the largest number of voters go to the second round (runoff), and majority is used to determine the winner.
- *Single Transferable Vote (STV)*[1]: Given a prespecified linear order $\triangleright$ over the candidates, the $STV^{\triangleright}$ rule proceeds in rounds (up to $m-1$). In each round, the candidate with the smallest number of voters ranking them first is eliminated and the votes who supported it now support their preferred candidate among those that remain. If two or more candidates are tied for the least number of votes, the tie is broken immediately using a prespecified linear order $\triangleright$ over the candidates. The last remaining alternative is the STV winner.
- *Coombs*: As in STV, Coombs rule needs up to $m-1$ rounds. In each round, instead of deleting the candidate with the fewest first-place votes as done in STV, we eliminate the candidate with the most last-place votes, and the votes who supported it now support their earlier preferred candidate among those that remain (using the tie-breaking priority if necessary). The last remaining alternative is the Coombs winner.

We illustrate the different voting rules seen above using the following example.

Example 1.2. *Let us consider a setting with five candidates* $A = \{a, b, c, d, e\}$ *and* $n = 100$ *voters having the following preference profile* P*.*

33	$a \succ b \succ c \succ d \succ e$
16	$b \succ d \succ c \succ e \succ a$
3	$c \succ d \succ b \succ a \succ e$
8	$c \succ e \succ b \succ d \succ a$
18	$d \succ e \succ c \succ b \succ a$
22	$e \succ c \succ b \succ d \succ a$

When considering positional scoring rules, the winner for different rules is computed as follows:

- *The plurality vector is* $\vec{s}_{Plurality} = (1, 0, 0, 0, 0)$ *and the scores of candidates* a, b, c, d, e *are 33, 16, 11, 18 and 22, respectively. The plurality winner is candidate* a*.*

[1]For single-winner elections, STV is often called *instant runoff voting* (IRV). We keep however the terminology STV, which seems to be more popular among the community, even for single-winner elections.

- *The Borda vector is* $\vec{s}_{Borda} = (4, 3, 2, 1, 0)$. *The score of candidate a is computed as follows:* $(33 \times 4) + (16 \times 0) + (3 \times 1) + (8 \times 0) + (18 \times 0) + (22 \times 0)$. *By the same process, the scores of candidates a, b,c,d and e are 135, 247, 244, 192 and 182, respectively. Then, the Borda winner is candidate b.*

- *The Harmonic vector is* $\vec{s}_{Harmonic} = (1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5})$. *The score of candidate a is computed as follows:* $(33\times1)+(16\times\frac{1}{5})+(3\times\frac{1}{4})+(8\times\frac{1}{5})+(18\times\frac{1}{5})+(22\times\frac{1}{5})$. *By the same process, the scores of candidates a, b,c,d and e are , 46.55, 48, 44.33, 43.25 and 46.2; respectively. Then, the Harmonic winner is candidate b.*

Under elimination based rules, the winners are determined as follows:

- *For plurality with runoff, candidates a and e have the highest plurality score with 33 votes to a and 22 votes to e. In the second round, candidate e has the majority by beating candidate a (64,36). Thus, e is the winner under plurality with runoff.*

- *For STV, Figure 1.1 presents the eliminated candidate in each round.*

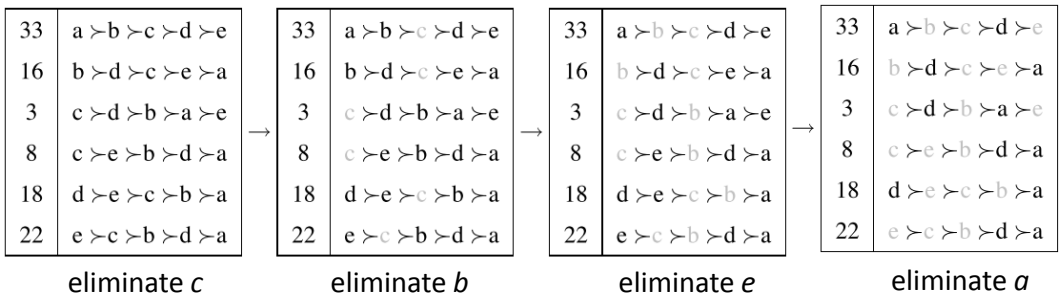


Figure 1.1: Eliminated candidate in each round under STV voting rule.

In the first round, candidate c receives the fewest number of votes (11 votes) and is eliminated. The votes for c are now transferred to d and e (3 votes to d and 8 votes to e). In the next round, candidate b is eliminated and the voters that supported it now support d. In the third round, candidate e is eliminated and its votes go to candidate d. Now, d has 67 votes against 33 for candidate a. Thus, d is the STV winner.

To illustrate the PUT version of STV, we consider the following profile of 10 voters over three candidates $A = \{a, b, c\}$*:*

4	$a \succ b \succ c$
2	$b \succ a \succ c$
1	$b \succ c \succ a$
3	$c \succ b \succ a$

Figure 1.2 shows how STV voting rule can lead to two different winners when considering the parallel universe version. In the first round, candidate b and c are tied where they both receive 3 points. If we consider a first scenario where we eliminate b (branch 1 in Figure 1.2), then a is the winner in the second round. However, if we consider the second scenario (branch 2 in Figure 1.2) by eliminating c in the first round, then b is the STV winner. Thus, a and b are the co-winners of STV following PUT version.

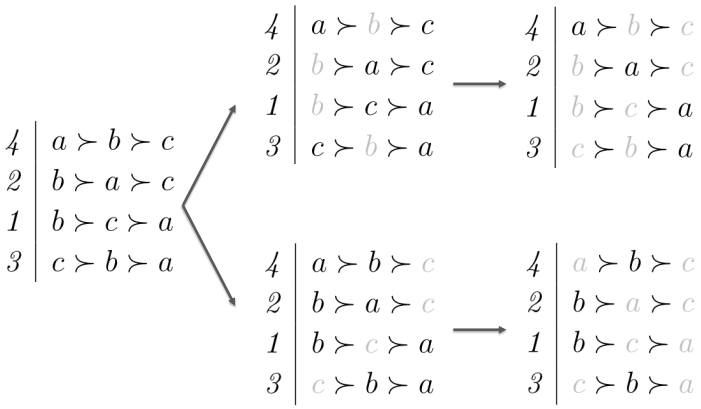


Figure 1.2: Example of STV voting rule when considering the parallel universe tie breaking version.

- *For Coombs voting rule candidate a receives the most last place votes (64 votes to a against 36 votes to e) which will be eliminated in the first round. Next, candidate e is eliminated with 52 votes against 30 to d and 18 to b. In the third round, candidate d is eliminated with 63 votes against 21 to b and 16 to c. Finally, candidate b is eliminated with 51 votes against 49 to c. Thus, only candidate c remains which is elected to be the winner under Coombs.*

For pairwise comparison voting rules, we consider a new example because in the first profile, the majority graph does not contain cycles under Copeland, thus the RP and Copeland winners are the same (i.e. candidate c). We consider the following profile with 15 voters and 4 candidates; we break ties using the following priority order $\rhd$*:* $a \succ b \succ c \succ d$*:*

$$\begin{array}{c|l} 3 & a \succ d \succ c \succ b \\ 3 & a \succ d \succ b \succ c \\ 5 & d \succ c \succ b \succ a \\ 4 & b \succ c \succ a \succ d \end{array}$$

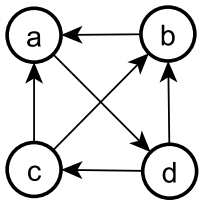


Figure 1.3: Majority graph representation.

- *For Copeland, Figure 1.3 shows the representation of the majority graph.*

  Candidate c beats candidates a (9,6) and b (8,7). Candidate d beats candidates b and c (11,4). Candidate a (resp. b) beats candidate d (10,5) (resp. candidate a (9,6)). Then, the scores of candidates a, b, c and d are 1, 1, 2, 2, respectively. Thus, d and c are tied and c is the winner thanks to the tie breaking.

- *For Maximin, Table 1.2 shows the pairwise majority matrix of P. For instance, the value 5 corresponds to* $N_P(a,d) - N_P(d,a) = 10 - 5 = 5$. *The last column* S_m *represents the minimum pairwise majority for each candidate.*

N_P	a	b	c	d	S_m
a	-	-3	-3	5	-3
b	3	-	-1	-7	-7
c	3	1	-	-7	-7
d	-5	7	7	-	-5

Table 1.2: Pairwise majority matrix representation for weighted majority graph.

  Under Maximin, the minimum pairwise majorities of the candidates are: $S_m(a) = -3$, $S_m(b) = -7$, $S_m(c) = -7$ *and* $S_m(d) = -5$. *Thus, the Maximin winner is the candidate with the highest minimum pairwise majority, namely: candidate a.*

- *For ranked pairs, Figure 1.4(a) shows the weighted majority graph induced from the majority matrix in Table 1.2, and Figure 1.4(b) presents the resulting acyclic graph when applying RP. Under RP, we add edges in descending order of weights and we retain an edge only if it does not create a cycle. We thus retain* $(d \to b)$, $(d \to c)$, $(a \to d)$, *omit* $(b \to a)$ *and* $(c \to a)$ *and retain* $(c \to b)$. *Hence, a is the RP winner which represents the only vertex without incoming edge in the resulting graph.*

(a) (b)

Figure 1.4: (a) Pairwise majority matrix representation, (b) the weighted majority graph induced from it, and (c) the resulting acyclic graph.

All the voting rules seen above are computable in polynomial time when considering the resolute version by breaking ties when necessary where:

- for positional scoring rules and plurality with runoff, the complexity is $O(nm)$.
- for Copeland, Maximin, ranked pairs and STV, the complexity is $O(nm^2)$.

Note that when considering the Parallel Universe Tie breaking (PUT) version, which consists in exploring all possibilities when different candidates are tied; some voting rules become NP-complete such as RP [15] and STV [23].

1.4 Distributional Preference Models

Analyzing voting rules by assuming that voters' preferences are drawn according to a prior distribution has become increasingly common in voting theory. In this section, we briefly review the most important distributional preference models used in social choice such as Impartial Culture (IC), Impartial Anonymous Culture (IAC), Mallows ϕ model, Mixture of Mallows and Single Peaked Impartial Culture (SP-IC) presented below:

- *Impartial Culture (IC)* [41]: Each voter's ranking is drawn uniformly at random from the set of all possible total orders over A. For m candidates, there are $m!$ possible strict rankings. IC assumes that the probability of observing any of the $m!$ preference orders is equally likely for each voter. If these are chosen by n voters, then there are $m!^n$ possible profiles.

- *Impartial Anonymous Culture (IAC)* [47]: Let us define a voting situation as a vector $\widetilde{n} = (n_1, n_2, ..., n_{m!})$, where n_i denotes the number of voters endowed with ranking number i (each of the $m!$ rankings being referred to by a number between 1 and $m!$). Under IAC, two profiles are equivalent if they correspond to the same voting situation. Then, instead of considering all profiles as equivalent as IC does, IAC considers all *voting situations* as equivalent. This assumes of course that the voting rule is anonymous, which implies that applying the rule to two equivalent profiles gives the same output.

Example 1.3. *Given two voters and two candidates a and b, IC assumes that all profiles are equally likely where four possible profiles (presented below) are generated with a probability of $\frac{1}{4}$ to occur.*

1	$a \succ b$	1	$b \succ a$	1	$a \succ b$	1	$b \succ a$
1	$a \succ b$	1	$b \succ a$	1	$b \succ a$	1	$a \succ b$

On the other hand, when considering IAC, we do not distinguish between profiles ($a \succ b$, $b \succ a$) and ($b \succ a, a \succ b$). Indeed, only three possible voting situations are presented: (2:$a \succ b$), (2:$b \succ a$) and (1:$a \succ b$,1:$b \succ a$) each one with a probability of $\frac{1}{3}$.

IC and IAC are the two most prominent distributions studied in the social choice literature and used for sampling voters' preferences. They provide a worst case assumption that do not generally reflect real world preferences [63]. More realistic probabilistic models of preferences, or parametrized families of distributions over rankings, have been proposed in statistics and econometrics. These models typically reflect some process by which people rank, judge or compare alternatives. A commonly used model adopted widely in voting theory is the *Mallows ϕ-model* [56].

- *Mallows ϕ model* [56]: (We will simply call the Mallows model hereafter) is parameterized by a modal or reference ranking σ and a dispersion parameter $\phi \in [0, 1]$, with $P(r; \sigma, \phi) = \frac{1}{Z}\phi^{d(r,\sigma)}$, where:
  - r is any ranking,
  - d is the Kendall tau distance, and
  - $Z = \sum_{r'} \phi^{d(r',\sigma)} = 1 \cdot (1 + \phi) \cdot (1 + \phi + \phi^2) \cdot ... \cdot (1 + ... + \phi^{m-1})$ is a normalization constant.

With small values of ϕ, the mass is concentrated around σ, while $\phi = 1$ gives the uniform distribution *(IC)*, where all profiles are equiprobable and all candidates are equally likely to be a winner.

Example 1.4. *Let us consider a reference ranking* $\sigma : a \succ b \succ c$ *over* 3 *candidates* $A = \{a, b, c\}$. *Table 1.3 presents the Kendall tau distance (the number of pairwise preference disagreements) of the possible rankings* r *with respect to* σ.

$a \succ b \succ c$	$a \succ c \succ b$	$b \succ a \succ c$	$b \succ c \succ a$	$c \succ a \succ b$	$c \succ b \succ a$
0	1	1	2	2	3

Table 1.3: Kendall tau distance of the possible rankings r with respect to the reference ranking $\sigma : a \succ b \succ c$.

Assume $\phi = 0.97$, *then* $Z = 1 \times (1 + 0.97) \times (1 + 0.97 + 0.97^2) = 5.734$. *In the following, we compute the probability of different rankings* r *with respect to* σ*:*

- $P(a \succ b \succ c) = \frac{1}{Z}\phi^{d(a\succ b\succ c, a\succ b\succ c)} = \frac{0.97^0}{5.734} = 0.174$
- $P(a \succ c \succ b) = \frac{1}{Z}\phi^{d(a\succ c\succ b, a\succ b\succ c)} = \frac{0.97^1}{5.734} = 0.169$
- $P(b \succ a \succ c) = \frac{1}{Z}\phi^{d(b\succ a\succ c, a\succ b\succ c)} = \frac{0.97^1}{5.734} = 0.169$
- $P(b \succ c \succ a) = \frac{1}{Z}\phi^{d(b\succ c\succ a, a\succ b\succ c)} = \frac{0.97^2}{5.734} = 0.164$
- $P(c \succ a \succ b) = \frac{1}{Z}\phi^{d(c\succ a\succ b, a\succ b\succ c)} = \frac{0.97^2}{5.734} = 0.164$
- $P(c \succ b \succ a) = \frac{1}{Z}\phi^{d(c\succ b\succ a, a\succ b\succ c)} = \frac{0.97^3}{5.734} = 0.159$

Doignon *et al.* [32] propose a practical method of sampling from the Mallows model using *Repeated Insertion Model (RIM)*. Given a reference ranking $\sigma = \sigma_1, \sigma_2, \ldots \sigma_m$ and a sequence of insertion probabilities p_{ij_i} for $j_i \leqslant i$, $i \leqslant m$ such that $\sum_{j=1}^{i} p_{ij} = 1$, $\forall i \leqslant m$; a ranking r is generated at random by first drawing an insertion vector $\mathbf{j} = (j_1, \ldots, j_m)$ satisfying $j_i \leqslant i$, $\forall i \leqslant m$ where each j_i is drawn independently with probability p_{ij}, and then applying a repeated insertion function $\Phi_\sigma(\mathbf{j})$ that maps an insertion vector $\mathbf{j}$ over a raking σ into a new ranking $\Phi_\sigma(\mathbf{j})$ by placing each σ_i, in turn into a ranking j_i for all $i \leqslant m$.

Assume a reference ranking $\sigma = a \succ b \succ c$ over three candidates a, b and c. At step 1 ($i = 1, j = 1$), a is added to r with a probability p_{11}. At step 2, b is inserted above a at the first position with a probability p_{21} ($i = 2, j = 1$) or below b at the second position with a probability p_{22} ($i = 2, j = 2$). At step 3, c is inserted at the first position with a probability p_{31} ($i = 3, j = 1$), at the middle with a probability p_{32} ($i = 3, j = 2$) or at the last position with a probability p_{33} ($i = 3, j = 3$). If we consider the insertion vector $\mathbf{j} = (1, 1, 2)$,

then $\Phi_\sigma(1,1,2) = (b \succ c \succ a)$ where according to σ, a is inserted in the first position, then we insert b in the first position by moving a down to get $b \succ a$, finally, c is inserted in the second position giving $b \succ c \succ a$.

Doignon *et al.* [32] show that the distribution induced by RIM with insertion function Φ_σ is identical to that of the Mallows ϕ model with reference ranking σ and dispersion parameter ϕ when:

$$p_{ij} = \frac{\phi^{i-j}}{1 + \phi + \cdots + \phi^{i-1}}, j \leqslant i \leqslant m \tag{1.1}$$

Example 1.5. *Let us consider the same setting as in Example 1.4 where* $\sigma : a \succ b \succ c$ *and* $\phi = 0.97$*. We start by computing the probability distributions using Eq. 1.1 as shown in the table below. For example* $p_{21} = \frac{0.97^{(2-1)}}{1+0.97} = 0.49$*.*

$\phi = 0.97$			
	$i = 1$	$i = 2$	$i = 3$
$j = 1$	$p_{11} = 1$	$p_{21} = 0.49$	$p_{31} = 0.325$
$j = 2$	-	$p_{22} = 0.51$	$p_{32} = 0.333$
$j = 3$	-	-	$p_{33} = 0.342$

Using the above probabilities, the model will generate different insertions vectors. Assume we consider the insertion vector $\boldsymbol{j} = (1,1,1)$*, then* $r = \Phi_\sigma(1,1,1) = (c \succ b \succ a)$ *where* a *is inserted first,* b *is inserted next in the first position by moving* a *down, finally,* c *is inserted in the first position by moving down* b *and* a*. Note that the probability induced when considering* $\boldsymbol{j}$ *and* σ *is equal to* $p_{11}.p_{21}.p_{31} = 1 \times 0.49 \times 0.325 = 0.159$ *which corresponds to the same probability produced from Mallows model in Example 1.4 when* $\sigma : a \succ b \succ c$ *and* $r = c \succ b \succ a$ *(i.e.* $P(c \succ b \succ a) = \frac{1}{Z}\phi^{d(c\succ b\succ a, a\succ b\succ c)} = \frac{0.97^3}{5.734} = 0.159$*).*

- *Mixture of Mallows model* [53]: To overcome the unimodal nature of Mallows model, mixtures of Mallow's have been considered. Let p be a positive integer, a mixture model consists of p Mallows models with a probability distribution over them. Formally, given:
  - reference rankings $\sigma_1, \ldots, \sigma_p$,
  - dispersion parameters $\phi_1, \ldots, \phi_p$, and
  - mixing coefficients (discrete probability distribution) $\lambda_1, \ldots, \lambda_p$ where each λ_i , $1 \leqslant i \leqslant p$ is between 0 and 1, and $\sum_{i=1}^{p} \lambda_p = 1$.

  We select rankings from the p models according to the probability distributions induced from RIM model [56, 53]. The mixing coefficients are used to choose

the appropriate model $i \in [1, p]$ for each generated ranking. One possible way (detailed in Algorithm 1.1) is to start with a uniform random real number in $[0, 1)$ and first subtract off the first probability λ_1, if the result is negative then we return the first model 1; otherwise, in a loop we select the next model and we subtract off the next probability until we get a negative result. Finally, we return the appropriate model.

Algorithm 1.1: Choose the Appropriate Model to Generate Rankings

Input : $\lambda = \{\lambda_1, \ldots, \lambda_p\}$, Model $= \{1, \ldots, p\}$

$i \leftarrow 1$
$x \leftarrow$ random real number $\in [0, 1)$
$draw \leftarrow x - \lambda_i$
while $draw > 0$ **do**
    $i \leftarrow i + 1$
    $draw \leftarrow draw - \lambda_i$
return $Model\,[i]$

Example 1.6. *Let us consider two Mallows models ($p = 2$) with parameters: $\langle \sigma_1 = b \succ c \succ a, \phi = 0.19, \lambda_1 = 0.22 \rangle$ and $\langle \sigma_2 = b \succ a \succ c, \phi = 0.34, \lambda_2 = 0.78 \rangle$. We start by computing the probability distributions for each value of ϕ using Eq. 1.1 as shown in the two tables below:*

$\phi = 0.19$			
	$i = 1$	$i = 2$	$i = 3$
$j = 1$	$p_{11} = 1$	$p_{21} = 0.16$	$p_{31} = 0.03$
$j = 2$	-	$p_{22} = 0.84$	$p_{32} = 0.16$
$j = 3$	-	-	$p_{33} = 0.81$

$\phi = 0.34$			
	$i = 1$	$i = 2$	$i = 3$
$j = 1$	$p_{11} = 1$	$p_{21} = 0.26$	$p_{31} = 0.08$
$j = 2$	-	$p_{22} = 0.74$	$p_{32} = 0.24$
$j = 3$	-	-	$p_{33} = 0.68$

Now we choose a model to generate the first ranking following Algorithm 1.1 where Model $= [1, 2]$. We start be considering the first model with $\lambda_1 = 0.22$, assume we take $x = 0.69$ then draw$= 0.69 - 0.22 = 0.47$. Since draw> 0, we take now the second model with $\lambda_2 = 0.78$ and draw $= 0.47 - 0.78 = -0.31 < 0$. Thus, the algorithm terminates and the second model with $\langle \sigma_2 = b \succ a \succ c, \phi = 0.34, \lambda_2 = 0.78 \rangle$ is considered. Now, assume we take the insertion vector $\boldsymbol{j} = (1, 1, 3)$; then $r = \Phi_{\sigma_2}(1, 1, 3) = (a \succ b \succ c)$ where b is inserted first, a is inserted next in the first position by moving b down, finally, c is inserted in the third position.

Among probabilistic analysis used in social choice, there is also an increasing focus on domain restrictions [69, 33, 57]. One of the most common domain restrictions

considered in social choice theory is that of single peaked preferences introduced by Black [11]. Single-peakedness is the idea that all voters have a certain most-preferred value along a shared axis, and derive less and less satisfaction from values that are further away from their preferred one.

- *Single peaked impartial culture (SP-IC)*: Generates singled peaked votes uniformly. The model draws rankings uniformly at random from the set of orders that are consistent with a given social axis (a fixed order over candidates). In general, given m candidates at integer points on the social axis form 1 to m, there are 2^{m-1} different singled peaked votes. Walsh [72] describes how to generate singled peaked votes uniformly. He shows that half of all these single peaked votes end in m and are made up of all the single peaked votes from 1 to $m-1$ augmented with m at their end. The other half of these single peaked votes end in 1 and are made up of all the single peaked votes from 2 to m augmented with 1 at their end.

  Example 1.7. *Let us consider a set of 3 candidates and 3 different social axis* $a \succ b \succ c$, $b \succ a \succ c$ *and* $c \succ a \succ b$. *Table 1.4 presents the probability distributions with respect to different social axis under SP-IC where the possible ranking are in the first row and social axis are in the first column. There are* $2^{m-1} = 2^2 = 4$ *consistent single peaked orders with an uniform distribution of* $\frac{1}{4}$.

  For example, given the social axis $a \succ b \succ c$, *half of the single peaked votes are made up of candidates* a *and* b, *and augmented with* c *at their end, i.e. rankings* $a \succ b \succ c$ *and* $b \succ a \succ c$. *The other half of the single peaked votes end in* a *and are made up of votes containing* b *and* c, *i.e. rankings* $b \succ c \succ a$ *and* $c \succ b \succ a$.

	$a \succ b \succ c$	$a \succ c \succ b$	$b \succ a \succ c$	$b \succ c \succ a$	$c \succ a \succ b$	$c \succ b \succ a$
$a \succ b \succ c$	1/4	0	1/4	1/4	0	1/4
$b \succ a \succ c$	1/4	1/4	1/4	0	1/4	0
$c \succ a \succ b$	1/4	1/4	1/4	0	1/4	0

Table 1.4: Single peaked rankings under SP-IC model with the social axis: $(a \succ b \succ c)$, $(b \succ a \succ c)$ and $(c \succ a \succ b)$.

1.5 Conclusion

In this chapter we have introduced basic notions of voting model used in preference aggregation. We have also presented the most important families of voting rules used

to select the winner based on the voters' preferences. Finally, we have discussed several distributional preference models used to generate preferences based on a prior distribution. In the next chapter, we focus on determining winners or making decisions with incomplete information about voters' preferences.

Chapter 2

Voting with Partial Preferences

Contents

In this chapter we relax the assumption that each voter provides a complete preference order over candidates. We focus on determining the winner or making decisions with incomplete information about voters' preferences. We describe the basic notations of voting with partial preferences. Then, we discuss existing solutions to determine the winners with partial voters' preferences. Finally, we tackle the problem of preference elicitation where we present existing work on the two types of vote elicitation, namely: interactive elicitation and non-interactive elicitation.

2.1 Introduction

Classical voting rules discussed in Chapter 1 assume that voters' ballots are complete preference orders over candidates (except plurality and approval voting). The process is easy when all the voters submit all their preferences to the voting center. However, asking voters to submit a complete ranking over the whole set of candidates is not viable for several reasons:

- when the number of candidates is large enough, requiring a complete specification of the voter's preference can be difficult and too costly in terms of time and/or cognitive effort.
- voters may wish to rank only a subset of candidates as it is not always possible for a voter to compare all available alternatives.
- voters may have privacy concerns which prevent them from revealing their full preference rankings.

This raises the challenge of partial voting where agents are allowed to cast partial preference orders.

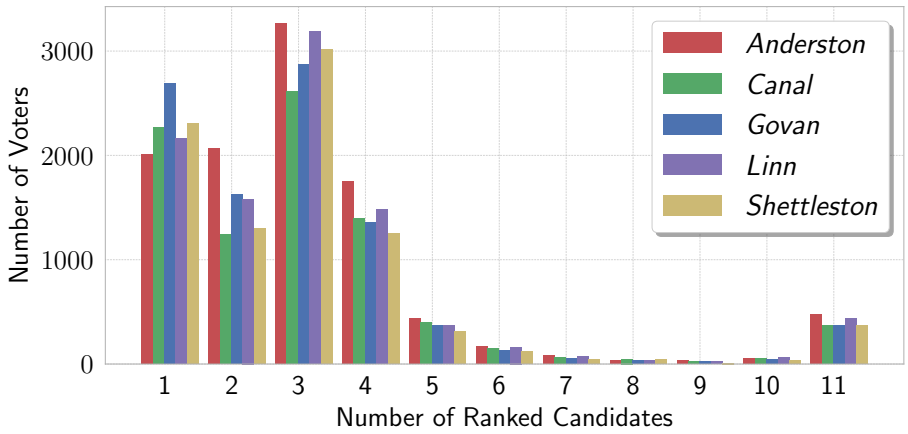


Figure 2.1: In the Glasgow election City Council, most of the voters rank only 1 to 4 candidates out of 11.

In practice, voting systems often permit voters to declare a partial order over a subset of the candidates. For example, in the 2002 Election for Dublin West, 9 candidates ran, but 29,988 voters ranked only a median of 4 candidates over the set of candidates, and only 12.7% of the voters cast a complete vote. Also, in the 2007 Glasgow City Council

elections separated by wards, available ballot data report the results of all the ward level elections which were originally held under STV. Figure 2.1 shows the number of ranked candidates by (around $10,000$) voters for 5 wards when they were asked to rank 11 candidates. Most voters chose to rank between 1 and 3 of the 11 candidates.

Partial voting can have a significant effect on elections [34]. For example, in elections for the Tasmanian Parliament, voters are allowed to submit partial ranking where they rank a minimum number of candidates; usually this number represents the number of members to be elected. Then, the Hare-Clark system is applied.[1] This voting system is also know as the *Proportional Representation through the Single Transferable Vote* (PR-STV) which corresponds to the multi-winner version of STV.

The chapter by Boutilier and Rosenschein [13] of *the Handbook of Computational Social Choice* provides background on work on incomplete information and communication in voting. In this chapter, we address the problem of partial preferences where: In Section 2.2 we describe the basic concepts about incomplete preferences. Section 2.3 introduces solution concepts to determine the winners with partial information: possible and necessary winners, regret based winner computation and probabilistic models. Finally, Section 2.4 describes elicitation techniques and their analysis.

2.2 Partial Votes and Profiles

In order to reduce the communication requirements of voting, a natural way consists in obtaining partial information about voters' preferences. Incomplete information represents a challenging problem in preference aggregation. We begin by introducing basic notations and defining several models of partial preferences that will be used throughout this chapter.

Formally, an incomplete vote is defined by an election $E' = (N, A, R)$ where:

- $N = \{1, ..., n\}$, the set of *voters* (or agents),
- $A = \{a, b, ...\}$, the set of *candidates* (or alternatives) such that $|A| = m$, and
- $R = \{R_1, \ldots, R_n\}$, the *partial profile* of voters in N over A where each R_i is a *partial order* of voter i over A. We will refer to R_i either as a partial vote or an incomplete vote; we use these terms interchangeably.

[1] see https://en.wikipedia.org/wiki/Hare-Clark_electoral_system

A partial vote R_i can represent information about voter i's responses to queries. The most natural queries that can be represented as a partial preference order are[2]:

- *Pairwise comparisons*: Voter i states which of two alternatives, a or b, is preferred to the other. For instance, given $A = \{a, b, c, d\}$, voter i prefers a to b and c to d, i.e. $a \succ b$ and $c \succ d$.
- *Top-k queries*: Voter i provides the *top-k* (ranked) alternatives from his complete preference order $\succ_i$ for some $1 \leqslant k \leqslant m$. For instance, given $A = \{a, b, c, d\}$ and $\succ_i = a \succ b \succ c \succ d$, the top-3 preferred candidates of i are: $a \succ b \succ c$.
- *Next best alternative*: Voter i states her next most preferred alternative: her k^{th} ranked candidate assuming that her $k-1$ candidates have already been provided (the next best alternative of voter i and her previous votes corresponds to answering a top-k query).
- *Set choice (the same S)*: All voters are presented with the same subset $S \subset A$, and asked to give complete or partial rankings over the candidates in S.
- *Set choice (not the same S)*: Each voter i is presented with a subset $S \subset A$ (not the same for all voters), and asked to give complete or partial ranking over the candidates in S.

A *completion* of R_i is any vote $\succ_i$ that extends R_i for each $i \in \{1, \ldots, n\}$. Let $C(R_i)$ denote the set of *completions* of R_i, that is, the set of all complete linear order $\succ_i$ that extend R_i. A partial profile is a collection of partial votes $R = \{R_1, \ldots, R_n\}$. Let $P = C(R) = C(R_1) \times ... \times C(R_n)$ be the set of completions of R.

Example 2.1. *Let us consider a partial profile $R = \{R_1, R_2, R_3\}$ of three voters who were asked to state a pairwise comparison between three candidates $A = \{a, b, c\}$. The partial preferences of the voters are shown in Figure 2.2. For example, voter 1 prefers $a \succ b$ and $a \succ c$, but he/she expresses no preference between candidates b and c.*

Figure 2.3 presents the different possible completions of the partial profile R in Figure 2.2 to complete preferences orders where $C(R)$ contains 12 profiles.

For example, a possible way to complete R_1 (resp. R_2, R_3) is by adding $b \succ c$ (resp. $b \succ a$, $b \succ a$ and $c \succ a$). As a result, $C(R_1) = \{a \succ b, a \succ c, b \succ c\}$, giving the following complete linear profile $P = \{a \succ b \succ c, b \succ a \succ c, b \succ c \succ a\}$.

[2]The work of Briskorn *et al.* [16] includes a complete classification of types of incomplete information

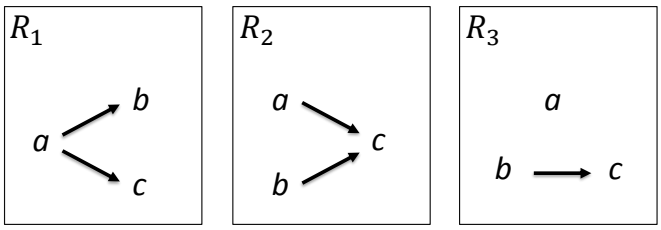


Figure 2.2: Example of a partial preference profile.

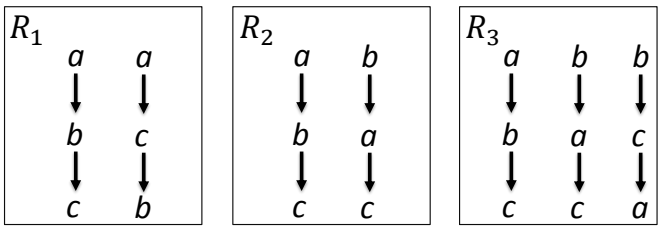


Figure 2.3: Completions of the partial preference profile in Figure 2.2.

2.3 Solution Concepts with Partial Preferences

In voting situations where the voters' preferences are partly known, a fundamental question arises: *Is the information at hand sufficient to determine a collective decision?* Several researches have focused on determining the winner given only incomplete voters' preferences where different solution concepts were considered to deal with such problem [7, 42, 46, 54]: the use of *possible* and *necessary winners*, *minimax regret* and *probabilistic models*.

2.3.1 Possible and Necessary Winners

The partial information about the voters' preferences may or may not be sufficient to output the winner for a specific voting rule f. In the case where the winner cannot be determined, the information at hand may inform us on the presence of certain candidates to be potential winners. In this context, Konczak and Lang [46] introduced the concept of possible and necessary winners denoted PW and NW, respectively; where:

- a candidate c is a *possible winner* for a partial profile R and a voting rule f if c wins under at least one completion of the partial votes in R, i.e. $C(R)$, and

- a candidate c is a *necessary winner* for a partial profile R and a voting rule f if it wins for all completions of the partial votes in R.

  If f is resolute, then there can be at most one necessary winner. Also, x is necessary winner if and only if it is the only possible winner.

Note that knowing the possible winners given a partial profile allows to identify candidates that cannot win called *necessary losers*. By definition, candidate x is a *necessary loser* if it is not a possible winner.

Example 2.2. *In Figure 2.2, the partial voter's preference R_1 (resp. R_2, R_3) can be completed by adding $b \succ c$ (resp. $b \succ a$, $b \succ a$ and $c \succ a$), resulting in the following linear complete profile: $P = \{a \succ b \succ c, b \succ a \succ c, b \succ c \succ a\}$ (other possible completions of R are in Figure 2.3). Under Borda rule, candidate a (resp. b, c) gets 3 (resp. 5, 1) points. Then, candidate b is a possible winner when considering the latter completion.*

Another possible extension of R_1 (resp. R_2, R_3) is by adding $b \succ c$ (resp. $b \succ a$, $a \succ b$ and $a \succ c$), resulting in the following linear profile: $P = \{a \succ b \succ c, b \succ a \succ c, a \succ b \succ c\}$. Under Borda rule, candidate a (resp. b, c) gets 5 (resp. 4, 0) points. Then, candidate a is a possible winner when considering the latter completion.

Note that given the partial profile R, there is no necessary winner because there are two possible winners.

Possible and necessary winners problems have gained a significant attention [8, 26, 69, 75]. Complexity of these problems depends on the voting rule used and the setting. Three main settings have been studied for general partial profiles:

1) the number of candidates is bounded and the votes are weighted i.e. a voter with integer weight α can be viewed as α voters voting identically.[3] The PW problem is NP-complete for STV, plurality with runoff, Borda, Maximin and Copeland rules. The NW problem is co-NP-complete for STV and plurality with runoff [26, 49, 62, 69].

2) the number of candidates is bounded and the votes are unweighted i.e. each voter's vote counts the same. The possible and necessary winner problems can always be solved in polynomial time, assuming the voting rule can be solved in polynomial time [26, 69].

[3]Weighted voting systems occur when some voters are more important than others and have more decision-power. These systems are used in a number of real-world settings like shareholder meetings and elected assemblies.

3) the number of candidates is unbounded and the votes are unweighted. Bartholdi and Orlin [8] show that for STV, PW is NP-complete and NW is co-NP-complete. Xia and Conitzer [75] show that: PW and NW are polynomial time solvable for Plurality. For Copeland and Ranked Pairs, PW is NP-complete and NW is co-NP-complete. For Maximin, PW is NP-complete but NW is polytime solvable. For positional scoring rules (including Borda), PW is NP-complete while NW is in P. The easiness of the necessary winner problem is due to the use of algorithms that find a completion of the partial profile that refute the claim.

2.3.2 Minimax Regret

Another way considered to determine the winner given a partial profile R is the use of *minimax regret* (MMR) concept proposed by Lu and Boutilier [54]. This measure is used to determine the quality of a proposed winner $x \in A$ given the information at hand by computing how far from the optimal solution could be given any completion of R in the worst case called the *maximum regret* of a candidate x (denoted by $MR(x, R)$). Then, the *minimax optimal solution* (denoted by x_R^*) is the candidate with the minimum MR. Formally, given a partial profile R:

$$MMR(R) = \min_{x \in A} MR(x, R) \tag{2.1}$$

$$x_R^* \in \operatorname*{argmin}_{x \in A} MR(x, R) \tag{2.2}$$

where:

$$MR(x, R) = \max_{P \in C(R)} Regret(x, P) \tag{2.3}$$

$$Regret(x, P) = \max_{y \in A} S(y, P) - S(x, P) \tag{2.4}$$

and $S(x, P)$ is the score of candidate x given P.

where $Regret(x, P)$ represents the *regret* of selecting x as the winner for a specific voting rule f rather than another candidate $y \in A$ given $P = C(R)$. Note that if $MMR(R) = 0$, then the minimax winner x_R^* is optimal in any completion $P \in C(R)$.

Example 2.3. *Let us consider a partial profile with $A \in \{a, b, c\}$ and $n = 3$ illustrated in Figure 2.4.*

We compute the maximum regret for different candidates in A under Borda rule:

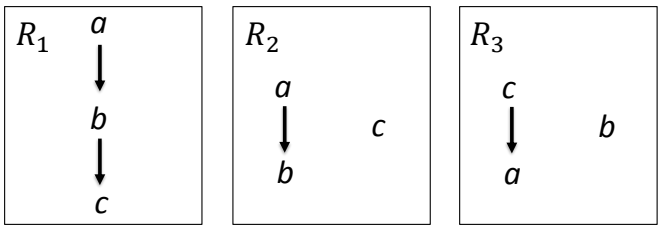


Figure 2.4: Example of a partial profile where candidate a is the optimal minimax regret following Lu and Boutilier [54].

- *candidate a: we consider the worst-case completion of the partial profile R by adding $c \succ a$ and $c \succ b$ (resp. $b \succ a$ and $c \succ b$) to R_2 (resp. R_3) giving $C(R) = \{a \succ b \succ c, c \succ a \succ b, c \succ b \succ a\}$. The scores of different candidates are 3, 2 and 4 for a, b and c; respectively. Then, $MR(a, R) = S(c, C(R)) - S(a, C(R)) = 4 - 3 = 1$.*

- *candidate b: we consider the worst-case completion of the partial profile R by adding $c \succ b$ and $a \succ c$ (resp. $a \succ b$ and $c \succ b$) to R_2 (resp. R_3) giving $C(R) = \{a \succ b \succ c, a \succ c \succ b, c \succ a \succ b\}$. The scores of different candidates are 5, 1 and 3 for a, b and c; respectively. Then, $MR(b, R) = S(a, C(R)) - S(b, C(R)) = 5 - 1 = 4$. Note that if we consider the completion $C(R) = \{a \succ b \succ c, c \succ a \succ b, c \succ a \succ b\}$ then $MR(b, R) = 3$, which do not correspond to the maximal MR.*

- *candidate c: we consider the completion: $C(R) = \{a \succ b \succ c, a \succ b \succ c, c \succ a \succ b\}$. The scores of different candidates are 5, 2 and 2 for a, b and c; respectively. Then, $MR(c, R) = S(a, C(R)) - S(c, C(R)) = 5 - 2 = 3$. Also, if we consider the completion $C(R) = \{a \succ b \succ c, a \succ b \succ c, b \succ c \succ a\}$ then $MR(c, R) = 3$.*

Hence, $MMR = 1$ and x_R^* is candidate a.

2.3.3 Probabilistic Concepts

Bachrach *et al.* [7] propose a probabilistic notion for possible winner candidates. Given a partial profile R, they compute a proportion of completions of R (i.e. $C(R)$) for which a specific candidate is a possible winner. They assume that voters' preferences are drawn according to a uniform distribution (i.e. impartial culture, cf. Chapter 1) and sample completions using the latter. They show the effectiveness of their algorithm

to provide proportion of the sampled profiles with high probability for voting rules computed in polynomial time.

A different approach was investigated by Hazon *et al.* [42] using probabilistic models to deal with incomplete information. Indeed, authors study the possibility of computing the probability of a particular candidate winning an election given partial information in the form of probability distributions over a set of candidates. Hazon *et al.* propose an algorithm that allows to compute the probability of candidate's victory under a variety of voting rules. For a constant bounded number of alternatives, they show that computing the probability of a winning candidate is polytime solvable for any voting rule solved in polynomial time. However, when the number of candidates is not bounded, they show that the problem becomes P-hard for plurality, Borda and Copeland rules.

Example 2.4. *Let us consider the following profile over a set of 3 candidates* $A = \{a, b, c\}$ *and 2 voters. Each voter* i*'s preference is associated with a probability and the sum of the probabilities is equal to 1.*

$Voter\ 1$	$Voter\ 2$
$\frac{1}{2}\ a$	$\frac{1}{4}\ a$
$\frac{1}{3}\ b$	$\frac{3}{4}\ b$
$\frac{1}{6}\ c$	

Table 2.1: Example of a partial profile with probability distributions over preferences following Hazon *et al.* [42].

$Voting\ results\ (a, b, c)$	$Probability$
$1, 1, 0$	$\frac{1}{3}\cdot\frac{1}{4} + \frac{1}{2}\cdot\frac{3}{4}$
$1, 0, 1$	$\frac{1}{6}\cdot\frac{1}{4}$
$0, 1, 1$	$\frac{1}{6}\cdot\frac{3}{4}$
$0, 2, 0$	$\frac{1}{3}\cdot\frac{3}{4}$
$2, 0, 0$	$\frac{1}{2}\cdot\frac{1}{4}$
$0, 0, 2$	0

Table 2.2: Probability's table with probabilistic preferences following Hazon *et al.* [42] algorithm.[4]

[4]The first row represents a *voting result* which is a vector of votes of the possible voting scenarios for candidates a, b and c, e.g. vector $(2, 0, 0)$ means that 2 votes are given to candidate a and 0 vote to b and c. The second column shows the probabilities for each possible voting scenario that consists in the product of the probabilities of the voters' preference, e.g. the probability associated to b is $\frac{1}{3} \times \frac{3}{4}$.

The voters' preferences are presented in Table 2.1. For instance, voter 2 will vote to candidate a (resp. b) with a probability of $\frac{1}{4}$ (resp. $\frac{3}{4}$) and she will not vote for candidate c. Following Hazon *et al.*'s approach, Table 2.2 presents the generated probabilities where the probability that candidate a (resp. b, c) is the winner for plurality rule is: $\frac{1}{2}.\left(\frac{1}{3}.\frac{1}{4}+\frac{1}{2}.\frac{3}{4}\right)+\frac{1}{2}.\left(\frac{1}{6}.\frac{1}{4}\right)+\frac{1}{2}.\frac{1}{4}=0.375$ (resp. 0.542 and 0.083).

2.4 Preference Elicitation

In previous section we described techniques to determine the winner given partial preferences. Such solution concepts may succeed to output the winner with incomplete ballots; however, they do not provide ways to minimize the amount of information elicited. Effectively eliciting preferences from voters is of high importance. This problem is known as *preference elicitation* which was investigated in different domains such as healthcare [74], recommender systems [51], crowdsourcing [22] and marketing [43] where the system asks questions to different users in order to learn about group's preferences. In voting situations, *preference elicitation* refers to elicit a partial profile from voters in an efficient way in order to minimize the amount of information required to determine a winning outcome of high quality.

A voting rule does not specify how the votes are elicited from the voters (these rules are just functions mapping the preferences of all the voters to a winner). Different ways of determining the winner of an election by eliciting preferences from voters are based on specific *protocols*. A protocol is similar to an algorithm, with instructions replaced by communication actions; such actions specify bits that the voter should communicate, depending on her knowledge. Formally, a *protocol for a voting rule* f is a protocol that computes $f(\succ_1,...,\succ_n)$, given that $\succ_i$ is the private information of voter i. The *(deterministic) communication complexity of a voting rule* f is the minimum cost of a protocol for f.

A communication protocol is held between two parts: the voters and the voting center where the former send their preferences to the latter who is responsible for collecting the votes, computing the final result, and communicating the final outcome to the voters. Efficient communication protocols allow to determine how the relevant parts of the votes are elicited from the voters in the aim to minimize the communication cost of the protocol. For instance, for plurality with runoff, one possible way of eliciting the votes is to ask every voter to report her complete preference order. Alternatively, a simpler protocol is held in three steps: (1) each voter sends only the name of her most-preferred candidate to the voting center, (2) the voting center sends the name of

the two finalists to the voters and (3) voters send the name of their preferred finalist to the voting center.

Example 2.5. *Let us consider a setting of 15 voters with the following profile P over three candidates $A = \{a, b, c\}$:*

6 votes	$a \succ b \succ c$	*3 votes*	$c \succ b \succ a$
5 votes	$b \succ a \succ c$	*1 votes*	$c \succ a \succ b$

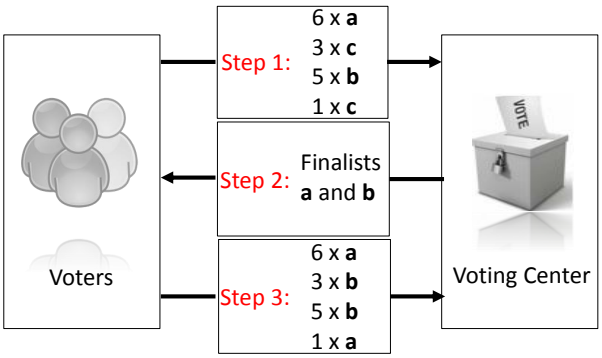


Figure 2.5: Communication protocol for plurality with runoff.

Figure 2.5 shows the communication protocol for plurality with runoff. In the first step, voters send their most preferred candidate where candidate a (resp. b and c) gets 6 (resp. 5 and 4) votes. In the second step, the voting center sends to the voters the candidates who received the highest scores using plurality, i.e. candidates a and b. Finally, in the third step, voters send their preferred candidate among the two finalists and candidate b is declared the winner.

One line of research focuses on the computational complexity of various issues related to elicitation. Conitzer and Sandholm [25] study the communication complexity of various voting rules and determine upper and lower bounds for communication. They show that an upper bound (resp. lower bound) of $‰(n \log m)$ (resp. $\Omega(n \log m)$) bits are needed for plurality and plurality with runoff. For Borda, Copeland and ranked pairs an upper bound (resp. lower bound) of $‰(n\ m\ log\ m)$ (resp. $\Omega(n\ m\ log\ m)$) bits are needed. For STV, an upper bound (resp. lower bound) of $O(n(log\ m)^2)$ (resp. $\Omega(n \log m)$) bits of communication are needed. (The communication complexity of STV will be discussed in Chapter 4.) Walsh [70] considers the question of how to decide when we can terminate eliciting preferences from the voters as the winner is determined. This problem is related to the computation of possible and necessary winners. Walsh shows that the complexity of deciding when to stop elicitation depends

on the elicitation strategy. Conitzer and Sandholm [24] show that determining how to elicit effectively voters preferences is NP-complete for Borda, Copeland, Maximin and STV while it is easy for plurality.

We now describe several techniques for the elicitation of voter preferences. Existing work on vote elicitation using partial profiles can be classified into two classes according to the type of interaction with the voters, namely: interactive elicitation and non-interactive elicitation (or one-shot elicitation). Some of work on these two classes uses the techniques described in Section 2.3 to guide the process of elicitation. These two approaches are described in the next two subsections.

2.4.1 Interactive Elicitation Protocols

An interactive elicitation protocol asks voters to expand their partial ballots in an incremental way, until the outcome of the vote is eventually determined. Recently, work on interactive elicitation received a considerable attention where algorithms that work well in practice have been proposed to elicit the relevant part of the voters' preferences. Authors consider different forms of queries through the elicitation process such as top-k, next best alternative, set of candidates and pairwise comparison queries.

This line of research starts with Kalech *et al.* [44] who consider top-k ballots in vote elicitation. Authors propose an iterative algorithm that allows the voters to cast their preferences incrementally starting by top-1 ballots, then top-2, etc., until there is sufficient information for knowing the winner using possible and necessary winner concept. Experiments on real data show that their deterministic algorithm is able to reduce the communication cost by 35% and determine the winner correctly. To further reduce the information elicited, Kalech *et al.* propose an approximation algorithm able to save up 90% of preferences communicated. While there is no guarantee to output the correct winner, authors show that in practice their heuristic predicts the correct outcome with high probability.

Lu and Boutilier [54] expand their work on determining the winner given partial preferences, using minimax regret solution, to vote elicitation. They propose a multi-round incremental elicitation process that uses minimax regret to determine the winner given partial profile and to guide the choice of the voter to query. They consider two forms of queries: pairwise comparisons and next best alternative. The elicitation scheme of Lu and Boutilier [54] asks one query of a single voter at each round while the scheme of Kalech *et al.* [44] asks at each round all voters to answer a particular query simultaneously which can effectively reduce the number of rounds.

Naamani Dery *et al.* [30] present two elicitation algorithms for finding a winner with little communication between voters. They show that their algorithms are able to reduce the communication between voters by half. In a recent work, Naamani Dery *et al.* [29] take preference elicitation a step further by making the connection between manipulation (or strategic voting) and incremental preference elicitation. Indeed, when eliciting preferences from voters in an incremental way and query them to cast their preferences between two available candidates, voters may have an intention to vote strategically and misreport their preferences in order to make a specific candidate win or lose. Naamani Dery *et al.* provide a practical elicitation algorithm to avoid such scenarios to occur when considering pairwise comparison queries.

While most previous work studied a specific type of queries to elicit from voters, Zhao *et al.* propose a more general incremental elicitation framework, with more types of elicitation questions, is cost-effective elicitation [77]. Indeed, the voting center may ask a voter not only for her top-k preferred alternatives but also about her preference between two alternatives or her set of preferred candidate. In each round, the question asked depends on the missing information that will be used to reveal the most relevant information about the voter's preferences. In their work [77], Zhao *et al.* show how to compute the cost-effective questions in a preference elicitation framework.

2.4.2 One-Shot Elicitation Protocols

In one-shot elicitation (aka. non-interactive elicitation) the voting center elicits the partial information at once and outputs a winner from the available information without requiring voters to provide extra preferences. Different forms of partial vote have been considered including: top-k preferences, pairwise comparison between candidates or the set of the voters' preferred candidate. A possibility consists in computing possible winners given these partial vote: this is the path followed by Baumeister *et al.* [9] who consider truncated ballots of the top-k, bottom-k and doubly truncated votes with top and bottom segments provided. Given these specific form of partial preferences, Baumeister *et al.* propose ways of extending some voting rules to deal with these partial preferences. Xia and Conitzer [75] consider partial vote in the form of pairwise comparisons between candidates. They propose polynomial-time algorithms to check whether a given candidate is a necessary winner for positional scoring rules, Maximin and plurality with runoff[5] given partial profile.

[5]For plurality with runoff, the unique necessary winner is in P; however, the necessary co-winner is coNP-complete.

Another possibility – which is the one we choose to follow in Chapter 3– consists in generalizing the definition of a voting rule so that it takes partial vote as input. In this line, Oren *et al.* [61] analyze *top-k* voting by assessing the values of k needed to ensure the true winner is found with high probability for specific preference distributions, both theoretically and empirically. Their theoretical results were derived in both a worst and an average case under impartial culture. Although the latter are a little bit discouraging, their empirical results on Mallows model show that small values of k work well in practice. Filmus and Oren [37] follow the same path of research where they study the performance of top-k voting under the impartial culture distribution (IC) for Borda, Harmonic and Copeland rules. They assess the values of k needed to ensure the true winner with high probability. More precisely, they show that for Borda and Harmonic (resp. Copeland) voting a lower bound of $k = \Omega(m)$ (resp. $k = \Omega\left(\frac{m}{\sqrt{\log m}}\right)$) is needed for n sufficiently large relative to m. Their theoretical analysis is completed by numerical experiments showing that under the impartial culture, in the setting where $m = 20$ and $n = 2000$, Harmonic rule gives the best results where $k = 15$ out of 20 is sufficient to determine the winner, while for Copeland and Borda the whole profile is needed to ensure the winner.

Narodytska and Walsh [59] consider the computational impact of partial vote on strategic voting when voters submit *top-k* ballots. For Borda rule, they consider two possible ways to deal with truncated ballots by adjusting the scoring vector, namely: *modified Borda Count* [35] and *average score modified Borda Count* [27, 40] (these two options will be considered in Chapter 3). For STV, they simply ignore votes once all their candidates are eliminated (this way to deal with truncated ballots for STV will be considered in Chapter 4). They show that under STV and *average score modified Borda Count*, partial voting does not change the situations where strategic voting is possible. However, under *modified Borda Count* partial voting can increase the situations where strategic voting is possible. Lu and Boutilier [55] propose a general framework for choosing the optimal threshold k for *top-k* elicitation in one-round protocols using minimax regret. Skowron *et al.* [67] use *top-k* voting as a way to approximate the Monroe and Chamberlin–Courant multi-winner voting rules. Their experimental evaluation on synthetic data and real word data set show that their algorithm is able to output the near-perfect solution. Lackner [48] studies the recognition of singled-peaked *top-k* profiles by proposing an algorithm to determine whether a given partial profile can be completed in such a way that it is single-peaked.

2.5 Conclusion

In this chapter we have relaxed the assumption that each voter provides a complete preference ranking over all candidates by dealing with incomplete voters' preferences. We have described the basic notations of voting with partial preferences. Then, we have discussed existing solution concepts for winner determination with incomplete ballots. Finally, we have tackled the problem of preference elicitation where we have discussed existing work on the two types of vote elicitation, namely: interactive elicitation and non-interactive elicitation. In the next chapters we propose ways to deal with partial preferences in the form of *top-k* ballots for different voting rules.

Part II
Contributions

Part II presents the contributions of this dissertation. Chapter 3 proposes "top-k approximations" of rules, which take only into account the top-k candidates of each ballot. Chapter 4 focuses on STV voting rule and proposes ways to minimize the amount of communication required to use single-winner STV. Finally, Chapter 5 proposes an incremental vote elicitation process using heuristics to guide the choice of the next voter to ask.

Chapter 3

Approximate Voting Rules from Truncated Ballots

Contents

In this chapter we consider top-k ballots where we suggest to fix a rank k, to ask all voters to specify their best k candidates, and then we propose "top-k approximations" of rules, which take only into account the top-k candidates of each ballot. We consider two measures of the quality of the approximation: the probability of selecting the same

winner as the original rule, and the score ratio. We make a worst-case study (for the latter measure only), and for both measures, an average-case study and a study from real data sets.

3.1 Introduction

The necessity of effectively aggregating voters' preferences is of great importance especially with systems containing large collections of candidates to choose from. We suggest to ask voters to report only their *top-k* candidates, for some (small) fixed value of k (the obtained ballots are then said to be *k-truncated*). The advantage of doing so is that not only it saves communication effort, but it is also often easier for voter to find out the upper part of their ranking (consisting of their preferred k candidates) than the rest of it. However, this raises the issue of how usual voting rules should be adapted to k-truncated ballots.

Reporting *top-k* truncated ballots is a specific form of *voting with incomplete preferences*, and is highly related to *vote elicitation*. More specifically existing work on *top-k* ballots can be classified into two classes according to the type of interaction with the voters (cf. Chapter 2):

- ***Interactive elicitation***: the voting center asks voters to expand their truncated ballots in an incremental way, until the outcome of the vote is eventually determined.
- ***Non-interactive elicitation***: the voting center elicits the *top-k* ballots at once, for a fixed value of k, and outputs a winner from this partial information without requiring voters to provide extra information.

Our contribution in this chapter concerns non-interactive elicitation. We generalize the definition of a voting rule such that it takes truncated ballots as input[1]: we define approximations of voting rules which take as input the *top-k* candidates of each ballot. The question is then: *Are these approximations good predictors of the original rule?* We answer this question by considering two measures:

[1]Note that outputting possible winners can also be seen as a way of generalizing the definition of a voting rule to truncated ballots, but the obtained rule then tends to be very irresolute when the size of the partial ballots is small.

- the probability that the approximate rule selects the 'true' winner. To do so, we sample random profiles of full rankings and truncate the rankings so that voters only rank their *top-k* choices. We then compute the probability that the winner of the *top-k* ballots is the same as the winner of the full ballots for different voting rules.
- the ratio between the scores (for the original rule) of the true winner and the winner of the approximate rule.

For the latter measure we give a worst-case theoretical analysis. For both measures we give an empirical study, based on randomly generated profiles and on real-world data. Even for small k, we find that *top-k* ballots are enough to identify the correct winner quite frequently for different voting rules, especially for data taken from real elections.

This contribution can be seen as a continuation of Filmus and Oren [37] (cf. Chapter 2, Section 2.4.2). We go further on several points: we consider more voting rules: Borda, Harmonic, Copeland, Maximin, ranked pairs and STV; beyond impartial culture, we consider a large scope of distributions; within the *Mallows* ϕ model [56] and Mixture of Mallows [53]. We also study score distortion; and we include experiments using real-world data sets.

Our interpretation of *top-k* ballots is *epistemic*: the central authority in charge of collecting the votes and computing the outcome ignores the voters' preferences below the *top-k* candidates of each voter, and has to cope with it as much as possible. Voters may very well have a complete preference order in their head (although it does not need to be the case), but they will simply not be asked to report it. These truncated ballots certainly do not mean that voters see the candidates below the *top-k* as being indifferent or incomparable.

This chapter is organized as follows: Section 3.2, defines the *top-k* approximations of different voting rules. Section 3.3 analyses empirically the probability that the approximate rule selects the true winner. Finally, Section 3.4 analyses (theoretically and empirically) the score distortion.

3.2 Approximating Voting Rules from Truncated Ballots

We consider *one-shot protocols* where all input information needs to be gathered at the same time. In this model, if we wish to compute the winner for a given voting rule with

certainty, we need to ask voters to report their entire preferences, i.e. to report their linear order. Let us relax the goal of certainty and instead aim to compute the winner with high-enough probability. In exchange, we ask for less information: Voters report *top-k-ballots* for a fixed $k \geqslant 1$, i.e. they report a ranked list of their k most-preferred candidates.

We define a *top-k* election as follows:

Definition 3.1. *Given $k \in \{1, ..., m-1\}$, a* top-k *election is a triple $E' = (N, A, R)$ where N and A are as before, and $R = (\succ_1^k, ..., \succ_n^k)$, where each $\succ_i^k$ is a ranking of k out of m candidates in A. R is called a* top-k profile*. If P is a complete profile, $\succ_i^k$ is the top-k truncation of $\succ_i$ (i.e. the best k candidates, ranked as in $\succ_i$), and $P_k = (\succ_1^k, ..., \succ_n^k)$ is the top-k-profile induced from P and k.*

Example 3.1. *Let us consider a profile of 6 voters with the following complete preferences over $A = \{a, b, c, d\}$:*

2 votes	$b \succ a \succ c \succ d$	*1 vote*	$a \succ d \succ b \succ c$
2 votes	$c \succ a \succ b \succ d$	*1 vote*	$d \succ b \succ c \succ a$

The 2-truncated profile induced from the above preferences is as follows:

2 votes	$b \succ a$	*1 vote*	$a \succ d$
2 votes	$c \succ a$	*1 vote*	$d \succ b$

A *top-k (resolute) voting rule* is a function f_k that maps each *top-k* election E' to a candidate in A. If we apply a *top-k* voting rule to a complete profile P this implies that we will use the *top-k* truncated ballots induced from P, i.e. $f_k(P) = f_k(P_k)$.

In this section we define several *top-k* rules that correspond to standard voting rules discussed in Chapter 1. We consider positional scoring rules (PSR), namely: Borda, Harmonic in Section 3.2.1; pairwise rules, namely: Copeland, Maximin and ranked pairs in Section 3.2.2, and one elimination based rule namely STV in Section 3.2.3.

3.2.1 Positional Scoring Rules

An extension of positional scoring rules to *top-k* voting as follows:

Definition 3.2. *A top-k PSR f_k^s is defined by a scoring vector $s = (s_1, s_2 \ldots, s_k, s^*)$ such that $s_1 \geqslant s_2 \geqslant ... \geqslant s_k \geqslant s^* \geqslant 0$ and $s_1 > s^*$. Each candidate in a top-k vote receives s_j points from each voter i who ranks her in the j^{th} position. A non-ranked candidate gets s^* points. The winner is the candidate with highest total score.*

When starting from a specific *PSR* for complete ballots, defined by scoring vector $s = (s_1, \ldots, s_m)$, three choices of s^* particularly make sense:

- zero score: the non-ranked candidates get a zero score, i.e. $s^* = 0$
- average score: the non-ranked candidates get the average score of the remaining candidates, i.e. $s^* = \frac{1}{m-k}(s_{k+1} + \ldots + s_m)$
- maximal score: the non-ranked candidates get the score of the candidate ranked in the $(k+1)^{th}$ position, i.e. $s^* = s_{k+1}$.

We denote the corresponding approximate rules as f_k^0 uses the zero score, f_k^{av} uses the average score and f_k^{max} uses the maximal score.

For Borda, the first two choices have been considered in the literature where $Borda_k^{av}$ is known under the name *average score modified Borda Count* [27, 40], while $Borda_k^0$ is known under the name *modified Borda Count* [35]).

Example 3.2. *Let us consider an election over $m = \{a, b, c, d, e\}$ candidates and a partial preference order of a voter i where $a \succ_i b$.*

The scoring vectors for $Borda_2^0$, $Borda_2^{av}$ and $Borda_2^{max}$ are: $\vec{s}_{Borda_2^0} = (4, 3, 0, 0, 0)$, $\vec{s}_{Borda_2^{av}} = (4, 3, 1, 1, 1)$ and $\vec{s}_{Borda_2^{max}} = (4, 3, 2, 2, 2)$, respectively.

The scoring vectors for $Harmonic_2^0$, $Harmonic_2^{av}$ and $Harmonic_2^{max}$ are: $\vec{s}_{Harmonic_2^0} = (1, \frac{1}{2}, 0, 0, 0)$, $\vec{s}_{Harmonic_2^{av}} = (1, \frac{1}{2}, 0.26, 0.26, 0.26)$ and $\vec{s}_{Harmonic_2^{max}} = (1, \frac{1}{2}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, respectively.

Young [76] characterized positional scoring rules by these four properties, which we describe informally (for resolute rules):

- *Neutrality*: all candidates are treated equally
- *Anonymity*: all voters are treated equally

- *Reinforcement*: if P and Q are two profiles (on disjoint electorates) and x is the winner for P and the winner for Q, then it is also the winner for $P \cup Q$.
- *Continuity*: if P and Q are two profiles and x is the winner for P but not for Q, adding sufficiently many votes of P to Q leads to elect x.

These four properties still make sense for truncated ballots. It is not difficult to generalize Young's result to *top-k* PSR:

Theorem 3.1. *A top-k voting rule is a top-k* PSR *if and only if it satisfies top-k, neutrality, anonymity, reinforcement, and continuity.*

Proof. The left-to-right direction is obvious. For the right-to-left direction, let us first define the *top-k-only* property: a standard voting rule is *top-k-only* if for any two complete profiles P, P', if $P_k = P'_k$, then $F(P) = F(P')$. Then (1) a positional scoring rule F is *top-k*-only if and only if $s_{k+1} = \ldots = s_m$ (if this equality is not satisfied, then it is easy to construct two profiles P, P' such that $P_k = P'_k$ and $F(P) \neq F(P')$). Now, assume f_k is a *top-k* rule satisfying neutrality, anonymity, reinforcement, and continuity. Let F be the standard voting rule defined by $F(P) = f_k(P_k)$. Clearly, F also satisfies neutrality, anonymity, reinforcement, and continuity, and due to Young's characterization result, F is a PSR, associated with some vector $(s_1, \ldots, s_m)$. Because F is also *top-k*-only, using (1) we have $s_{k+1} = \ldots = s_m$, therefore, f_k is a *top-k*-PSR. □

3.2.2 Rules Based on Pairwise Comparisons

With incomplete preference orders, comparing all pairs of candidates is not possible due to the lack of information submitted by the voters. Now, given a truncated ballot $\succ_i^k$ and two candidates $a, b \in A$, we say that a dominates b in $\succ_i^k$, denoted by $a >_i^k b$, if one of these two conditions holds:

1) a and b are listed in $\succ_i^k$, and $a \succ_i^k b$;
2) a is listed in $\succ_i^k$, and b is not.

Example 3.3. *For $A = \{a, b, c, d\}$, $k = 2$, and $\succ_i^2 = (a \succ b)$, then a dominates b (condition 1 is satisfied), both a and b dominate c and d (condition 2 is satisfied), but c and d remain incomparable in $\succ_i^2$.*

Now, the notions of majority graph and weighted majority graph (cf. Chapter 1) are extended to *top-k* truncated profiles in a straightforward way. We define the *k-truncated majority graph* and the *k-truncated weighted majority graph*, based on the above description of pairwise comparison, as follows :

Definition 3.3. ***k-truncated majority graph*** *Given a k-truncated profile R, $N_R(a,b) = \#\left\{i, a \succ_i^k b\right\}$ is the number of voters in R for whom a dominates b. The k-truncated majority graph $M_k(R)$ induced by R is the graph whose set of vertices is the set of candidates A and in which there is a directed edge from a to b if the number of voters i such that $a \succ_i b$ is strictly larger than the number of voters j such that $b \succ_j a$ i.e. $N_R(a,b) > N_R(b,a)$.*

Definition 3.4. ***k-truncated weighted majority graph*** *Given a k-truncated profile R, for any tow candidates a, b, $N_R(a,b) = \#\left\{i, a >_i^k b\right\}$ is the number of voters in R for whom a dominates b. The k-truncated pairwise majority matrix (or k-truncated weighted majority graph) $MW_k(R)$ is the $m \times m$ matrix defined by $MW_k(R)(a,b) = N_R(a,b) - N_R(b,a)$.*

Then, given a k-truncated profile $R = (\succ_1^k, \ldots, \succ_n^k)$, the truncated voting rules $Copeland_k$, $Maximin_k$ and RP_k are defined exactly as their standard counterparts $Copeland$, $Maximin$ and RP, starting from the k-truncated (weighted or unweighted) majority graph instead of the standard one.

Example 3.4. *Let us consider a profile of 62 voters where $A = \{a, b, c, d\}$ with the following complete preferences:*

20 votes	$a \succ d \succ c \succ b$	*10 votes*	$b \succ c \succ d \succ a$
15 votes	$c \succ d \succ b \succ a$	*17 votes*	$d \succ c \succ a \succ b$

Figure 3.1 shows the representation of the top-k majority graph for $k = \{1, 2, 3\}$ where for each k, the Copeland winner is shaded. For $Copeland_3$, candidate d is the winner. For $Copeland_1$, candidate a has the majority by beating candidates b (20 to 10), c (20 to 15) and d (20 to 17). When we consider $Copeland_2$, candidate d wins.

Figure 3.2 shows the top-k pairwise majority matrix and the $Maximin_k$ winner for $k = \{1, 2, 3\}$. For $Maximin_3$, the winner is also candidate d with a maximum pairwise majority equal to 12. For $Maximin_1$, the minimum pairwise majority of different candidates are: $S_m(a) = 3$, $S_m(b) = -10$, $S_m(c) = -5$ and $S_m(d) = -3$. Then, candidate a wins with a maximum pairwise comparison equal to 3.

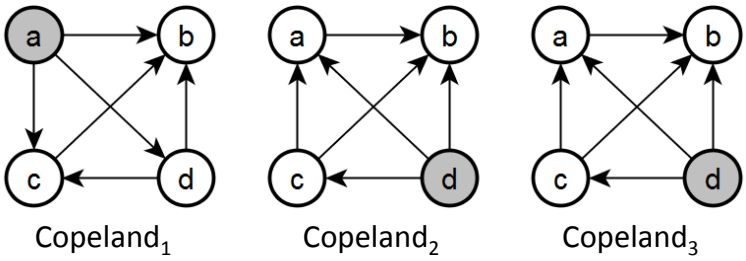


Figure 3.1: k-truncated majority graph, and k-truncated approximations of Copeland for $k = \{1, 2, 3\}$

	a	b	c	d	S_m
a	-	10	5	3	**3**
b	-10	-	-5	-7	-10
c	-5	5	-	-2	-5
d	-3	7	2	-	-3

Maximin$_1$

	a	b	c	d	S_m
a	-	10	-22	-22	-22
b	-10	-	-22	-42	-42
c	22	22	-	-12	-12
d	22	42	12	-	**12**

Maximin$_2$

	a	b	c	d	S_m
a	-	12	-22	-22	-22
b	-12	-	-42	-42	-42
c	22	42	-	-12	-12
d	22	42	12	-	**12**

Maximin$_3$

Figure 3.2: k-truncated pairwise majority matrix, and k-truncated approximations of Maximin for $k = \{1, 2, 3\}$

For $Maximin_2$, the minimum pairwise majority of different candidates are: $S_m(a) = -22$, $S_m(b) = -42$, $S_m(c) = -12$ and $S_m(d) = 12$ and candidate d wins.

For RP, the winner under RP_k for $k \in \{1, 2, 3\}$ is the same as the winner under $Copeland_k$ since the k-truncated majority graph does not create cycles. To illustrate RP_k we consider the following profile with 15 voters and 4 candidates where we break ties using the priority order $\rhd$: $a \succ b \succ c \succ d$:

3	$a \succ d \succ c \succ b$
3	$a \succ d \succ b \succ c$
5	$d \succ c \succ b \succ a$
4	$b \succ c \succ a \succ d$

Figure 3.3 presents the k-truncated weighted majority graph and RP winner for $k \in \{1, 2, 3\}$. For RP_1, all edges are retained since they do not create cycles and candidate a is the winner. For RP_2, we retain edges $(d \to b)$, $(d \to c)$, $(c \to a)$ and $(a \to b)$, we omit $(a \to d)$ since it creates cycle and retain $(c \to b)$. Then the winner is d. For RP_3, we retain edges $(d \to b)$, $(d \to c)$, $(a \to d)$, we omit $(b \to a)$ and $(c \to a)$ and retain $(c \to b)$. Then the winner is a which corresponds to the RP winner.

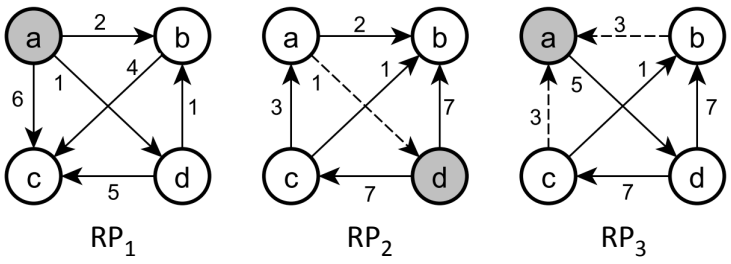


Figure 3.3: k-truncated weighted majority graph, and k-truncated approximations of RP for $k = \{1, 2, 3\}$. The dashed edges are omitted since they create cycles.

3.2.3 Single Transferable Vote Rule

Given *top-k* ballots as input, we propose a natural generalization of STV rule which is quite popular. It is used, for example, in local elections in San Francisco (with $k = 3$). The analogous rule that allows submitting partial rankings of any length (not fixed to some k) is also widely used, for example in the Irish presidential elections, voters are allowed to only cast a *partial* ranking by ranking a subset of the candidates, and leave the rest unranked.

For each $1 \leqslant k \leqslant m$, we define STV_k as follows: Just like STV, in each round, we eliminate a candidate ranked first by the smallest number of voters (breaking ties using $\rhd$ if necessary). If all the k candidates in some ballot have been eliminated, the vote is ignored in later rounds (and is called *exhausted*). We repeat this process until one candidate remains, who is the winner according to STV_k.

Example 3.5. *Let $A = \{a, b, c, d, e\}$ and consider the following 21 top-2 ballots with tie-breaking priority $\rhd$: $a \rhd b \rhd c \rhd d \rhd e$.*

$$\begin{array}{c|l} 6 & a \succ e \\ 5 & d \succ e \\ 4 & c \succ e \\ 6 & b \succ c \end{array}$$

Under STV_2, e is eliminated first, then c. The votes $c \succ e$ are now exhausted. Candidate d is eliminated next. Now, a and b are tied. Finally, a is the STV_2 winner thanks to tie breaking priority.

Although the *top-k* rules can be seen as a voting rule on their own, we view them as an approximation of the standard rule. In this context, we take the *top-k* rule to be

a rule which takes profiles of full linear orders as input, truncates the preferences to become *top-k* ballots, and then proceeds as above.

For all our rules we considered (Borda, Harmonic, Copeland, Maximin, ranked pairs and STV) we observe that:

- for $k = m - 1$ (or equivalently, $k = m$), the k-truncated version of the rule coincides with the standard rule, i.e. $f_{m-1} = f$ (for instance, $Copeland_{m-1}$ coincides with $Copeland$, etc.).
- for all rule f we consider, when $k = 1$, f_1 coincides with plurality.

3.3 Probability of Selecting the True Winner

The first way of measuring the quality of the *top-k* approximations is to determine the probability that they output the 'true winner'; that is, the winner of the original voting rule, under various Mallows ϕ model (Subsection 3.3.1), mixture of Mallows (Subsection 3.3.2) and real-world data (Subsection 3.3.3). We will ask: *How often do the k-truncated rule and the standard one declare the same candidate to be the winner?* Our main practical objective is to obtain, depending on the context, a value of k small enough to allow for painless communication of preferences, but large enough so that the probability of obtaining the true winner from the top-rules is high.

In order to assess the ability of the k-truncated approximations to predict the correct winner, we propose now an empirical approach. Since we want to measure experimentally the probability that the output of the *top-k* approximation of a rule coincides with the true winner, we need to know this true winner, and therefore we need to have access to the true profile. For doing so, we consider the following process:

1) take a complete profile P (with n voters and m candidates);
2) for $k = 1$ to $m - 2$: compare $f_k(P)$ to $f(P)$.

Step 1 varies according to whether we are in the random generation setting or the real world data setting. For the former, we draw a profile according to a given distribution (Subsections 3.3.1 and 3.3.2). For the latter, we draw a profile by selecting n votes uniformly at random from the collection of votes found in the data set (Subsection 3.3.3).

We iterate these two steps a sufficiently large number of times (over 1000 profiles) and we obtain empirically the probability that the original rule and its approximation select the same winner. To approximate Harmonic, we consider the average score version.

Note that we cannot simply stop the process if we reach a k such that $f_k(P) = f(P)$, because there is no guarantee that $f_{k'}(P)$ will still be equal to $f(P)$ for $k' > k$. For instance, consider the profile composed of 5 votes $b \succ a \succ c \succ d$, 4 votes $a \succ b \succ c \succ d$, 3 votes $c \succ d \succ b \succ a$ and one vote $d \succ a \succ b \succ c$: under STV the winner is b. For STV_1, it is b too. However, for STV_2, it is a. Iterating this process a number of times allows us to evaluate the quality of *top-k* rules for different values of k.

Here we follow the research direction initiated by Filmus and Oren [37], but we consider more rules, and beyond *Impartial Culture* we also consider correlated distributions within the *Mallows* model parameterized by a reference ranking σ and a dispersion parameter $\phi \in [0, 1]$, Mixture of Mallows (cf. Chapter 1) and real data sets.

3.3.1 Experiments Using Mallows ϕ Model

For each experiment we draw 1000 random preference profiles. In the first set of experiments, we take $m = 7$, we let n and ϕ vary, and we measure the accuracy of the approximate rule for $k = 1$ and $k = 2$. Results are reported on Table 3.1. Note that for $k = 1$, our results can be viewed as answering the question: *With which probability does the true winner with respect to the chosen rule coincide with the plurality winner?*

For $k = 1$: when $n \leqslant 100$ and $\phi \leqslant 0.7$, prediction reaches 90% for Borda, Copeland Maximin and STV, 92% for RP, and 94% for Harmonic. Better results are obtained as we increase n ($n = 500$): for $n \geqslant 500$ and $\phi \leqslant 0.7$, the accuracy is perfect for all rules. For $\phi = 0.8$, the success rate decreases but results are still good with a large number of voters e.g. when $\phi = 0.8$ and $n = 500$, it is around 0.95 for *Copeland*, *Maximin*, *RP* and STV, and 0.96 for *Borda* and Harmonic. For $\phi = 0.9$ and $n = 500$, the rate reaches 86% for Harmonic and 72% for Copeland, with intermediate (and similar) results for Borda, Maximin, *RP* and STV. For larger ϕ ($\phi = 1$), the accuracy is much lower. Indeed, for the *IC*, the rate decreases dramatically when k becomes small, except for Harmonic (73% when $n = 500$ against 46% for STV, 31% for Copeland and around 40% for the remaining rules).

For $k = 2$: when $\phi < 1$, the accuracy of all truncated rules improves significantly if the number of voters is increased. Indeed, the probability of selecting the true winner reaches 100% (resp. 98%) when $\phi \leqslant 0.7$ (resp. $\phi = 0.8$) and $n \geqslant 400$ (resp. $n \geqslant 500$).

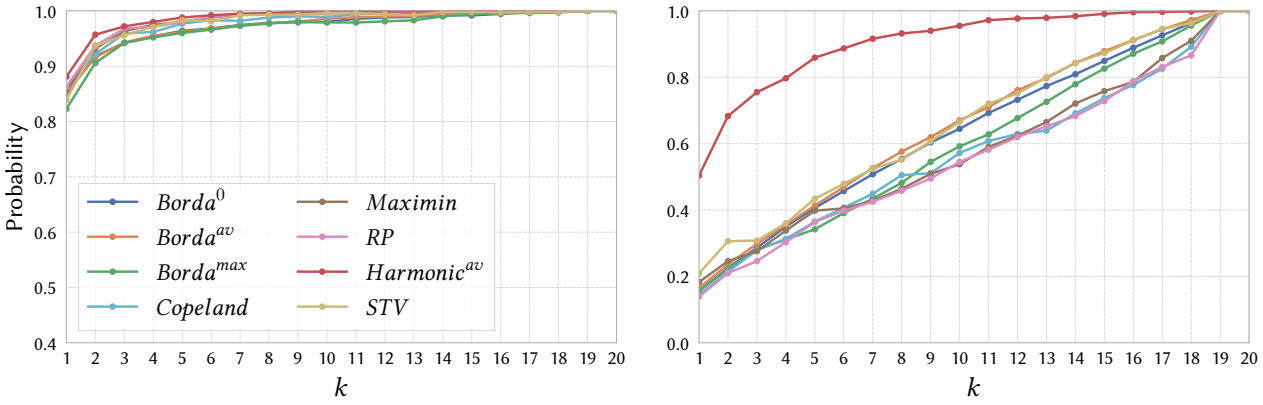
Table 3.1: Success rate, $m = 7$, varying n, k and ϕ.

ϕ	n=100	n=200	n=300	n=400	n=500	n=100	n=200	n=300	n=400	n=500
	$Borda_1^0$					$Borda_2^0$				
0.7	0.902	0.958	0.986	0.992	**1.0**	0.952	0.984	0.997	**1.0**	1.0
0.8	0.77	0.855	0.9	0.94	0.963	0.859	0.92	0.944	0.965	0.988
0.9	0.588	0.694	0.685	0.718	**0.771**	0.722	0.788	0.815	0.854	0.859
1	0.434	0.445	0.424	0.422	**0.397**	0.582	0.571	0.591	0.553	**0.554**
	$Borda_1^{av}$					$Borda_2^{av}$				
0.7	0.902	0.958	0.986	0.992	**1.0**	0.951	0.98	0.992	**1.0**	1.0
0.8	0.77	0.855	0.9	0.94	0.963	0.853	0.913	0.956	0.972	0.986
0.9	0.588	0.694	0.685	0.718	**0.771**	0.772	0.805	0.827	0.846	0.873
1	0.434	0.445	0.424	0.422	**0.397**	0.576	0.56	0.586	0.598	**0.584**
	$Borda_1^{max}$					$Borda_2^{max}$				
0.7	0.902	0.958	0.986	0.992	**1.0**	0.947	0.987	0.992	**1.0**	1.0
0.8	0.77	0.855	0.9	0.94	0.963	0.861	0.938	0.945	0.962	0.971
0.9	0.588	0.694	0.685	0.718	**0.771**	0.704	0.794	0.802	0.814	0.845
1	0.434	0.445	0.424	0.422	**0.397**	0.567	0.581	0.555	0.56	**0.532**
	$Harmonic_1$					$Harmonic_2$				
0.7	0.941	0.986	0.996	1.0	**1.0**	0.98	0.992	1.0	**1.0**	1.0
0.8	0.895	0.916	0.958	0.959	0.968	0.958	0.974	0.987	0.988	0.996
0.9	0.805	0.808	0.83	0.866	**0.863**	0.895	0.921	0.934	0.939	0.952
1	0.725	0.742	0.74	0.697	**0.737**	0.872	0.867	0.859	0.861	**0.859**
	$Copeland_1$					$Copeland_2$				
0.7	0.908	0.968	0.991	0.994	**1.0**	0.947	0.99	1.0	**1.0**	1.0
0.8	0.736	0.847	0.891	0.934	0.949	0.822	0.904	0.952	0.984	0.982
0.9	0.497	0.567	0.655	0.684	**0.726**	0.62	0.69	0.77	0.805	0.838
1	0.325	0.332	0.323	0.343	**0.319**	0.458	0.432	0.45	0.442	**0.425**
	$Maximin_1$					$Maximin_2$				
0.7	0.908	0.969	0.986	0.99	**1.0**	0.968	0.991	1.0	**1.0**	1.0
0.8	0.787	0.856	0.915	0.939	0.955	0.872	0.934	0.961	0.976	0.977
0.9	0.57	0.633	0.691	0.717	**0.748**	0.735	0.76	0.794	0.838	0.869
1	0.415	0.4	0.423	0.393	**0.391**	0.52	0.532	0.544	0.545	**0.525**
	RP_1					RP_2				
0.7	0.926	0.972	0.995	0.995	**1.0**	0.963	0.994	1.0	**1.0**	1.0
0.8	0.778	0.856	0.908	0.939	0.957	0.871	0.928	0.967	0.983	0.989
0.9	0.587	0.64	0.674	0.718	**0.749**	0.725	0.765	0.777	0.838	0.862
1	0.426	0.405	0.416	0.375	**0.385**	0.558	0.524	0.557	0.498	**0.519**
	STV_1					STV_2				
0.7	0.907	0.981	0.985	0.998	**1.0**	0.959	0.993	0.997	**1.0**	1.0
0.8	0.808	0.865	0.917	0.918	0.943	0.882	0.933	0.962	0.966	0.974
0.9	0.603	0.64	0.721	0.729	0.763	0.742	0.776	0.792	0.855	0.846
1	0.45	0.464	0.477	0.471	0.468	0.576	0.593	0.61	0.592	0.585

With high values of ϕ, Harmonic still outperforms other rules followed by $Borda^{av}$ and STV then the other rules. Results for Borda rule suggest that $Borda_k^{av}$ has the best results followed by $Borda_k^0$ and then $Borda_k^{max}$, especially with high values of ϕ. In all cases, *top*-2 ballots seem to be always sufficient in practice to predict the winner with 100% accuracy with a low value of ϕ.

In the second set of experiments, we are interested in determining the value of k needed to predict the correct winner with large elections and with a high value of ϕ. We take $k = \{1, ..., m\}$, $n = 2000$, $\phi = \{0.9, 1\}$ and $m = 20$. (Here we consider the

same setting used by Filmus and Oren [37] in their experiments when $\phi = 1$.) Note that when $\phi \leqslant 0.8$, our results show that $k = 1$ always suffices for 100% accuracy for all truncated rules. Figure 3.4 shows depicted results where 1000 random preference profiles are generated for each experiment.

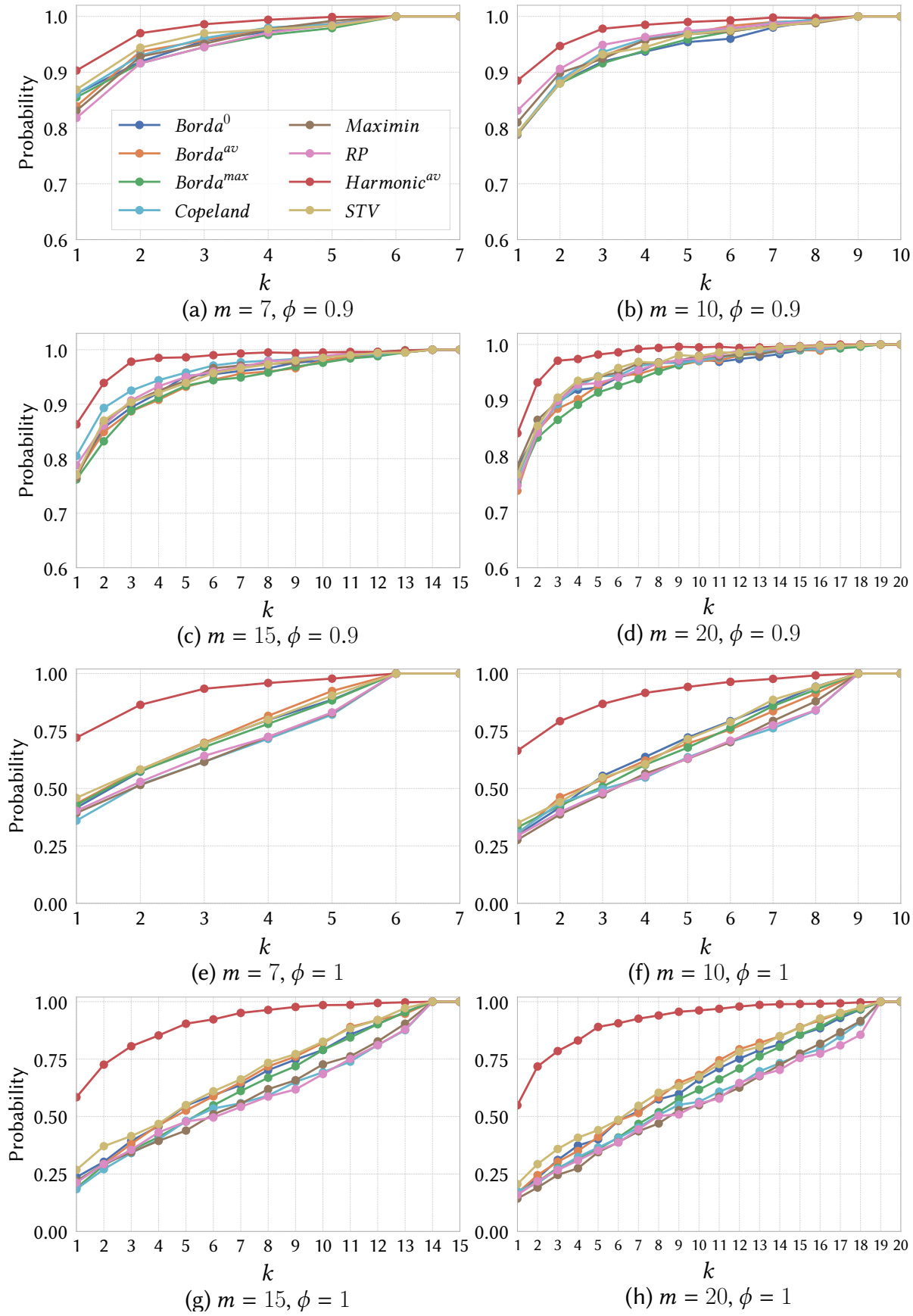


(a) $\phi = 0.9$, $m = 20$ (b) $\phi = 1$, $m = 20$

Figure 3.4: Success rate, $n = 2000$, $m = 20$: Mallows ϕ model when $\phi \in \{0.9, 1\}$ and varying k.

Results suggest that in large elections and unless ϕ is very high ($\phi = 0.9$), *top-k* rules succeed to identify the true winner when $k = 8$ (resp. $k = 14$) for Harmonic (resp. the remaining rules) out of $m = 20$. For $\phi = 1$, all voters' preferences are needed to determine the winner except for Harmonic that requires only $k = 15$ voters' preferences over 20 available. We can also observe the behavior of different truncated rules when $\phi = 0.9$: the best accuracy is obtained again by Harmonic and the accuracy of all other rules are very close, which we found surprising. When $\phi = 1$, the latter behavior changes: Harmonic still has the best results, followed by $Borda^{av}$ and STV, then the remaining rules. The good performance of Harmonic in all cases can be explained by the fact that the closer the scoring vector to plurality, the better the prediction.

In the next experiments, in the aim to determine the value of k (as a function of m) for which the k-truncated rules maximizes the probability of coinciding with the standard ones, we vary the number of candidates $m \in \{7, 10, 15, 20\}$ and $\phi \in \{0.9, 1\}$ when $n = 1000$. Results are depicted in Figure 3.5 where 1000 random preference profiles are generated for each experiment. We also comment on results when $n = 2000$.



(a) $m = 7, \phi = 0.9$ (b) $m = 10, \phi = 0.9$

(c) $m = 15, \phi = 0.9$ (d) $m = 20, \phi = 0.9$

(e) $m = 7, \phi = 1$ (f) $m = 10, \phi = 1$

(g) $m = 15, \phi = 1$ (h) $m = 20, \phi = 1$

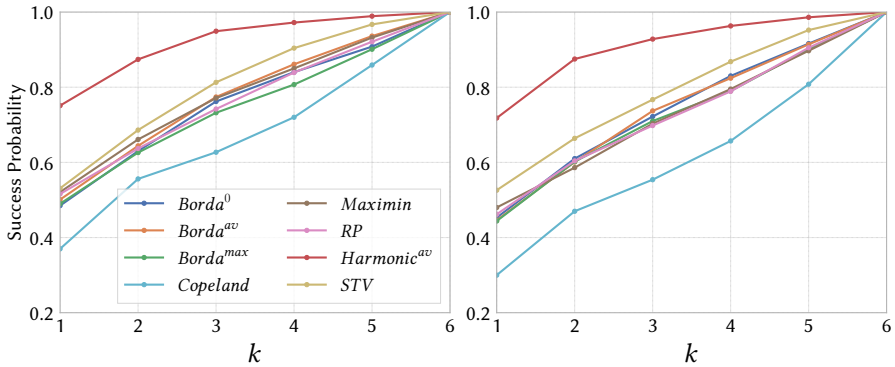
Figure 3.5: Success rate, $m \in \{7, 10, 15, 20\}$, $n = 1000$: Mallows ϕ model when $\phi \in \{0.9, 1\}$ and varying k.

The results for $Borda^{av}$ are:

- for $\phi = 0.7$, $k = 1$ is always sufficient, whatever m.
- for $\phi = 0.8$, $k = 2$ (resp. $k = 1$) is always sufficient for $n = 1000$ (resp. $n = 2000$), whatever the value of m.
- for $\phi = 0.9$, we observe that the minimal value of k such that the correct winner is always correctly predicted is around $\frac{4}{5}m$ (for $n = 1000$) and $\frac{7}{10}m$ (for $n = 2000$).
- for $\phi = 1$, the minimal value of k is $m - 1$: we always find a generated profile for which we get an incorrect result if the profile is not complete.

The results for Copeland, Maximin, RP and STV are similar to those for Borda. For Harmonic, we observe that $k = 1$ is always sufficient for $\phi \leqslant 0.8$ and $n = 2000$, and that for $\phi = 0.9$ (resp. $\phi = 1$), the value of k needed is around $\frac{2}{5}m$ (resp. $\frac{3}{4}m$).

In order to see how our approximations behave with a small number of voters and a high dispersion parameter, we take $k = \{1, ..., m - 1\}$, $n = 15$, $m = 7$, and $\phi \in \{0.9, 1\}$. The results are on Figure 3.6.



(a) $\phi = 0.9$ (b) $\phi = 1$ (*IC*)

Figure 3.6: Success rate, $m = 7$, $n = 15$: Mallows ϕ model when $\phi \in \{0.9, 1\}$ and $k = \{1, ..., 6\}$.

The best performance is obtained by Harmonic: even with few voters, winner prediction is almost perfect when $k = 4$ over 7 candidates. With a small number of voters, STV outperforms Borda and pairwise rules where the success rate reaches 98% when $k = 5$. The worst performance is obtained with Copeland, while the other rules perform more or less equally well. These results are consistent with the results obtained

by Skowron *et al.* [67] for multiwinner rules: elections with few voters and high dispersion appear to be the worst-case scenario for predicting the correct winner using top-truncated ballots.

3.3.2 Experiments Using Mixture of Mallows model

Now, we consider Mixture of Mallows (cf. Section 1.4) with p Mallows models when $p = 1$, $p = 2$ and $p = 3$ (Figure 3.7 (a), (b) and (c), respectively) and we measure the accuracy of the approximate rules. For each experiment we draw 1000 random preference profiles. We take $m = 7$, $n = 500$ and we vary $p \in \{1, 2, 3\}$ and $k \in \{1, \ldots, 7\}$.

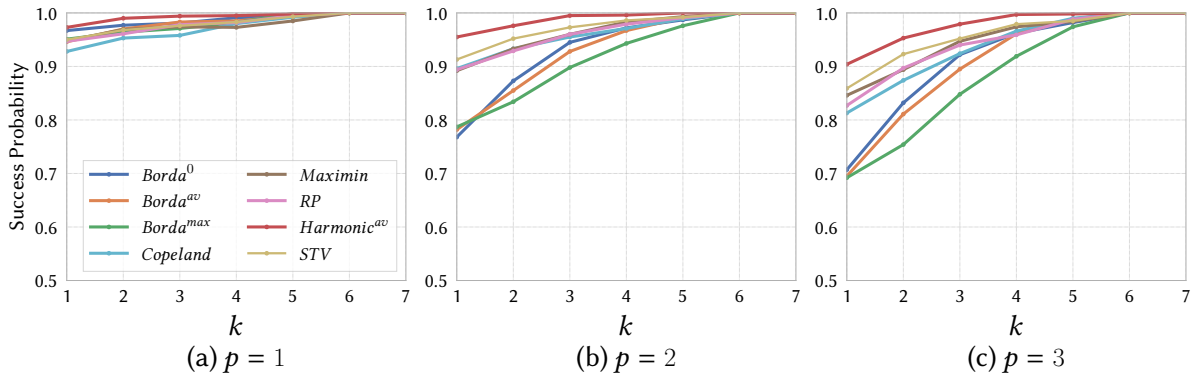


Figure 3.7: Success rate, $m = 7$, $n = 500$: Mixture of p Mallows models with $p = \{1, 2, 3\}$.

From the results, we observe that inducing more correlation between voters decreases the prediction of the correct winner for all rules. Indeed, increasing the number of models tend towards the impartial culture (all orders are increasingly equally likely), while a Mallows model with $p = 1$ will have a more tightly correlated set of votes, tending towards a Mallows model with a low ϕ. When $p = 1$, Harmonic rule gives the best performance, followed by Borda. With a mixture of two and three mallows models, the latter behavior changes where Harmonic still has the best prediction, followed by pairwise rules, then Borda. For $p = 2$ (resp. $p = 3$), *top*-3 (resp. *top*-4) ballots are sufficient to output the correct winner under Harmonic. For the remaining rules, the winner is predicted with 99% accuracy using *top*-5 ballots when $p = \{2, 3\}$.

3.3.3 Experiments Using Real Data Sets

We now consider real data set from *Preflib* [58]: 2002 election for Dublin North constituency with 12 candidates and 3662 voters. Dublin data contains some incomplete ballots. Since we need full ballots to run our experiments, we excluded all partial ballots.[2] We are interested in predicting the result for small and large elections. To do so, we consider data with samples of n^* voters among n ($n^* < n$), starting by $n^* = 10$ and increment n^* in steps of 10. In each experiment, 1000 random profiles are constructed with n^* voters; then we consider the *top-k* ballots obtained from these profiles, with $k = \{1, 2, 3\}$, and we compute the frequency with which we select the true winner. Figure 3.8 shows results for Dublin data with small elections ($n^* = \{10, ..., 100\}$).

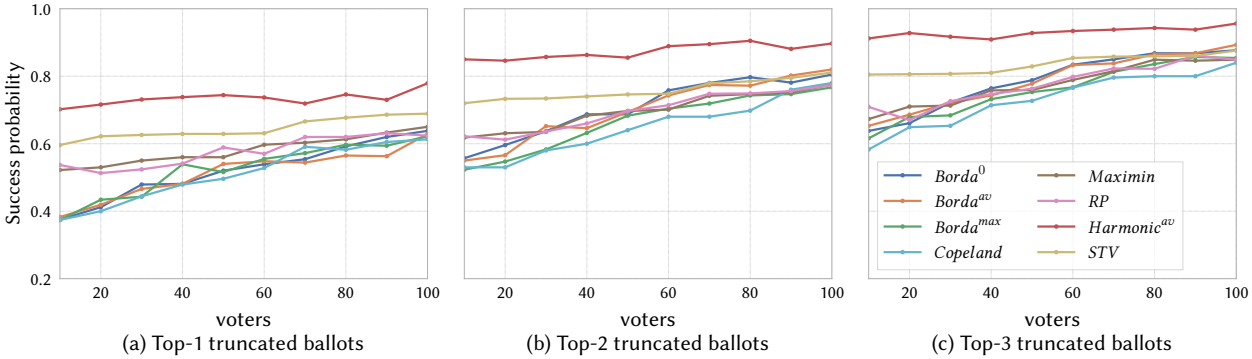


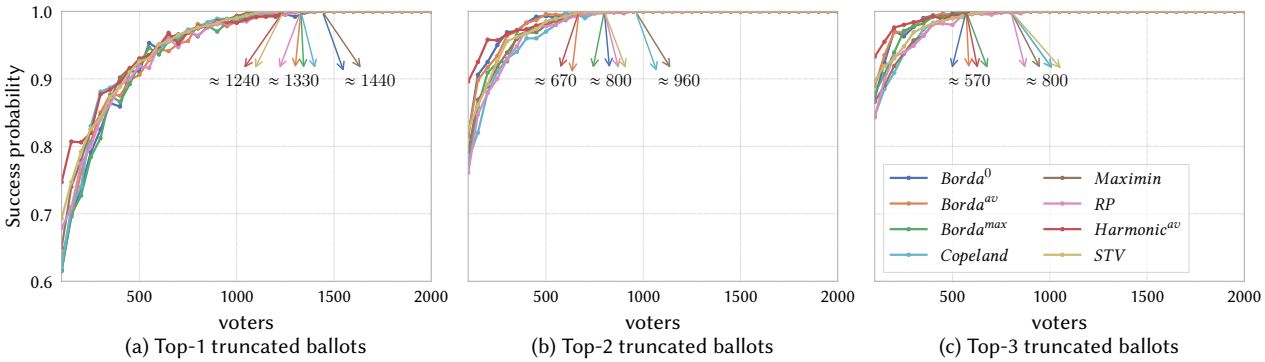
Figure 3.8: Success rate, Dublin North: varying $k \in \{1, 2, 3\}$ and $n^* = \{10, \ldots, 100\}$.

Our results suggest that predicting the correct winner for small elections fails rather often when k is too small ($k \leqslant \frac{1}{4}m$). We also see that performance increases with the number of voters. Consistently with the results of Figure 3.6, with a small number of voters, Harmonic gives the best performance followed by STV then the remaining rules e.g. For Harmonic (resp. STV), 92% (resp. 82%) accuracy is reached with $k = 3$ over 12 candidates and $n^* = 50$ against around 75% for the remaining rules.

The next experiments uses again Dublin data, but now we consider larger elections. We sample n^* voters, for $n^* \in \{100, \ldots, 2000\}$ with samples of n^* voters among n ($n^* < n$), starting by $n^* = 100$ and increment n^* in steps of 50. Similarly to the above experiments, 1000 random profiles are constructed with n^* voters; then we consider the *top-k* ballots obtained from the sampled profiles, with $k \in \{1, 2, 3\}$. Results are reported on Figure 3.9. Arrows indicate the number of voters from which the prediction is perfect.

[2] There are 43,942 ballots; 3662 are complete.

For large elections, performance increases with n^* and k. For $k = 1$, the different approximations exhibit almost the same behavior except Harmonic that performs better especially with few voters. Obviously, increasing the value of k leads to a decrease in the number of voters needed for correct winner selection.



(a) Top-1 truncated ballots (b) Top-2 truncated ballots (c) Top-3 truncated ballots

Figure 3.9: Success rate, Dublin North: varying $k \in \{1, 2, 3\}$ and $n^* = \{100, \ldots, 2000\}$.

Indeed, eliciting only $k = \frac{1}{4}m$ candidates over 12 for each voter is sufficient to predict the correct winner when $n^* \geqslant 570$ (resp. $n^* \geqslant 800$) for Borda and Harmonic (for the remaining rules). In general, the different approximations needs a sufficient number of voters to converge to the correct prediction. Scoring rules tend to require less voters.

3.4 Measuring the Approximation Ratio

In the previous section, we measured empirically the quality of the approximation by the frequency with which it outputs the true winner of the original rule. Now, we take a different path where we propose to evaluate the quality of the approximations by computing the score ratio between the score of the true winner and of the winner of the approximate rule. In Section 3.4.1 we analyze theoretically the score distortion in the worst case. We complete our study by an empirical analysis of the average-case study and a study from real data sets in Section 3.4.2 and Section 3.4.3, respectively.

3.4.1 Worst Case Study

In order to measure the quality of approximate voting rules whose definition is based on score maximization, a classical method consists in computing the worst-case approximation ratio between the scores (for the original rule) of the 'true' winner and of

the winner of the approximate rule. Using worst-case score ratios is classical: they are defined for measuring the quality of approximate voting rules [18, 66], for defining the price of anarchy of a voting rule [14] or for measuring the distortion of a voting rule [12].

Worst-case score ratios particularly make sense if the score of a candidate is meaningful beyond its use for determining the winner. This is definitely the case for Borda, as the Borda count is often seen as a measure of social welfare (see [28]). This worst-case score ratio is called the *price of top-k truncation*. Note that as STV and RP are not based on score maximization (for a discussion, see [23]) they will not be considered in our study.

Definition 3.5. *Let f be a voting rule defined by the maximization of a score S, and f_k a top-k approximation of f. The price of top-k-truncation for f, f_k, m, and k, is defined as*

$$R(f, f_k, m, k) = \max_{P \in \mathcal{P}_m} \frac{S(f(P))}{S(f_k(P_k))} \tag{3.1}$$

Example 3.6. *Let P be a complete profile of 9 voters having preferences over three candidates $A = \{a, b, c, d\}$ as follows:*

$$\begin{aligned} &4 \text{ votes } a \succ c \succ b \succ d \\ &3 \text{ votes } b \succ a \succ d \succ c \\ &2 \text{ votes } b \succ a \succ d \succ c \end{aligned}$$

When considering the complete profile, the score of different candidates is: S(a)=22, S(b)=19 and S(c)=8 and S(d)=5. The Borda winner is candidate a.

For $k = 1$, under $Borda_1^0$ candidate b is the winner. Then, $R(Borda, Borda_1^0, 4, 1) = \frac{22}{19}$.

For $k = 2$, under $Borda_2^0$ candidate a is the winner. Then, $R(Borda, Borda_2^0, 4, 2) = \frac{22}{22}$*=1.*

We devote Subsection 3.4.1.1 to study the price of *top-k*-truncation for positional scoring rules. Subsections 3.4.1.2 and 3.4.1.3 give the price of *top-k*-truncation for Copeland and Maximin, respectively.

3.4.1.1 Positional Scoring Rules

Let f^s be a positional scoring rule defined with scoring vector s. Assume the tie-breaking priority favors x_1. Let $f_k^{\bar{s}}$ be a *top-k* approximation of f^s, associated with vector $\bar{s} = (s_1, \ldots, s_k, s^*)$, with the same tie-breaking priority. Let $s' = (s_1 - s^*, \ldots, s_k - s^*, 0) = (s'_1, \ldots, s'_k, 0)$, i.e. $s'_i = s_i - s^*$ for $i = 1, \ldots, k$. Obviously, $f_k^{\bar{s}} = f_k^{s'}$. For instance, if $f^{\bar{s}}$ is the average-score approximation of the Borda rule, then $\bar{s} = (m-1, \ldots, m-k, \frac{m-k-1}{2})$ and $s' = (m-1-\frac{m-k-1}{2}, \ldots, m-k-\frac{m-k-1}{2}, 0)$.

Let $S(x, P)$ be the score of x for P under f^s and $S'_k(x, P_k)$ be the score of x for P_k under $f_k^{s'}$. From now on when we write scores we omit P and P_k, i.e. we write $S(x)$ instead of $S(x, P)$, $S'_k(x)$ instead of $S'_k(x, P_k)$ etc. In the rest of Subsection 3.4.1 we assume $k \geqslant 2$.

Let $x_1 = f_k^{s'}(P_k)$ and $x_2 = f^s(P)$.

Upper bound: We start by computing an upper bound of the worst-case score ratio.

Lemma 3.1.

$$R(f^s, f_k^{s'}, m, k) \leqslant 1 - \frac{s_{k+1}}{s'_1} + \left(1 + \frac{s^*}{s'_1}\right) \frac{m s_{k+1}}{s'_1 + \ldots + s'_k}$$

Proof. The total number of points given to candidates under $f_k^{s'}$ is $n(s'_1 + \ldots + s'_k)$, therefore $S'_k(x_1) \geqslant \frac{n}{m}(s'_1 + \ldots + s'_k)$.

Let us write $S(x_2) = S_{1\to k}(x_2) + S_{k+1\to m}(x_2)$, where $S_{1\to k}(x_2)$ (resp. $S_{k+1\to m}(x_2)$) is the number of points that x_2 gets from the *top-k* (resp. bottom $m-k$) positions of the ballots in P. Let γ be the number of ballots in which x_2 is not in the *top-k* positions. Then $S_{k+1\to m}(x_2) \leqslant \gamma s_{k+1}$.

As x_2 appears in at least $\frac{S'_k(x_2)}{s'_1}$ *top-k* ballots, we have $\gamma \leqslant n - \frac{S'_k(x_2)}{s'_1}$. Moreover we have $S(x_1) \geqslant S_{1\to k}(x_1) = S'_k(x_1) + ns^* \geqslant S'_k(x_2) + ns^* = S_{1\to k}(x_2)$. Now,

$$\begin{aligned}
S(x_2) &\leqslant S_{1\to k}(x_2) + \left(n - \tfrac{S'_k(x_2)}{s'_1}\right) s_{k+1} \\
&\leqslant S_{1\to k}(x_2) + \left(n - \tfrac{S_k(x_2) - ns^*}{s'_1}\right) s_{k+1} \\
&\leqslant (1 - \tfrac{s_{k+1}}{s'_1}) S_{1\to k}(x_2) + n s_{k+1} + \tfrac{ns^* s_{k+1}}{s'_1} \\
&\leqslant (1 - \tfrac{s_{k+1}}{s'_1}) S(x_1) + n s_{k+1} + \tfrac{ns^* s_{k+1}}{s'_1}
\end{aligned}$$

$$\begin{aligned}
\tfrac{S(x_2)}{S(x_1)} &\leqslant 1 - \tfrac{s_{k+1}}{s'_1} + n s_{k+1}(1 + \tfrac{s^*}{s'_1}) \tfrac{m}{n(s'_1 + \ldots + s'_k)} \\
&\leqslant 1 - \tfrac{s_{k+1}}{s'_1} + s_{k+1}(1 + \tfrac{s^*}{s'_1}) \tfrac{m}{s'_1 + \ldots + s'_k} \qquad \square
\end{aligned}$$

Lower bound: We now focus on the lower bound. We build the following pathological complete profile P such that:

- the winner for P_k (resp. P) is x_1 (resp. x_2).
- in P_k, all candidates get the same number of points (x_1 wins thanks to tie-breaking), and x_1 and x_2 get all their points from *top*-1 positions.
- in P, the score of x_1 is minimized by ranking it last everywhere where it was not in the *top*-k positions, and the score of x_2 is maximized by ranking it in position $k+1$ everywhere where it was not in the *top*-k positions.
- P_k is symmetric in $\{x_3, \ldots, x_m\}$.

Formally, P_k is defined as follows:

1) for each ranked list L (resp. L') of $k-1$ (resp. k) candidates in $\{x_3, \ldots, x_m\}$: α votes x_1L and α votes x_2L (resp. β votes L'). α and β will be fixed later.
2) α and β are chosen in such a way that all candidates get the same score $S'_k(.)$.
3) the tie-breaking priority relation favors x_1.

Now, P is obtained by completing P_k as follows:

1) each *top*-k vote x_1L is completed into x_1Lx_2-. "$-$" means the remaining candidates are in an arbitrary order.
2) each *top*-k vote x_2L is completed into $x_2L - x_1$.
3) each *top*-k vote L' is completed into $L'x_2 - x_1$.

Example 3.7. *For $m = 5$ and $k = 3$, we obtain the profile presented in Table 3.2.*

Lemma 3.2.

$$S'_k(x_1) = S'_k(x_2) = \alpha(m-2)s'_1M$$

and for $i \geqslant 3$, $S'_k(x_i) = 2\alpha(s'_2 + \ldots + s'_k)M + \beta(m-k-1)(s'_1 + \ldots + s'_k)M$

where $M = \frac{(m-3)!}{(m-k-1)!}$

Table 3.2: Profile P when $m = 5$ and $k = 3$

P_3		P	
α votes:	$x_1x_3x_4$	α votes:	$x_1x_3x_4x_2x_5$
α votes:	$x_1x_3x_5$	α votes:	$x_1x_3x_5x_2x_4$
α votes:	$x_1x_4x_3$	α votes:	$x_1x_4x_3x_2x_5$
α votes:	$x_1x_4x_5$	α votes:	$x_1x_4x_5x_2x_3$
α votes:	$x_1x_5x_3$	α votes:	$x_1x_5x_3x_2x_4$
α votes:	$x_1x_5x_4$	α votes:	$x_1x_5x_4x_2x_3$
α votes:	$x_2x_3x_4$	α votes:	$x_2x_3x_4x_5x_1$
α votes:	$x_2x_3x_5$	α votes:	$x_2x_3x_5x_4x_1$
α votes:	$x_2x_4x_3$	α votes:	$x_2x_4x_3x_5x_1$
α votes:	$x_2x_4x_5$	α votes:	$x_2x_4x_5x_3x_1$
α votes:	$x_2x_5x_3$	α votes:	$x_2x_5x_3x_4x_1$
α votes:	$x_2x_5x_4$	α votes:	$x_2x_5x_4x_3x_1$
β votes:	$x_3x_4x_5$	β votes:	$x_3x_4x_5x_2x_1$
β votes:	$x_3x_5x_4$	β votes:	$x_3x_5x_4x_2x_1$
β votes:	$x_4x_3x_5$	β votes:	$x_4x_3x_5x_2x_1$
β votes:	$x_4x_5x_3$	β votes:	$x_4x_5x_3x_2x_1$
β votes:	$x_5x_3x_4$	β votes:	$x_5x_3x_4x_2x_1$
β votes:	$x_5x_4x_3$	β votes:	$x_5x_4x_3x_2x_1$

Proof. In P_k, x_1 and x_2 appear in top position in a number of votes equal to α times the number of different permutations (ordered lists) of $(k-1)$ candidates out of $(m-2)$, i.e. $\alpha\frac{(m-2)!}{(m-k-1)!}$ times. Thus $S'_k(x_1) = S'_k(x_2) = \alpha\frac{(m-2)!}{(m-k-1)!}s'_1$.

To compute $S'(x_i)$; for each $i \geqslant 3$, x_i appears in $(2, \ldots, k)$ position in a number of votes equal to 2α times the number of different permutations (ordered lists) of $(k-2)$ candidates out of $(m-3)$ i.e. $2\alpha\frac{(m-3)!}{(m-k-1)!}$.

Also, x_i appears in $(1, \ldots, k)$ position in a number of votes equal to β times the number of different permutations (ordered lists) of $(k-1)$ candidates out of $(m-3)$ i.e. $\beta\frac{(m-3)!}{(m-k-2)!}$.

Thus,

$$\begin{aligned} S'_k(x_i) &= 2\alpha\tfrac{(m-3)!}{(m-k-1)!}s'_2 + \cdots + 2\alpha\tfrac{(m-3)!}{(m-k-1)!}s'_k \\ &+ \beta\tfrac{(m-3)!}{(m-k-2)!}s'_1 + \cdots + \beta\tfrac{(m-3)!}{(m-k-2)!}s'_k. \end{aligned}$$

Then,

$$S'_k(x_i) \quad = \quad 2\alpha\tfrac{(m-3)!}{(m-k-1)!}(s'_2 + \cdots + s'_k) + \beta\tfrac{(m-3)!}{(m-k-2)!}(s'_1 + \cdots + s'_k). \quad \square$$

Lemma 3.3. *All candidates have the same score in P_k if and only if*

$$\frac{\beta}{\alpha} = \frac{(m-2)s'_1 - 2(s'_2 + \ldots + s'_k)}{(m-k-1)(s'_1 + \ldots + s'_k)}$$

Proof. Using Lemma 3.2 and in order to ensure that all candidates have the same score in P_k, we must have:

$$\begin{aligned} & S'_k(x_1) = S'_k(x_i) \\ \Rightarrow\ & \alpha(m-2)s'_1 M = 2\alpha(s'_2 + \ldots + s'_k)M + \beta(m-k-1)(s'_1 + \ldots + s'_k)M \\ \Rightarrow\ & \alpha((m-2)s'_1 - 2(s'_2 + \ldots + s'_k)) = \beta(m-k-1)(s'_1 + \ldots + s'_k) \end{aligned}$$

Then we get: $\alpha = (m-k-1)(s'_1 + \ldots + s'_k)$ and $\beta = (m-2)s'_1 - 2(s'_2 + \ldots + s'_k)$

This holds if and only if

$$\frac{\beta}{\alpha} = \frac{(m-2)s'_1 - 2(s'_2 + \ldots + s'_k)}{(m-k-1)(s'_1 + \ldots + s'_k)}$$

□

Thus, in P_k, thanks to the tie-breaking priority, the winner is x_1. Thanks to the tie-breaking priority, the winner in P_k is x_1. Now, in P, the winner is x_2 and the scores of x_1 and x_2 are given by the following Lemma:

Lemma 3.4.

$$\begin{aligned} S(x_1) &= Q\alpha s_1 \\ S(x_2) &= Q\alpha s_1 + Q\alpha s_{k+1} + Q(m-k-1)\beta s_{k+1} \end{aligned}$$

$$where \quad Q = \tfrac{(m-2)!}{(m-k-1)!}$$

Proof. x_1 appears at the top of $\frac{(m-2)!}{(m-k-1)!}\alpha$ votes and at the bottom of all others, hence $S(x_1) = Q\alpha s_1$.

x_2 appears $\alpha\frac{(m-2)!}{(m-k-1)!}$ times top position, and in position $(k+1)$ in the remaining votes, *i.e.*, $\alpha\frac{(m-2)!}{(m-k-1)!} + \beta\frac{(m-2)!}{(m-k-2)!}$ where:

- x_2 appears in top position in a number of votes equal to α times the number of different permutations of $(k-1)$ candidates out of $(m-2)$ i.e. $\alpha\frac{(m-2)!}{(m-k-1)!}$ with a score of s_1
- x_2 appears in position $(k+1)$ in a number of votes equal to α times the number of different permutations of $(k-1)$ candidates out of $(m-2)$ i.e. $\alpha\frac{(m-2)!}{(m-k-1)!}$ with a score of s_{k+1}
- x_2 appears in position $(k+1)$ in a number of votes equal to β times the number of different permutations of k candidates out of $(m-2)$ i.e. $\beta\frac{(m-2)!}{(m-k-2)!}$ with a score of s_{k+1}

Thus

$$\begin{aligned} S(x_2) &= \alpha\tfrac{(m-2)!}{(m-k-1)!}s_1 + \tfrac{(m-2)!}{(m-k-1)!}s_{k+1} + \beta\tfrac{(m-2)!}{(m-k-2)!}s_{k+1} \\ S(x_2) &= \alpha\tfrac{(m-2)!}{(m-k-1)!}(s_1+s_{k+1}) + \beta\tfrac{(m-2)!}{(m-k-2)!}s_{k+1} \quad \square \end{aligned}$$

Lemma 3.5.

$$R(f^s, f_k^{s'}, m, k) \geqslant 1 - \frac{s_{k+1}}{s_1} + \frac{s_{k+1}}{s_1}\frac{ms'_1}{s'_1+\ldots+s'_k}$$

Proof. From Lemma 3.4 we get $\frac{S(x_2)}{S(x_1)} \geqslant 1 + \frac{s_{k+1}}{s_1} + (m-k-1)\frac{s_{k+1}}{s_1}\frac{\beta}{\alpha}$.

Finally, using the expression of $\frac{\beta}{\alpha}$ in Lemma 3.3 we get

$$\frac{S(x_2)}{S(x_1)} \geqslant 1 + \frac{s_{k+1}}{s_1} + (m-k-1)\frac{s_{k+1}}{s_1}\frac{(m-2)s'_1 - 2(s'_2+\ldots+s'_k)}{(m-k-1)(s'_1+\ldots+s'_k)}$$

From this we conclude:

$$\begin{aligned} R(f^s, f_k^{s'}, m, k) &\geqslant 1 + \tfrac{s_{k+1}}{s_1} + \tfrac{s_{k+1}}{s_1}\tfrac{(m-2)s'_1 - 2(s'_2+\ldots+s'_k)}{s'_1+\ldots+s'_k} \\ &\geqslant 1 + \tfrac{s_{k+1}}{s_1} + \tfrac{s_{k+1}}{s_1}\tfrac{(m-2)s'_1 + 2s'_1 - 2(s'_1+\ldots+s'_k)}{s'_1+\ldots+s'_k} \\ &\geqslant 1 + \tfrac{s_{k+1}}{s_1} + \tfrac{s_{k+1}}{s_1}\left(\tfrac{ms'_1}{s'_1+\ldots+s'_k} - 2\right) \\ &\geqslant 1 - \tfrac{s_{k+1}}{s_1} + \tfrac{s_{k+1}}{s_1}\tfrac{ms'_1}{s'_1+\ldots+s'_k} \quad \square \end{aligned}$$

Proposition 3.1.

$$1 - \tfrac{s_{k+1}}{s_1} + \tfrac{s_{k+1}}{s_1}\tfrac{ms'_1}{s'_1+\ldots+s'_k} \leqslant R(f^s, f_k^{s'}, m, k) \leqslant 1 - \tfrac{s_{k+1}}{s'_1} + \left(1 + \tfrac{s^*}{s'_1}\right)\tfrac{ms_{k+1}}{s'_1+\ldots+s'_k}$$

Proof. Putting Lemmas 3.1 and 3.5 together. $\square$

As a corollary, when $s^* = 0$ the lower and upper bounds coincide giving a tight worst-case approximation ratio for this class of approximations.

Corollary 3.1.

$$When \quad s^* = 0, \quad R(f^s, f_k^{s'}, m, k) \quad = 1 - \tfrac{s_{k+1}}{s'_1} + \left(1 + \tfrac{s^*}{s'_1}\right) \tfrac{m s_{k+1}}{s'_1 + \ldots + s'_k}$$

Proof. When $s^* = 0$, we have $s'_1 = s_1$ then the lower bound is equal to:

$$1 - \frac{s_{k+1}}{s_1} + \frac{m s_{k+1}}{s'_1 + \ldots + s'_k}$$

The upper bound is equal to:

$$\begin{array}{lll} & 1 - \tfrac{s_{k+1}}{s'_1} + \left(1 + \tfrac{s^*}{s'_1}\right) \tfrac{m s_{k+1}}{s'_1 + \ldots + s'_k} & \\ = & 1 - \tfrac{s_{k+1}}{s'_1} + \tfrac{m s_{k+1}}{s'_1 + \ldots + s'_k} & \text{where }, s^* = 0 \\ = & 1 - \tfrac{s_{k+1}}{s_1} + \tfrac{m s_{k+1}}{s'_1 + \ldots + s'_k} & \text{where }, s'_1 = s_1 \end{array}$$

□

This is however not guaranteed when $s^* > 0$ (the reason being that the pathological profile used in the proof of Lemma 3.1 may not be the worst). In particular, for different approximate scoring rules, the lower and lower bounds are as follows:

1) Case of $Borda_k^0$:

   - $s_i = m - i, s^* = 0, s'_i = m - i$
   - $(s'_1, \ldots, s'_k) = (m-1, \ldots, m-k) = km - \frac{k(k+1)}{2}$
   - $s_{k+1} = m - k - 1$

   $R(Borda, Borda_k^0, m, k) = \frac{k}{m-1} + \frac{2m(m-k-1)}{k(2m-k-1)}$

2) Case of $Borda_k^{av}$:

   - $s_i = m - i, s^* = \frac{m-k-1}{2}, s'_i = m - i - \frac{m-k-1}{2}$
   - $(s'_1, \ldots, s'_k) = (\frac{m+k-1}{2}, \ldots, \frac{m-k+1}{2}) = \frac{mk}{2}$
   - $s_{k+1} = m - k - 1$

   $1 - \frac{m-k-1}{m-1} + \frac{(m-k-1)(m+k-1)}{k(m-1)} \leqslant R(Borda, Borda_k^{av}, m, k) \leqslant \frac{k(3k-m+1)+4(m-k-1)(m-1)}{k(m+k-1)}$

3) Case of $Borda_k^{max}$:

- $s_i = m - i, s^* = m - k - 1, s'_i = m - i - (m - k - 1) = k - i + 1$
- $(s'_1, \ldots, s'_k) = (k, \ldots, 1) = \frac{k(k+1)}{2}$
- $s_{k+1} = m - k - 1$

$$1 - \frac{m-k-1}{m-1} - \frac{2mk(m-k-1)}{k(m-1)(k+1)} \leqslant R(Borda, Borda_k^{max}, m, k) \leqslant 1 - \frac{m-k-1}{k} + (1 + \frac{m-k-1}{k})\frac{2m(m-k-1)}{k(k+1)}$$

4) Case of $Hamonic_k^0$:

- $s_i = \frac{1}{i}, s^* = 0, s'_i = \frac{1}{i}$
- $(s'_1, \ldots, s'_k) = (1, \ldots, \frac{1}{k}) = 1 + \frac{k-1}{2k^2}$
- $s_{k+1} = \frac{1}{k+1}$

$$R(Harmonic, Harmonic_k^0, m, k) = \frac{k}{k+1} + \frac{m}{(k+1)(1+\frac{1}{2}\cdots+\frac{1}{k})}$$

Table 3.3 summarizes the obtained upper and lower bounds for different approximate voting rules:

f_k	lower bound	upper bound
$Borda_k^0$	$\frac{k}{m-1} + \frac{2m(m-k-1)}{k(2m-k-1)}$	$\frac{k}{m-1} + \frac{2m(m-k-1)}{k(2m-k-1)}$
$Borda_k^{av}$	$1 - \frac{m-k-1}{m-1} + \frac{(m-k-1)(m+k-1)}{k(m-1)}$	$\frac{k(3k-m+1)+4(m-k-1)(m-1)}{k(m+k-1)}$
$Borda_k^{max}$	$1 - \frac{m-k-1}{m-1} - \frac{2mk(m-k-1)}{k(m-1)(k+1)}$	$1 - \frac{m-k-1}{k} + (1 + \frac{m-k-1}{k})\frac{2m(m-k-1)}{k(k+1)}$
$Harmonic_k^0$	$\frac{k}{k+1} + \frac{m}{(k+1)(\left(1+\frac{1}{2}\ldots+\frac{1}{k}\right)}$	$\frac{k}{k+1} + \frac{m}{(k+1)(\left(1+\frac{1}{2}\ldots+\frac{1}{k}\right)}$

Table 3.3: Summary of upper and lower bounds of the price of *top-k* truncation for $Borda_k^0$, $Borda_k^{av}$, $Borda_k^{max}$ and $Harmonic_k^0$.

Note that for k'-approval with $k' > k$ and $s^* = 0$, the (exact) worst-case ratio $\frac{m}{k}$ does not depend on k'.

As a corollary, we get the following order of magnitudes when m grows:

Corollary 3.2.

$$R(Borda, Borda_k^0, m, k) = \Theta\left(\frac{m}{k}\right) \quad , \quad R(Borda, Borda_k^{av}, m, k) = \Theta\left(\frac{m}{k}\right)$$
$$R(Borda, Borda_k^{max}, m, k) = \Theta\left(\frac{m}{k}\right) \quad , \quad R(Harmonic, Harmonic_k^0, m, k) = \Theta\left(\frac{m}{k \log k}\right)$$

3.4.1.2 Copeland

For the Copeland rule, the ratio makes less sense, because the Copeland score is less meaningful as a measure of social welfare. Moreover, there are several ways of defining the Copeland score (assign 0 point for every lost contest, 1 point for every won contest and α points (with possibly $\alpha \neq \frac{1}{2}$) for every draw), all leading to the same rule. However, this has no impact on the negative result below, as long as a Condorcet loser[3] has score 0. Still, for the sake of completeness we give the following result:

Proposition 3.2. $R(Copeland, Copeland_k, m, k) = \infty$.

Proof. Let P be the following profile:

- P_k contains two votes $x_1 x_2 \dots x_k$, and one vote L for each ordered list of k candidates among m.
- P is obtained by completing P_k by adding x_1 (resp. x_2) in last position (resp. in position $k+1$) when it is not in the *top-k* positions.

In P_k, the winner for $Copeland_k$ is x_1. In P, the Copeland winner is x_2. Now, with respect to P, the Copeland score of x_1 (resp. x_2) is 0 (resp. $m-1$), hence the result. □

3.4.1.3 Maximin

Let $Maximin$ be the Maximin rule with tie-breaking priority order $x_1 \rhd \dots \rhd m$, and $Maximin_k$ be the k-truncated version of the Maximin rule with the same tie-breaking priority order. Let $S_{Mm}(x_2, P)$ and $S_{Mm}(x_1, P_k)$ be the Maximin scores of x_2 and x_1 for P and P_k, respectively, with $S_{Mm}(x_2, P) = \min y \neq x_2 N_P(x_2, y)$ and similarly for P_k. Let P be a profile, and let $x_1 = Maximin_k(P_k)$ and $x_2 = Maximin(P)$. All candidates have the same Maximin score in P_k, therefore, by tie-breaking priority, $Maximin_k(P_k) = x_1$.

Upper bound

Lemma 3.6. $R(Maximin, Maximin_k, m, k) \leqslant m - k + 1$.

[3] A *Condorcet loser* is a candidate defeated by a majority in pairwise contests against any other alternatives.

Proof. Because $x_1 = Maximin_k(P_k)$, we must have $S_{Mm}(x_1, P_k) \geqslant 1$ (otherwise we would have $S_{Mm}(x_1, P_k) \geqslant 0$, meaning that x_1 does not belong to any *top-k* ballot, and in this case we cannot have $x_1 = Maximin_k(P_k)$). Now, $S_{Mm}(x_2, P) \leqslant S_{Mm}(x_2, P_k) + (m-k) \leqslant S_{Mm}(x_1, P_k) + (m-k)$, therefore,

$$\begin{array}{rcl} \frac{S_{Mm}(x_2,P)}{S_{Mm}(x_1,P)} & \leqslant & \frac{S_{Mm}(x_1,P_k)+(m-k)}{S_{Mm}(x_1,P_k)} \\ & \leqslant & m-k+1 \quad \square \end{array}$$

Lower bound: Now, for the lower bound, we consider the following cyclic profile Cyc:

Cyc				
x_1	x_2	$\dots$	$m-1$	m
x_2	x_3	$\dots$	m	x_1
x_3	x_4	$\dots$	x_1	x_2
$\dots$	$\dots$	$\dots$	$\dots$	
m	x_1	$\dots$	$m-2$	$m-1$

Now, let P be obtained from Cyc by the following operations for every vote in Cyc:

- if x_1 is not in the *top-k* positions in the vote, we move it to the last position (and move all candidates who were below x_1 one position upward)
- if x_2 is not in the *top-k* positions in the vote, we move it to the $(k+1)^{th}$ position (and move all candidates who were between position $k+1$ and 2's position one position downward).

Example 3.8. *For $m = 5$, $k = 2$, we get the profile P in Table 3.4.*

Table 3.4: Cyclic profile when $m = 5$ and $k = 2$

P_2		P				
x_1	x_2	x_1	x_2	x_3	x_4	x_5
x_2	x_3	x_2	x_3	x_4	x_5	x_1
x_3	x_4	x_3	x_4	x_2	x_5	x_1
x_4	x_5	x_4	x_5	x_2	x_3	x_1
x_5	x_1	x_5	x_1	x_2	x_3	x_4

Lemma 3.7. $R(Maximin, Maximin_k, m, k) \geqslant m - k$.

Proof. $Maximin(P) = x_2$, and the Maximin scores of x_1 and x_2 in P are:

$$S_{Mm}(x_1, P) = 1 \text{ and } S_{Mm}(x_2, P) = m - k.$$

Hence $\frac{S_{Mm}(x_2,P)}{S_{Mm}(x_1,P)} = m - k$. □

Proposition 3.3. $m - k \leqslant R(Maximin, Maximin_k, m, k) \leqslant m - k + 1$.

Proof. Putting Lemmas 3.6 and 3.7 together. □

This worst-case ratio is quite bad, except if k is close to m. However, arguably, the Maximin score makes less sense *per se* (i.e. as a measure of social welfare) than a positional score such as the Borda count.

3.4.1.4 Discussion

The obtained worst-case bounds are rather negative: very negative for Copeland and Maximin, less so for Borda, and even less so for Harmonic. However, the Maximin and Copeland scores make less sense as a measure of social welfare than positional scores.

For different approximations, the score ratio is very diverse. For positional scoring rules, the approximation ratio is reasonable, especially if k is not too small; for Maximin and Copeland, this is much less good, but scores for these two rules are also arguably less meaningful *per se*. For instance, if $k = \frac{m}{4}$ (which means that voters have to report one fourth of their rankings), the approximation ratios for Harmonic, Borda, Maximin and Copeland are respectively $\frac{4}{log(\frac{m}{4})}$, 4, $\frac{3m}{4}$ and ∞. Now, we may wonder whether these worst cases do occur frequently in practice or if they correspond to rare pathological profiles. The next two subsections show that the latter is the case.

We devote the next two sections to an experimental study, first considering randomly generated profiles and then real data from the *Preflib* library. We show that average-case results are much more positive than in the worst case.

3.4.2 Average Case Evaluation

This section presents the evaluation of the approximation ratio for different truncated voting rules using data generated from the Mallows model and Mixture of Mallows. For each experiment, we draw 10000 random profiles.

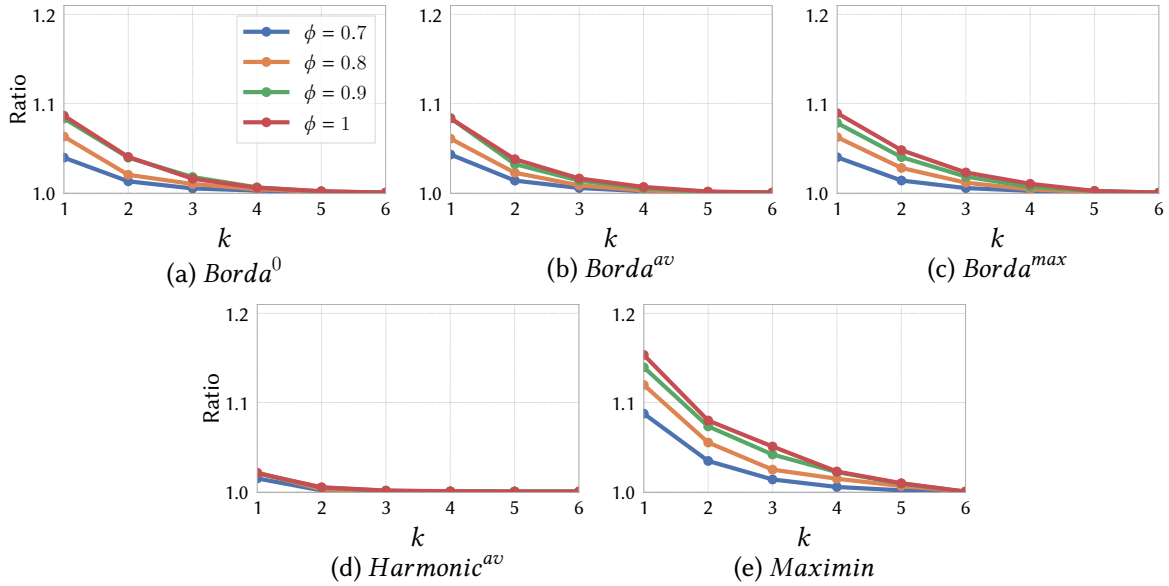


Figure 3.10: Approximation ratio when $n = 15$, $m = 7$ and $\phi \in \{.7, .8, .9, 1\}$: Mallows model.

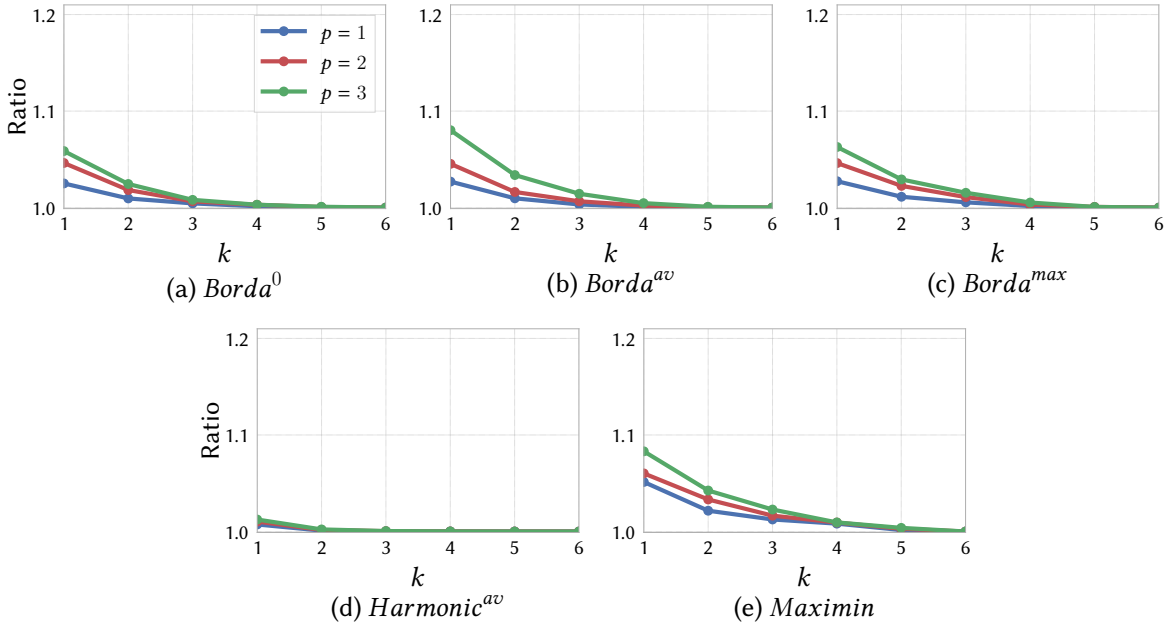


Figure 3.11: Approximation ratio when $n = 15$, $m = 7$ and $p = \{1, 2, 3\}$: Mixture of p Mallows models.

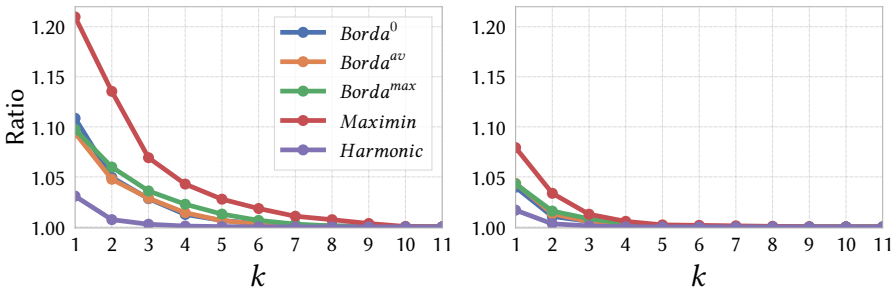
We present simulation results with $m = 7$ and $n = 15$. We simulate the elicitation of

top-k ($k \in \{1 \dots 6\}$) preferences with different values of $\phi = \{.7, .8, .9, 1\}$ and mixture of p Mallows models $p = \{1, 2, 3\}$. We present results of the approximation ratio for different truncated voting rules. Figure 3.10 shows obtained results with different values of ϕ. Figure 3.11 reports on results with mixture of p Mallows when $p = \{1, 2, 3\}$.

Results suggest that the value of the approximation ratio decreases when k increases. In practice, results are much better than in the worst case: best results are obtained by Harmonic, followed by Borda and finally Maximin.

3.4.3 Real Data Sets

Again we consider 2002 Dublin North data ($m = 12, n = 3662$) with samples of n^* voters among n ($n^* < n$) where $n^* = \{15, 100\}$. In each experiment 1000 random profiles are constructed with n^* voters; then we consider the *top-k* ballots obtained from these profiles with $k = \{1, \dots, 11\}$. Again, the results are very positive especially with a large number of voters.



(a) $n = 15$ (b) $n = 100$

Figure 3.12: Approximation ratio with Dublin North data set for different truncated voting rules.

3.5 Conclusion

In this chapter we have considered "k-truncated approximations" of rules (Borda, Harmonic, Copeland, Maximin, ranked pairs and STV) which take only the *top-k* candidates of each ballot. Then, we have proposed two measures of the quality of the approximation: the probability of selecting the same winner as the original rule, and the score ratio.

For the first measure, we have measured empirically the quality of the k-truncated rules by the frequency with which they output the true winner. Empirical results demonstrate the practical viability and advantages of our approximations. Our results suggest that small values of k work very well in practice: (1) with real data, *top*-3 ballots over 12 candidates are good predictors of the correct winner with a sufficient number of voters, (2) with data generated from Mallows ϕ model and when $\phi \leqslant 0.8$ (resp. $\phi = 0.9$, $\phi = 1$), eliciting $k = 1$ ($k = \frac{7}{10}$, $k = m - 1$) is sufficient to output the true winner when $n = 2000$ voters. (These results are valid for all rules except Harmonic which needs only $k = 1$ (resp. $k = \frac{2}{5}$, $k = \frac{3}{4}$) when $\phi \leqslant 0.8$ (resp. $\phi = 0.9$, $\phi = 1$).) (3) with data generated from Mixture of Mallows, best prediction is obtained with Harmonic where *top*-3 (resp. *top*-4) out of 7 are sufficient to output the correct winner under Harmonic when $p = 2$ (resp. $p = 3$).

For the second measure, we have studied the theoretical bounds of these approximations, for rules whose definition is based on score maximization (Borda, Harmonic, Copeland and Max imin), by identifying the order of the worst-case ratio. Also, we have tested the ability of these truncated rules to predict the standard voting rules based on both randomly generated profiles and real data. While the theoretical bounds are, at best, moderately encouraging, our experiments show that in practice the approximation ratio is much better than in the worst case: our results on the evaluation of the accuracy on random Mallows model as well as real data suggest that a very small value of k work very well in practice which largely contrast the theoretical bounds.

Chapter 4

STV: Incomplete Knowledge and Communication Issues

Contents

In this chapter, we focus on Single Transferable Vote (STV) (cf. Chapter 1). We start by studying empirically to which (quantitative) extent clone-proofness, which is a key property of STV, is preserved when replacing complete ballots by truncated ballots. Also, we test the resistance of RP_k to cloning since, among the ordinal voting rules, apart from STV; RP also satisfies clone-proofness. Then, we study empirically the proximity

to plurality with runoff of the rules STV_k for varying k. In the second part, we study the computational complexity of the possible winner (PW) problem for top-k ballots under STV where we show that for $k = 1$, it can be solved in polynomial time, but is NP-complete when $k \geqslant 2$. While the problem in difficult in the worst case, we show that in practice there is a simple algorithm that is able to label almost all candidates as possible winners or necessary losers. In the third part, we study a possible way to minimize the amount of communication required to use single-winner STV. We consider interactive communication protocols for STV. Building on a protocol proposed by Conitzer and Sandholm (2005), we show how we can reduce the amount of communication required in practice. We then study empirically the average communication complexity of these protocols, based on randomly generated profiles, and on real-world election data.

4.1 Introduction

We focus on a specific, but particularly important, single-winner voting rule: *single transferable vote* (STV) which is of particular interest for voting. There are good reasons for that:

- First, STV is appealing, because of its (apparent) technical simplicity which makes it easy to understand, especially because its several-round definition is reminiscent of that of *plurality with runoff* (note that the two rules coincide for three alternatives).
- Second, STV is NP-hard to manipulate, even for one voter [8], although this worst-case difficulty does not really carry on to the average case [71, 50].
- Third, STV enjoys a very important normative property: *clone-proofness* introduced by Tideman [68] which requires a voting rule to be robust to the introduction of similar candidates.[1] Freeman *et al.* [38] gave an axiomatic characterization of STV as a social welfare function. They show that STV is the only rule in a family of iterative elimination rules that satisfies clone-proofness. On the other hand, STV fails to satisfy a number of other important properties, such as monotonicity, participation, or Condorcet-consistency[2], but in many contexts, being

[1] Most voting rules with ordinal input fail to be clone-proof. Apart from STV, other noticeable exceptions are *ranked pairs* and *Schulze*. Among rules where the input does not consist of rankings over candidates, we find other clone-proof rules, especially *approval voting* and *range voting*.

[2] Marquis de Condorcet introduced the *Condorcet winner* as a candidate who beats any other candidate by a majority of votes. Then, a voting rule is Condorcet-consistent if it outputs the Condorcet winner whenever there is one.

sensitive to cloning balances the failure of these other properties. To give a simple example: at the 2002 French presidential, it has been argued that sensitivity to cloning was the cause of the socialist candidate Lionel Jospin not to go to the second round (which he would probably have won). Sensitivity to cloning is of primary importance not only for political elections, but also for low-stake collective decision making [65, 38].

Also, out of all the common voting rules of the zoo, STV is among the few ones that is used in political elections, both for single-winner voting and multi-winner voting. Indeed, the world's largest elections that allow voters to *rank* candidates (and not just select the top choice) are probably the elections for the Australian House of Representatives. Each of 150 districts sends a representative, who is chosen by single transferable vote. Voters are asked to rank/order the candidates in their district. (Partial rankings are not allowed.) While it would be easy to compute the winner from electronic ballots, counting the paper ballots is a major undertaking; in the 2016 elections, it took more than a week. Since this makes for bad TV, the "two-party-preferred" heuristic is used to quickly estimate the winner: officials guess which two candidates are most likely to win, and the majority margin between these is counted and reported.

STV_k (cf. Chapter 3) presents a better way to handle STV elections, that can find the election winner quickly enough for TV, and that do not require voters to give full rankings. In many legislatures, such as for the Irish presidential elections, voters are allowed to only submit a *partial* order by ranking a subset of the candidates, and leave the rest unranked; If all ranked candidates are eliminated by STV, the vote is then 'exhausted' and ignored during further counting. This variant of STV with partial rankings corresponds to STV_k with a fixed k. The freedom to give partial rankings is popular with voters. In the 2002 elections in Dublin, for which full ballot data is available, most voters chose to rank between 3 and 5 of the 12 candidates, with only 8% of voters submitting a full ranking (see Figure 4.1).

Although STV is clone proof, unfortunately this does not carry on to STV_k for $k \leqslant m-2$: For $k = 1$, STV_1 coincides with plurality which is highly sensitive to cloning while STV_{m-1} (=STV) is clone proof. What happens in between? The goal here is to check to which (quantitative) extent clone-proofness, which is a key property of STV, is preserved when replacing complete ballots by truncated ballots. Where there are few other known clone-proof rules; a noticeable exception is ranked pairs since among the ordinal rules seen in Chapter 1, ranked pairs also satisfies clone-proofness. Therefore, it is an interesting question for which one of STV and RP clone-proofness resists more to truncation. We answer this question empirically and we show that RP_k and STV_k resist to cloning almost similarly.

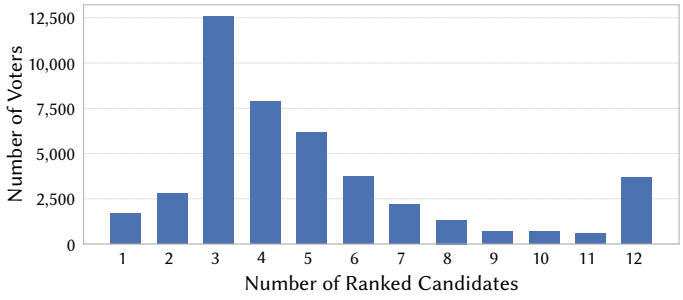


Figure 4.1: In the Dublin election, most of the 43,942 voters rank only 3 to 5 candidates out of 12.

Also, we study empirically the connection of STV_k to the plurality with runoff rule. Indeed, plurality and STV are at the two extremities of the families of rules STV_k. An intriguing question is the proximity of plurality with runoff to rules of this family. Plurality with runoff is intuitively related to plurality (since its selection of finalists is based on plurality scores) and to STV (since it proceeds by elimination of candidates based on plurality scores, and both rules coincide for $m = 3$). It is thus natural to wonder to study the proximity to plurality with runoff of the rules STV_k for varying k. We answer this question empirically and we show that STV is closer to plurality with runoff than plurality.

As specified, STV_k ignores exhausted votes, which implicitly assumes that we act as if the voters are indifferent between all unranked candidates. Given these *top-k* ballots, we would want to compute all *possible winners* (cf. Chapter 2) of STV. A problem with this proposal is that it is not clear how to efficiently decide which candidates are possible winners. It is known that determining PW for STV with truncated votes is NP-complete in the general case; however, with "uniform truncated ballots" (fixed k) the complexity of the PW problem has not been studied, except of course for $k = m-1$ (since $STV_{m-1} = STV$). In this chapter we study the computational complexity of the possible winner problem for *top-k* ballots. Indeed, we show that the problem is NP-complete, even for *top*-2 ballots. When given *top*-1 ballots (that is, we only know the plurality scores of the candidates), we give a simple characterization of the set of possible winners, which also yields a polynomial-time algorithm. While the problem is difficult for $k \geqslant 2$, we show that there is a simple algorithm that is able to label almost all candidates as possible winners or necessary losers. Indeed, we propose an approximation of the problem: given k-truncated ballots and a candidate c, we propose an algorithm, namely $PW_{top\text{-}k}$, that returns three possible results: (1) YES for which case c is a PW, (2) NO for which case c is not a PW and (3) MAYBE for which case we don't know if c is a PW or not. Results show that $PW_{top\text{-}k}$ allows to label almost all candidates as YES or NO and to reduce the set of candidates that we cannot decide

on (MAYBE).

Suppose we are not satisfied to merely approximate the true STV winner. Another way to lower the workload for voters by allowing *interactive* communication protocols (cf. Chapter 2), that can ask voters for more information in an adaptive fashion. In 2005, Conitzer and Sandholm [25] studied such protocols for several common voting rules that take rankings as input. They found that, asymptotically and in the worst case, for many popular rules (such as Borda, Copeland, and ranked pairs) we cannot do better than to ask for the entire ranking outright. This can be annoying to voters (as it is to many Australians), and costly. In contrast, Conitzer and Sandholm [25] found a natural communication protocol for STV where the average voter only needs to name a *logarithmic* number of candidates. The Conitzer–Sandholm (CS) protocol begins by asking each voter for their top candidate. Based on this information, the candidate with the lowest plurality score is eliminated. The protocol then asks the supporters of this candidate for their next-most-preferred choice. The key insight is that only a small minority of voters can be supporters of the eliminated candidate (by choice of that candidate); indeed at most n/m supporters are possible, where n and m are the total number of voters and of remaining candidates, respectively. Repeating the elimination process, we see that in total we ask at most $\frac{n}{m} + \frac{n}{m-1} + \cdots + \frac{n}{2} \approx n \cdot \log m$ many questions during the elimination phase.[3]

Can this protocol be improved? We do not offer an alternative with a better worst-case guarantee, but we propose a protocol that needs less communication in practice, and that never requires more communication than the CS protocol. Our protocol first asks voters for their initial top choices. Then, we identify a set of candidates that can safely be eliminated simultaneously. This avoids querying the same voter many times in a row, if that voter likes many niche candidates. This basic idea (slightly adapted to handle tie-breaking issues) allows for lower communication cost, on both random profiles and on real election data. On data from Australian elections, we find that many candidates are necessary losers, and can therefore immediately be eliminated, giving a more rigorous alternative to the current two-party-preferred heuristic.

The remainder of this chapter is structured as follows: Section 4.2 addresses the empirical evaluation of the resistance of STV_k and RP_k to cloning. Section 4.3 studies empirically the connection of STV_k to plurality-with-runoff rule. Section 4.4 inves-

[3]The existence of this protocol explains why counting STV elections on paper is feasible in the first place. We can view a paper ballot as an agent who is costly to 'communicate' with. First sort the papers into m physcial stacks according to top-ranked candidate. The Conitzer–Sandholm analysis shows that, to find the election winner, we only need to touch each ballot $O(\log m)$ times on average while redistributing.

tigates the possible winner problem for STV given k-truncated ballots. Section 4.5 focuses on communication protocols.

We mention that part of this chapter includes results from our papers:

- "*Manel Ayadi, Nahla Ben Amor, Jérôme Lang and Dominik Peters, title: Single Transferable Vote: Incomplete Knowledge and Communication Issues. In Proceedings of AAMAS 2019*" [6].
- "*Manel Ayadi, Nahla Ben Amor and Jérôme Lang, title: The Communication Burden of Single Transferable Vote, in Practice. In Proceedings of SAGT 2018, short paper*" [4].
- "*Manel Ayadi, Nahla Ben Amor and Jérôme Lang, title: The Communication Burden of Single Transferable Vote, in Practice. In Proceedings of COMSOC 2018, long paper*" [5].

4.2 Resistance to Cloning of STV_k and RP_k

Tideman [68] introduced the notion of *independence of clones*, which requires a voting rule to be robust to the introduction of similar candidates. Notably, while most voting rules fail Tideman's condition, the parallel-universe version of STV and RP satisfy it, and the resolute version essentially satisfies it as well (see below). In this section, we study empirically to what extent STV_k and RP_k stay clone-proof.

Let us make the notion of clone-proofness formal. Given a candidate x, introduce a new set of candidates X' called *clones of* x. Let $A' = (A \setminus \{x\}) \cup X'$. A ranking $\succ'$ over A' is compatible with a ranking $\succ$ over A if all elements of X' are ranked contiguously in $\succ'$. A profile $P' = (\succ'_1, \ldots, \succ'_n)$ over A' is compatible with a profile $P = (\succ_1, \ldots, \succ_n)$ over A if for every i, $\succ'_i$ is compatible with $\succ_i$. A (possibly irresolute) voting rule is *clone-proof* if, given a profile P and a profile P' compatible with P, x is a winner in P if and only if one of the clones of x is a winner in P, and for any $y \neq x$, y is a winner in P if and only if it is a winner in P'.

Let f be either STV or RP. Then the resolute version of f with immediate tie-breaking is clone-proof, provided that tie-breaking is consistent with cloning: Let $P = (\succ_1, \ldots, \succ_n)$ be a profile over A, and $P' = (\succ'_1, \ldots, \succ'_n)$ a profile over $A' = A \setminus \{x\} \cup X'$ compatible with P. Take a tie-breaking relation $\rhd$ over A, and suppose $\rhd'$ over A' is

compatible with $\rhd$. In this case, if $f^{\rhd}(P) = x$ then $f^{\rhd'}(P') \in X'$, and if $f^{\rhd}(P) = y \neq x$ then $f^{\rhd'}(P') = y$.

In order to evaluate the resistance of f_k to cloning, we propose an empirical approach by cloning a candidate and measuring experimentally the probability that cloning significantly changes the outcome. We clone the winner and a candidate chosen at random from A. For doing so we repeatedly do the following:

1) generate a complete profile P (with n voters and m candidates), and then
2) for each $k \in \{1, \dots m\}$:
   - we construct a profile P' obtained from P by cloning the winner (note that there are $m+1$ candidates in P'), and
   - we compare $f_k(P)$ to $f_k(P')$.

These steps are iterated a sufficient number of times to obtain meaningful results.

Example 4.1. *Let P be a profile with $n = 3$ and $A = \{a, b, c\}$:*

4	$a \succ c \succ b$
3	$b \succ a \succ c$
2	$c \succ b \succ a$

Let $\rhd$: $a \succ b \succ c$. The STV_2 (= STV) winner is b and the RP_2 winner is a. Let us clone b to $\{b, b'\}$. A compatible profile P' is $\{4 : a \succ c \succ b' \succ b, 3 : b \succ b' \succ a \succ c, 2 : c \succ b' \succ b \succ a\}$. For $k = 2$, the k-truncated profile is $\{4 : a \succ c, 3 : b \succ b', 2 : c \succ b'\}$ $\rhd'$: $a \succ b \succ b' \succ c$. Then, candidate a is the winner for both STV_2 and RP_2.

4.2.1 Experiments Using Mallows ϕ Model

For each experiment we draw 1000 profiles. We present simulation results for small and large elections when $m = 5$ as we vary ϕ and n. We simulate the elicitation of *top-k* preferences where $k \in \{1, \dots, 5\}$ for $n \in \{30, 500\}$ with $\phi \in \{0.7, 0.8, 0.9, 1\}$. We say that cloning sensitivity does not occur if the winner after cloning is the same as before, or a clone of the winner before cloning. We give the probability that clone sensitivity does not occur under STV_k and RP_k, when we clone: (i) one candidate chosen randomly (Figure 4.2) and (ii) the winner under STV_k or RP_k (Figure 4.3).

Resistance to cloning increases rapidly with k and decreases with ϕ. Also, it significantly increases with the number of voters. When we clone one candidate at random (Figure 4.2), experiments suggest that with small and large elections, the frequency of clone resistance is about 84% when $k = 1$. As we increase k, performance increases. Best results are obtained with a large number of voters and a small ϕ ($\phi \leqslant 0.8$). With a large ϕ ($\phi > 0.8$), STV_k and RP_k fail clone-proofness even when $k = 4$. From the results, STV_k and RP_k resist to cloning almost similarly. Indeed RP_k barely outperforms STV_k when $\phi \leqslant 0.8$ with an average performance rate of 0.03%.

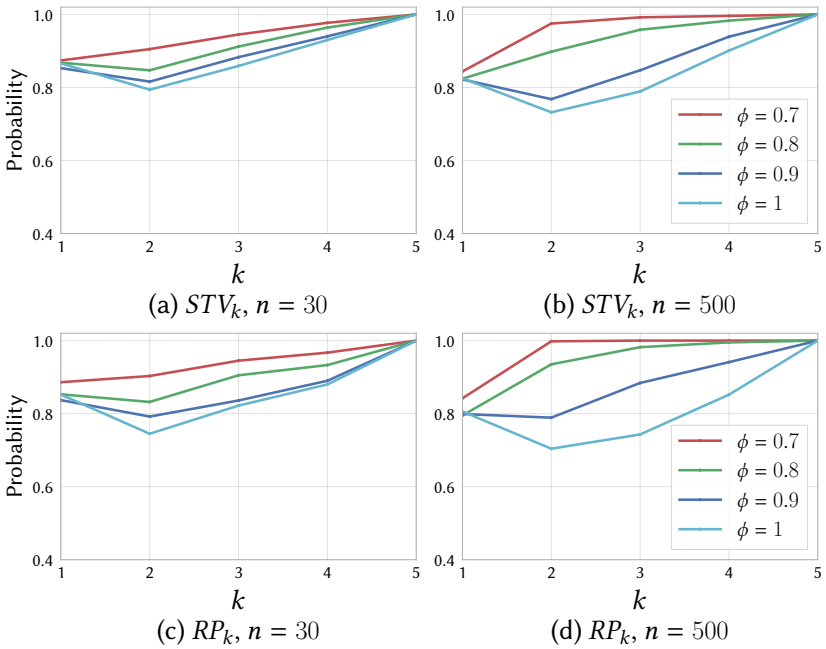


Figure 4.2: Resistance to cloning of STV_k and RP_k, a candidate at random cloned: Mallows ϕ model, $m = 5$, $n \in \{30, 500\}$ and $\phi \in \{0.7, 0.8, 0.9, 1\}$.

When $\phi = 1$, the latter behavior changes: STV_k barely outperforms RP_k with an average performance rate of 0.04%. For $\phi = 0.7$ (resp. $\phi = 0.8$), $k = 2$ (resp. $k = 4$) and $n = 500$, RP_2 (resp. RP_4) is clone-proof while STV_2 (resp. STV_4) resists to cloning with a probability of 97.5% (resp. 98%). For $\phi = 1$, STV_4 resists to cloning with a probability of 90% against 85% for RP_4.

Unsurprisingly, cloning the STV_k or RP_k winners (Figure 4.3) lead to a decrease in the probability of selecting the correct result especially when $k = 1$ and with a large number of voters since it is plurality. For instance, for STV_1 the accuracy is around 30% (resp. 0.35%) when $n = 30$ (resp. $n = 500$) and $\phi = 0.8$. As we increase k, cloning sensitivity decreases. Consistently with the above experiments, as we clone the winner, RP_k barely outperforms STV_k when $\phi \leqslant 0.9$ with a negligible performance rate

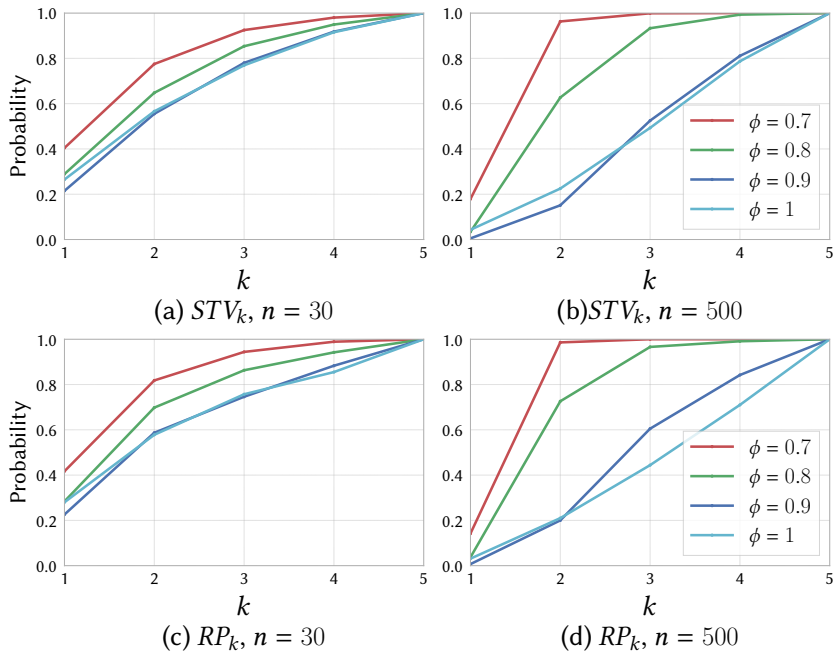


Figure 4.3: Resistance to cloning of STV_k and RP_k, winner cloned: Mallows ϕ model, $m = 5$, $n \in \{30, 500\}$ and $\phi \in \{0.7, 0.8, 0.9, 1\}$.

of 0.02%. The latter behavior changes when $\phi = 1$, e.g. For $\phi = 0.7$, RP_2 (resp. STV_3) is clone proof. For $\phi = 1$ and $n = 500$, STV_3 resists to cloning with a probability of 50% against 44% to RP_3.

4.2.2 Experiments Using Real Data Sets

We test the resistance of STV_k and RP_k to cloning using Dublin data (12 candidates and 3662 voters) with samples of n^* voters among n ($n^* < n$) where $n^* = \{30, 100, 1000\}$ then we clone: (i) one candidate at random (Figure 4.4(a) and (c)) and the STV_k and RP_k winner (Figure 4.4(b) and (d)). In each experiment 1000 random profiles are constructed with n^* voters.

With real data sets, STV_k and RP_k behave similarly. Consistently with the above experiments, the two truncated rules are clone-proof for small values of k ($k \geqslant 3$) and a large number of voters ($n = 1000$) as we clone a candidate at random. When we clone the winner, STV_k and RP_k often fail clone-proofness for small k. As we (barely) increase k, ($k \geqslant 4$); they converge to the correct selection with a large number of voters, e.g. STV_4 and RP_4 are clone-proof when cloning the winner with 1000 voters.

To conclude, resistance to cloning increases rapidly with k and decreases with ϕ. Also,

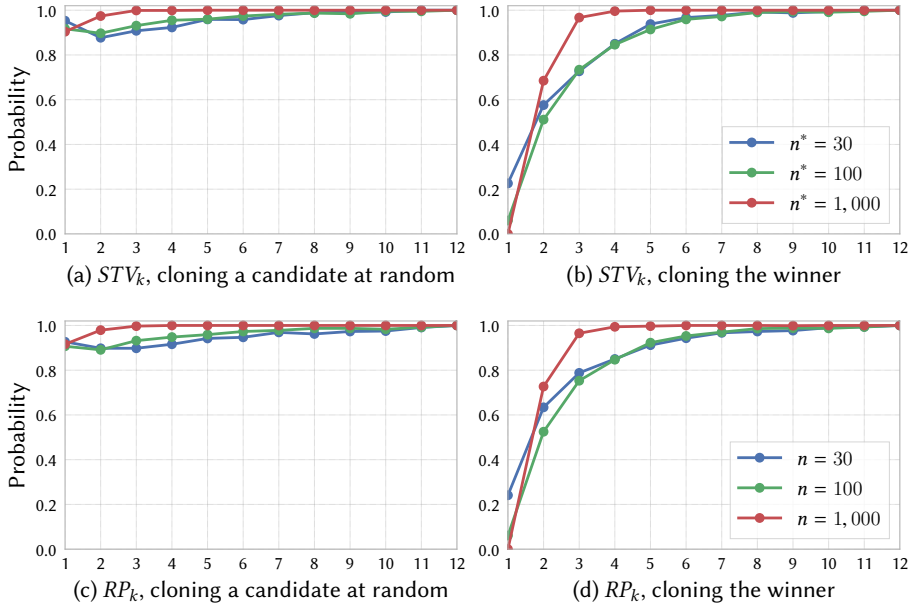


Figure 4.4: Resistance to cloning of STV_k and RP_k: Dublin data.

it significantly increases with the number of voters. With data from Mallows model, results show that RP_k and STV_k resist to cloning almost similarly with a performance rate of 0.03% for RP_k. Best results are obtained with a small ϕ and a large number of voters. For real data, the two truncated rules behave similarly where resistance to cloning increases rapidly with k, even more rapidly than with randomly generated profiles.

4.3 STV_k and Plurality with Runoff

In this section we study the probability that STV_k outputs the same winner as plurality with runoff, depending on k. STV and plurality with runoff belong to the same family of elimination-based rules. Also, when $m = 3$, STV coincides with plurality with runoff. On the other hand, plurality with runoff has much in common with plurality. Now, our family of STV$_k$-rules has plurality at one end and STV at the other. It is therefore an interesting question to know where in this spectrum we have an output that is likely to coincide with the output of plurality with runoff.

We answer this question empirically. For each experiment, we generate 1000 random preference profiles according to a Mallows distribution, for $m = 7$ candidates as we vary the number of voters $n \in \{100, ..., 500\}$ and the dispersion parameter

$\phi \in \{0.7, 0.8, 0.9, 1\}$. We simulate the elicitation of *top-k* ($k \in \{1 \dots 6\}$) preferences where for each value of k, we compare the winner of STV_k with the winner of plurality with runoff.

Figure 4.5 shows the probability that the STV_k-winner coincides with the winner of plurality with runoff. In particular, results in Figure 4.5 give the probability that STV and plurality with runoff give the same winner (for $k = m - 1$) and the probability that plurality ($=STV_1$) and plurality with runoff give the same winner (for $k = 1$).

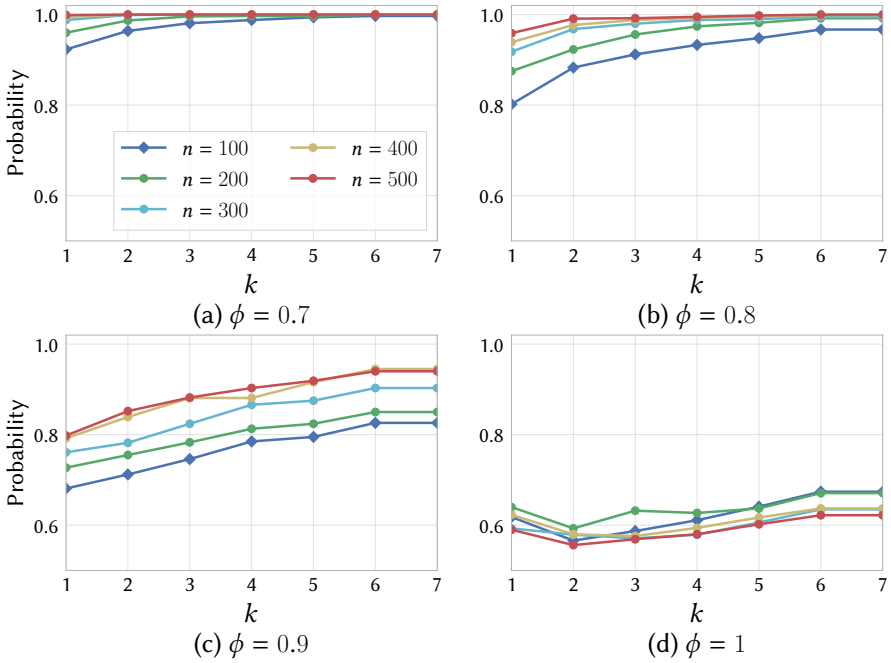


Figure 4.5: Probability that STV_k coincides with plurality with runoff: $m = 7$, $n = \{100, \dots, 500\}$, k and $\phi \in \{0.9, 0.8\}$.

For $\phi = 0.7$ (resp. $\phi = 0.8$), we find that STV_1 (resp. STV_3) and plurality with runoff always coincide with a large number of voters (i.e. $n = 500$). Unsurprisingly, with greater values of dispersion parameter $\phi \geqslant 0.9$, the probability decreases, and good results are obtained only for a large number of voters. Perhaps more surprisingly, for impartial culture, the closeness between STV_k and plurality with runoff initially decreases from $k = 1$ to $k = 2$ or $k = 3$ (depending on n), but beyond that it increases with k. We do not observe this phenomenon with smaller values of ϕ.

Table 4.1 summarizes results of the average probability (for different values of n) for which: plurality with runoff and plurality ($=STV_1$) coincide, and plurality with runoff and STV coincide. From the results, STV is closer to plurality with runoff than

plurality when $\phi \leqslant 0.8$. Indeed, when $\phi = 0.7$ (resp. $\phi = 0.8$) plurality with runoff and STV coincide with 100% (resp. 99%) accuracy while plurality with runoff and plurality coincide only in 97% (resp. 90%) of cases when $\phi = 0.7$ (resp. $\phi = 0.8$). As we increase ϕ, plurality and plurality with runoff are significantly different: they coincide only in 75% (resp. 61%) of cases when $\phi = 0.9$ (resp. $\phi = 1$). Also, when $\phi = 1$, STV is very different from plurality with runoff where we obtain the same winner in only 65% of cases.

	Plu. w/ runoff and Plu. coincide	Plu. w/ runoff and STV coincide
$\phi = 0.7$	0.97	1
$\phi = 0.8$	0.90	0.99
$\phi = 0.9$	0.75	0.89
$\phi = 1$	0.61	0.65

Table 4.1: Average probability for $n \in \{100, \ldots, 500\}$ for which: plurality with runoff and plurality coincide and plurality with runoff and STV coincide when $\phi \in \{0.7, 0.8, 0.9, 1\}$.

In the following, in the aim to determine the value of k (as a function of m) from which STV_k maximizes the probability of coinciding with plurality with runoff, we vary the number of candidates $m \in \{8, 10, 15, 20\}$ and $\phi \in \{0.7, 0.8, 0.9, 1\}$ when $n = 500$. Results are depicted in Figure 4.6 where 1000 random preference profiles are generated for each experiment.

Experiments suggest that with all values of m and when $\phi = 0.7$, STV_k for $1 \leqslant k \leqslant m$ coincides with plurality with runoff. Thus, the winner under plurality, plurality with runoff and STV is the same. As we increase ϕ to 0.8, we find that STV_k for $k \approx \frac{2}{5}m$ coincides with plurality with runoff, e.g. For $m = 15$ (resp. $m = 20$), the two rules converge from $k = 6$ (resp. $k = 8$). Consistently with the above experiments, when $\phi = 0.9$ plurality with runoff and STV coincide only in 90% on average for different values of m. Finally, for $\phi = 1$, the same behavior as in Figure 4.5(d) still occurs.

To conclude, STV is closer to plurality with runoff than plurality especially with a low value of ϕ and a large number of voters. Indeed, STV_k for $k \approx \frac{2}{5}m$ (resp. $k = 1$) coincides with plurality with runoff when $\phi = 0.8$ (resp. $\phi = 0.7$). When $\phi = 0.9$, the two rules coincide in only 90% of cases. For $\phi = 1$, STV is very far from plurality with runoff, even further from it with STV_2 which is somewhat surprising.

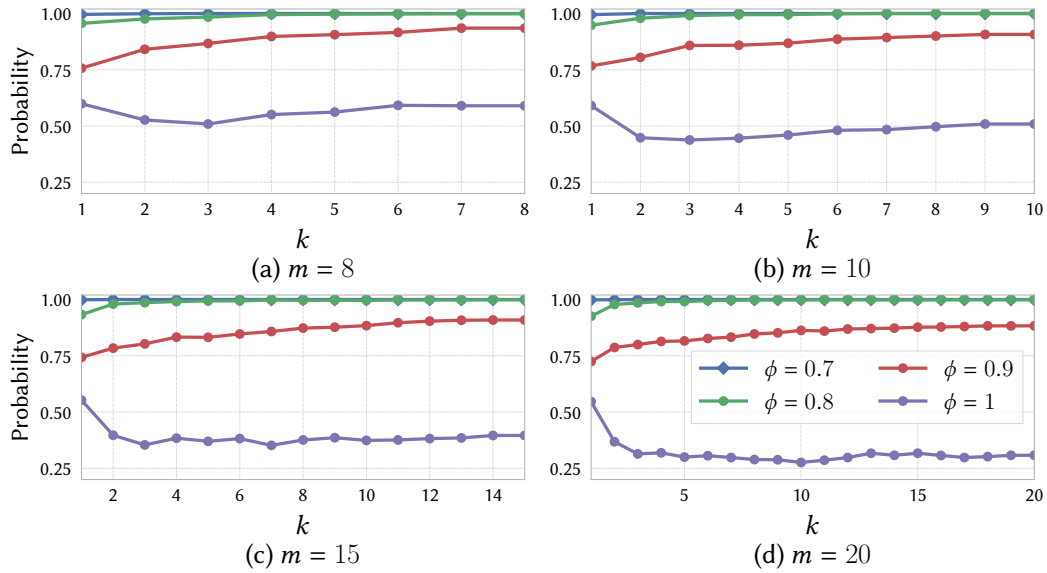


Figure 4.6: Probability that STV_k coincides with plurality with runoff when $n = 500$: vary $m \in \{8, 10, 15, 20\}$, $k \in \{1, \ldots, m\}$ and $\phi \in \{0.7, 0.8, 0.9\}$.

4.4 Possible Winners with *top-k* ballots

In this section, we study the possible winner problem for STV with *top-k* ballots. It is known that it is NP-complete to decide whether a given candidate is a possible winner for STV in a given partial profile. This follows immediately from the NP-completeness of constructive manipulation for STV [8] which is equivalent to the possible winner problem with a profile where $n - 1$ votes are fully specified, but we do not have any information about the last vote. While this case is known to be hard, the complexity of the possible winner problem for STV has not been studied for other 'shapes' of partial profiles.

We consider *truncated ballots* where a *top-k profile* is a collection of n *top-k* ballots given $k \leqslant m$. We show that the problem is polynomial-time computable if $k = 1$, but that it is NP-complete for each fixed $k \geqslant 2$. While the problem is difficult in the worst case, we show that in practice there is a simple algorithm that is able to label almost all candidates as possible winners or necessary losers.

4.4.1 Possible Winners with *top*-1 Ballots

The next result gives a closed-form characterization of possible winners for STV. This condition (without tie-breaking) was also considered (without proof) in [73] as a reduction technique for computing parallel universe STV co-winners using search algorithms.

Proposition 4.1. *Let P be a top-1 profile over A. Let $S(x_i)$ be the plurality score of x_i in P, that is, the number of votes in P ranking x_i on top. Relabel candidates so that $S(x_1) \leqslant S(x_2) \leqslant \cdots \leqslant S(x_m)$, and such that if $S(x_i) = S(x_{i+1})$ then $x_{i+1} \rhd x_i$. Then x_i is a possible winner for $STV^{\rhd}$ if there is no $j > i$ such that either $S(x_j) > \sum_{s=1}^{j-1} S(x_s)$, or $S(x_j) = \sum_{s=1}^{j-1} S(x_s)$ and $j \rhd i$.*

Proof. Assume there is a $j > i$ such that $S(x_j) > \sum_{s=1}^{j-1} S(x_s)$, or $S(x_j) = \sum_{s=1}^{j-1} S(x_s)$ and $j \rhd i$. Then, there is no way for x_i to avoid eliminating before x_j, because it can only benefit from the transfers from the votes that initially support x_s for $s < j$.

For the converse, consider any completion Q of P where x_i is ranked second in every vote whose top is not x_i. Assume there is a step where x_i is eliminated, and let j be the smallest index such that $j \neq i$ and such that x_j has not been eliminated yet at that stage. The votes supporting x_j are only those $S(x_j)$ that supported it initially. Since all candidates x_s with $s < j$ and $s \neq i$ have been eliminated, their votes have all been transferred to x_i, and thus x_i has exactly $\sum_{s=1}^{j-1} S(x_s)$ supporting votes. Since x_i is eliminated before x_j, either $\sum_{s=1}^{j-1} S(x_s) < S(x_j)$, or $\sum_{s=1}^{j-1} S(x_s) = S(x_j)$ and $j \rhd i$. □

Example 4.2. *Let P be such that $S(x_1) = 1$, $S(x_2) = 2$, $S(x_3) = 4$, $S(x_4) = 6$ and $S(x_5) = 12$. Let us give a few comments: in the best case for x_2, the unique vote for x_1 is transferred to it; at the second round, it has 3 votes, but is eliminated right after that because $S(x_3) = 4 > 3$. In the best case for x_3, at the end of the second round, all votes for x_1 and x_2 have been transferred to it and it gets 7 votes. Since $7 > S(x_4) = 6$, in this case x_4 is eliminated next and if all its votes are transferred to x_3, then x_3 gets 13 votes and wins against x_5. In conclusion: x_1 and x_2 are necessary losers and The possible winners are x_3, x_4, x_5.*

We recall that candidate x is a *necessary loser* if it is not a possible winner. It turns out that in real STV elections, we can identify many necessary losers, even if we only know the plurality scores. Figure 4.7 shows the number of possible winners in elections for the Australian House of Representatives, for which plurality scores are publicly available for each of the 150 districts ('divisions') in each election year. We can see that in most divisions, there is only 1 possible winner (because they received

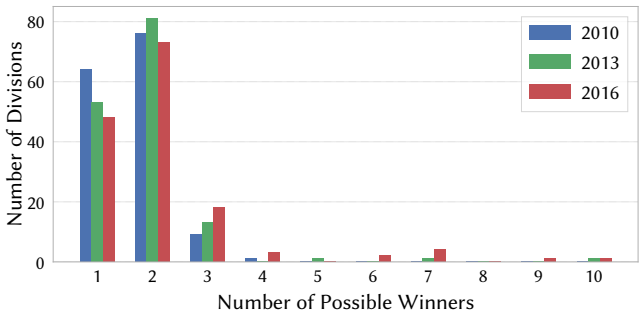


Figure 4.7: In most divisions in recent elections for the Australian House of Representatives, there were only one or two possible winners, given plurality scores.

a majority of first-place votes), or there are 2 possible winners (usually from the two major parties). It is very uncommon for there to be 3 or more possible winners.

Using Proposition 4.1, one can find the possible winners for STV given *top*-1 ballots in polynomial time. The complementary notion of *necessary* winner (i.e. a candidate who wins for all completions of the partial profile), is easily characterized: candidate x is the necessary winner iff it is top-ranked by a majority of voters (assuming n is odd; the characterization for even n depends on tie-breaking).

4.4.2 Algorithm for Possible Winners with *top-k* Ballots

The positive result for *top*-1 ballots does not extend to larger values of k: the possible STV-winner problem for *top-k* ballots is NP-complete, even for $k = 2$. This result is due to Dominik Peters and the proof is available in our accepted paper at *AAMAS-2019* [6].

Proposition 4.2. *The possible STV-winner problem is NP-complete given top-2 truncated ballots.*

In the worst case, the problem is difficult but we show that in practice there is a simple algorithm that is able to label almost all candidates as possible winners or necessary losers. Given *top-k* ballots and a candidate c, we propose an algorithm that returns three possible results: YES (in which case we know that c is a PW), NO (in which case we know that c is not a PW) and MAYBE (in which case we do not know if c is a possible winner or not).

Given *top-k* ballots, we start by removing necessary losers in the *top*-1 profile following Proposition 4.1. Then, we apply STV_k until all k candidates in some ballot have

been eliminated. We refer to the obtained partial profile by an *incomplete plurality profile* which corresponds to a partial profile where the top candidate of some voters is unknown. For an incomplete plurality profile, we associate $(s_\emptyset, S(x_1), \ldots, S(x_m))$, where $s_\emptyset$ is the number of voters for whom we do not know what their top candidate is, and $S(x_i)$ are the number of voters for whom we know that x_i is their top candidate (i.e. the plurality score of x_i).

Example 4.3. *Let P be a profile with $n = 14$ and $A = \{a, b, c, d, e, f\}$ where $\rhd$: $a \rhd b \rhd c \rhd d \rhd e$:*

1	$a \succ e$
2	$b \succ f$
4	$c \succ e$
7	$d \succ e$

1	$\cancel{a} \succ \cancel{e}$
2	$b \succ \cancel{f}$
4	$c \succ \cancel{e}$
7	$d \succ \cancel{e}$

When applying STV_2 (on the left profile), candidates e and f are eliminated first, then candidate a having the fewest number of votes is eliminated next. The obtained partial profile corresponds to the incomplete plurality profile (the profile on the right) issued from P such that: $s_\emptyset = 1$, $S(b) = 2$, $S(c) = 4$ and $S(d) = 7$.

Proposition 4.3 gives an approximation of possible winner for STV given top-k ballots.

Proposition 4.3. *Let R be a top-k profile and V the incomplete plurality profile obtained by applying STV_k to R until all k candidates in some ballots have been eliminated. Given $(s_\emptyset, S(x_1), \ldots, S(x_m))$ associated to V, relabel candidates so that $S(x_1) \leqslant S(x_2) \leqslant \cdots \leqslant S(x_m)$, and such that if $S(x_i) = S(x_{i+1})$ then $x_{i+1} \rhd x_i$. We propose two sufficient but not necessary conditions to label the available candidates where, we say that $x_i \in A$ is a:*

1) *necessary loser for $STV^\rhd$ if there is $j > i$ such that either $S(x_j) > s_\emptyset + \sum_{s=1}^{j-1} S(x_s)$, or $S(x_j) = s_\emptyset + \sum_{s=1}^{j-1} S(x_s)$ and $j \rhd i$.*

2) *possible winner for $STV^\rhd$ if x_i is the STV winner when we repeatedly do the following:*

   - *place x_i on top of the empty votes,*
   - *recompute the plurality scores and remove the candidate with the fewest votes.*

if neither (1) nor (2), then we do not know if x_i is a possible winner or not.

Proof.

1) Assume there is a $j > i$ such that $S(x_j) > s_\emptyset + \sum_{s=1}^{j-1} S(x_s)$, or $S(x_j) = s_\emptyset + \sum_{s=1}^{j-1} S(x_s)$ and $j \rhd i$. Then, there is no way for x_i to avoid eliminating before x_j, because it can only benefit from the transfers from the votes that initially support x_s for $s < j$.

2) Placing x_i on top of the empty votes corresponds to a possible extension of the incomplete plurality profile. If x_i wins when applying STV on the obtained profile then, by definition, x_i is a possible winner.

3) Since conditions (1) and (2) are sufficient but not necessary, some candidates will not be labeled as possible winners or necessary losers.

Given *top-k* ballots, Algorithm $PW_{top\text{-}k}$ outlines the whole process. It allows to answer the question: *Is candidate $x \in A$ a possible winner?* by returning three possible results: (1) YES: in which case x is a possible winner, (2) NO: in which case x is a necessary loser and, (3) MAYBE: in which case we don't know if x is a possible winner or not.

Algorithm 4.1: $PW_{top\text{-}k}$

Input: $R = \{\succ_i^k, \ldots, \succ_n^k\}, A$
Initialize: $YES \leftarrow \emptyset$, $MAYBE \leftarrow \emptyset$, $L \leftarrow \emptyset$
$NL_1 \leftarrow$ necessary losers given *top*-1 ballots using Proposition 4.1
repeat
    $d \leftarrow$ candidate ranked first by the fewest voters (breaking ties)
    Remove d from the set of available candidates
    $L \leftarrow L \cup \{d\}$
until *there is an empty vote*
$NL_2, PW, A \setminus (NL_2 \cup PW) \leftarrow$ label candidates in $A \setminus L$ using Proposition 4.3
$YES \leftarrow PW$
$NO \leftarrow L \cup NL_1 \cup NL_2$
$MAYBE \leftarrow A \setminus (YES \cup NO)$
return $YES, NO, MAYBE$

Example 4.4. *Let us consider the following top-2 ballots of 6 voters over $A \in \{a, b, c, d, e, f\}$ where $\rhd$: $a \succ b \succ c \succ d \succ e \succ f$:*

Following Algorithm $PW_{top\text{-}k}$, given top-1 ballot of each voter the scores of different candidates are: $S(c) = 0$, $S(b) = 0$, $S(f) = 1$, $S(d) = 1$, $S(a) = 1$ and $S(e) = 3$. Candidates b, c and f are necessary losers, giving: $NL_1 = \{b, c, f\}$. Then we apply

1 vote : $e \succ b$	1 vote : $d \succ f$
1 vote : $a \succ e$	1 vote : $f \succ c$
1 vote : $e \succ a$	1 vote : $e \succ f$

STV_2: candidates b and c are removed first then f with the lowest number of votes, giving: $L = \{b, c, f\}$. Now, the vote $f \succ c$ is empty, the resulting incomplete plurality profile is: $s_\emptyset = 1$, $S(d) = 1$, $S(a) = 1$ and $S(e) = 3$.

Following the conditions in Proposition 4.3, there is no more necessary losers to remove, giving: $NL_2 = \emptyset$. Now we test whether candidates a, d and e are possible winners or not by ranking them on top of the empty vote as follows:

- *Rank a on top of the empty vote*

Round 1: eliminate d		Round 2: eliminate $e \Rightarrow a$ wins	
1 vote : $e \succ \not b$	1 vote : $d \succ \not f$	1 vote : $e \succ \not b$	1 vote : $\not d \succ \not f \succ a$
1 vote : $a \succ e$	1 vote : $\not f \succ \not c \succ a$	1 vote : $a \succ e$	1 vote : $\not f \succ \not c \succ a$
1 vote : $e \succ a$	1 vote : $e \succ \not f$	1 vote : $e \succ a$	1 vote : $e \succ \not f$

- *Rank d on top of the empty vote*

Round 1: eliminate a		Round 2: eliminate $d \Rightarrow e$ wins	
1 vote : $e \succ \not b$	1 vote : $d \succ \not f$	1 vote : $e \succ \not b$	1 vote : $d \succ \not f$
1 vote : $a \succ e$	1 vote : $\not f \succ \not c \succ d$	1 vote : $\not a \succ e$	1 vote : $\not f \succ \not c \succ d$
1 vote : $e \succ a$	1 vote : $e \succ \not f$	1 vote : $e \succ \not a$	1 vote : $e \succ \not f$

- *Rank e on top of the empty vote*

Round 1: eliminate d		Round 2: eliminate $a \Rightarrow e$ wins	
1 vote : $e \succ \not b$	1 vote : $d \succ \not f$	1 vote : $e \succ \not b$	1 vote : $\not d \succ \not f \succ e$
1 vote : $a \succ e$	1 vote : $\not f \succ \not c \succ e$	1 vote : $a \succ e$	1 vote : $\not f \succ \not c \succ e$
1 vote : $e \succ a$	1 vote : $e \succ \not f$	1 vote : $e \succ a$	1 vote : $e \succ \not f$

Giving the above completions, we have: $YES = \{a, e\}$ and $MAYBE = \{d\}$. To conclude, given the top-2 ballots and $A = \{a, b, c, d, e, f\}$ we can say that: (1) b, c and f are necessary losers, (2) a and e are possible winners, and (3) we don't know if d is a PW or not.

In the following, we present simulation results when $m = 7$, $n = 100$ and varying ϕ. For each $k \in \{1, \ldots, 6\}$ we measure the average number of possible winners (denoted by YES), necessary losers (denoted by NO) and candidates that we cannot decide on (denoted by MAYBE). For each experiment, we draw 1000 random profiles. Results are reported on Figure 4.8. Depicted results suggest that with *top*-1 ballots, it is always possible to classify the available candidates to possibles winners and necessary losers. As we increase k ($k \geqslant 2$), removing necessary losers leads to reduce the set of candidates that we cannot decide on, thus improve the identification of possible winners in a *top*-k profile. Also, from Figure 4.8 we notice that the candidates that we cannot decide on (MAYBE) are more present in profiles with low value of $\phi = 0.7$ and when $k = 2$. As we increase ϕ, the set of candidates that we cannot decide on decreases. For example, when $\phi = 0.7$ only one candidate out of 7 is neither possible winner nor necessary loser given *top*-2 ballots. For $\phi = 0.8$, MAYBE's set is present only with a probability of 2.7% and it decreases to 1.6% (resp. 1.08%) when $\phi = 0.9$ (resp. $\phi = 1$).

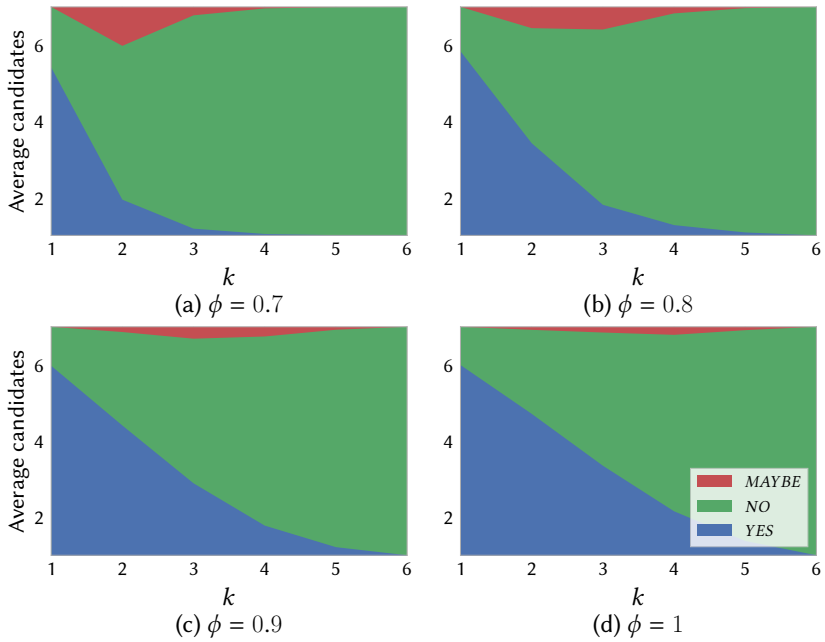


Figure 4.8: Identifying possible winners (YES), necessary losers (NO) and candidates that we cannot decide on (MAYBE), when $m = 7$, $n = 100$ and $\phi \in \{0.7, 0.8, 0.9, 1\}$: Mallows ϕ model.

4.5 Communication Protocols for STV

We now take a different path: we consider the determination of the STV winner for the complete profile, and consider *interactive* protocols (cf. Chapter 2) where voters may report their preferences incrementally, when the central authority asks them to do so. Indeed, we are no longer interested in computing an approximation of STV, but in computing the real STV winner.

In Section 4.5.1 we start from Conitzer and Sandholm's protocol [25] and we give a practical amelioration of it. In Section 4.5.2 we propose an improvement of the latter protocol by identifying in each step *strong necessary losers* (necessary losers whose removal does not change the winner) and removing them. Then, the question we are interested in is : *What is the sufficient number of bits (in average) that must be communicated by the voters to the voting center so that the winner is determined?* We answer the latter question in Section 4.5.3 where we study the average communication complexity of these protocols and their practical communication complexity, based on randomly generated profiles and on real data sets from *PrefLib*.

4.5.1 Conitzer and Sandholm's Protocol

Conitzer and Sandholm [25] studied the communication complexity of several voting rules, by giving specific protocols and by proving lower bounds obtained via fooling sets. For the case of STV, they showed that a lower bound on the communication complexity of $\Omega(n \log m)$, which is the cost of communicating every voter's top choice. They were able to match this lower bound up to a log factor using the following protocol for STV, which we call P_1.

Protocol 4.2: P_1

```
for each voter i ∈ N do
    ask i to send the name of her top candidate
repeat
    d ← candidate ranked first by the fewest voters (breaking ties)
    Remove d from the set of available candidates
    for each voter i ∈ N do
        if the top candidate of i was d then ask i to send the name of her next
          preferred candidate
until there exists a candidate c ranked first by a majority of voters
return c
```

Example 4.5. *Let* $A = \{a, b, c, d\}$ *and* $P = \{3 : a \succ c \succ d \succ b, 3 : c \succ a \succ b \succ d, 1 : d \succ b \succ a \succ c, 1 : b \succ d \succ c \succ a\}$ *with tie-breaking priority* $a \rhd b \rhd c \rhd d$*. We run* P_1*. In the first round,* d *is eliminated. We ask the voter who supports* d *to name her next preferred candidate. We get* $P = \{3 : a, 3 : c, 2 : b\}$*. Next,* b *is eliminated. We ask the two voters supporting* b *for their next preferred candidate. We get* $P = \{4 : a, 4 : c\}$*. By tie-breaking,* a *wins.*

Conitzer and Sandholm [25] show that P_1 requires communication of at most $O(n(\log m)^2)$ bits. To see this, note that at a step where k candidates remain, at most $\frac{n}{k}$ voters rank the eliminated candidate first. Thus, the number of times we need to ask voters for their new favorite is at most $\frac{n}{m} + \frac{n}{m-1} + \ldots + n$, which is bounded by $n \log m$. We can actually say something more precise: the worst case occurs when, in each step, all candidates are tied for elimination; in this case, when k candidates remain, exactly $\frac{m}{k}$ voters will send $\log k$ bits, and this goes on until $k = 1$. This gives an exact worst case cost of $n\left(\log m + \sum_{k=1}^{m-1} \frac{\log k}{k+1}\right)$ bits. In the subsequent discussion, we will not focus on the number of bits transmitted, and instead will count how often voters have to report their favorite remaining candidate. We refer to this as the *number of questions* asked. It is easy to see that, in the worst-case, for fixed n and m, the number of questions asked by protocol P_1 is $P_{Worst} = n \cdot \left(1 + \sum_{k=1}^{m-1} \frac{1}{(k+1)}\right)$.

4.5.2 An Improved Protocol

At each step in the execution of the protocol, the central authority has partial knowledge of the votes. Therefore, it makes sense to identify those candidates that can still win (the possible winners) and those that cannot (the necessary losers). This is especially useful when interaction with the voters takes time: assume that the vote is about a meeting date; the execution of the protocol can take several days (due to some voters reacting slowly to their emails). If at some point in the execution of the protocol, we know that for sure the meeting will not be on November 22 nor November 24, this is useful information for voters, who can plan something else on these two days, and for the central authority, which does not have to pre-book a room for these days.

We start by noticing that at each step of the protocol P_1, the information known by the protocol is essentially a *top*-1 profile. Therefore, Proposition 4.1 is applicable and we can calculate, at each step of the protocol, the remaining possible winners and necessary losers. Knowing the possible winners (and the necessary losers) at each step of the protocol is useful information, but it turns out that eliminating a candidate as soon as it becomes a necessary loser can change the final outcome:

Example 4.6. *Let us consider* $P = \{5 : c \succ a \succ d \succ e \succ b, 1 : a \succ e \succ b \succ d \succ c, 2 : e \succ b \succ d \succ a \succ c, 2 : b \succ a \succ e \succ c \succ d\}$*, and let the tie-breaking priority be* $a \rhd b \rhd c \rhd d \rhd e$*. The winner for* P *is* c*. The necessary losers given top-1 ballots are* a*,* d *and* e*. If we eliminate those three candidates, the winner is* b*.*

The source of the paradox lies in the tie between the initial score of c and the cumulative score of a, b, d, and e. However, we now identify a subset of necessary losers that can always be eliminated without changing the winner.

Let us rename the candidates so that they are ranked by increasing order of plurality score, then by tie-breaking priority: for each i, either $S(x_i) < S(x_{i+1})$, or $S(x_i) = S(x_{i+1})$ and x_{i+1} has priority over x_i. We say that x_i is *dominant* if either $S(x_i) > \sum_{j=1}^{i-1} S(x_j)$, or $S(x_i) = \sum_{j=1}^{i-1} S(x_j)$ and x_i has priority over x_j, where j is the largest index smaller than i such that x_j is a dominant candidate. Note that to check whether x_i is dominant, we need to iterate over all candidates x_j, $j < i$. We say that x_j is a *strong necessary loser* if there is an index $i > j$ such that x_i is dominant. By Proposition 4.1, a strong necessary loser is a necessary loser, but the converse is not always true: in Example 4.6, only d and a are strong necessary losers.

Proposition 4.4. *The removal of strong necessary losers does not change the winner.*

Proof. Assume x_i is the largest dominant candidate in the sequence (x_1, ..., x_{i-1} are thus the strong necessary losers). Then no matter in which order the candidates $x_1, \ldots, x_{i-1}$ are eliminated, they will all be eliminated before all candidates in $\{x_i, ..., x_m\}$. Therefore, after $i-1$ elimination steps, the currently eliminated candidates are exactly x_1, ..., x_{i-1}. Eliminating them directly allows to "jump" over $i-1$ elimination steps, and the rest of the process continues exactly as if this jump had not been performed. □

Protocol P_1 can be improved by checking at each step if there are strong necessary losers, and if so, eliminate them in addition to the current loser, and then send a query to all voters whose current top candidate has just been eliminated. A further improvement is possible by querying one voter at a time: it might be possible to rule out further candidates without knowing every voters' current top choice. To do this we need to generalize the notion of dominant candidate and strong necessary loser to *incomplete plurality profiles*. Assume again that candidates are renamed so that they are ranked by increasing order of plurality score, then by tie-breaking priority. A candidate x_i is a *safe necessary loser* (SNL) if there is $j > i$ such that $S(x_j) > s_\emptyset + S(x_1) + \cdots + S(x_{j-1})$. Clearly, if x_i is a safe necessary loser for an incomplete plurality profile then it is a

strong necessary loser for every of its completions into a plurality profile, which implies that eliminating an SNL candidate cannot change the winner. This leads us to define the following protocol:

Protocol 4.3: P_2

1 Query every voters' top candidate
2 Let $(s_\emptyset, S(x_1), \ldots, S(x_m))$ be the resulting plurality profile, with $s_\emptyset = 0$
3 **repeat**
4 **for each** *safe necessary loser* x_i **do**
5 Remove x_i
6 $s_\emptyset \leftarrow s_\emptyset + S(x_i)$
7 Select an empty voter and query their new top alternative
8 Let x_j be this new top alternative
9 $S(x_j) \leftarrow S(x_j) + 1$; $s_\emptyset \leftarrow s_\emptyset - 1$
10 **until** *the set of possible winners is a singleton*

The protocol terminates because at each step, either a candidate is removed, or $s_\emptyset$ is decreased by 1. If $s_\emptyset$ queries are made successively without any candidate being removed, then $s_\emptyset$ reaches 0 and then there is at least one SNL (the current plurality loser with lowest priority). The winner returned at the end of the protocol is the STV winner because removing SNLs does not influence the winner. It can be proven that P_2 is as least as cheap as P_1, in the sense that no voter is queried more often with P_2 than with P_1.

Example 4.7. *Let the candidates be* a, b, c, d, e. *After the first step, the plurality scores are* $(S(a) : 1, S(b) : 1, S(c) : 3, S(d) : 4, S(e) : 8)$. a *and* b *are SNL and are removed. The resulting incomplete plurality profile is* $(s_\emptyset : 2, c : 3, d : 4, e : 8)$. *We query a voter who votes for* d, *giving* $(s_\emptyset : 1, S(c) : 3, S(d) : 5, S(e) : 8)$. c *is now a SNL and is removed, giving* $(s_\emptyset : 4, S(d) : 5, S(e) : 8)$. *We query a voter who votes for* e, *resulting in* $(s_\emptyset : 3, S(d) : 5, S(e) : 9)$. d *is now an SNL, and the winner is* e.

Finally, in the aim to further improve P_2, we propose an adjustment on line 7 in P_2. Indeed, several ways exist to select the empty voter among the available ones to query her next preferred candidate. In the following we propose three possible options to make the selection:

1) select a voter randomly among the available voters with empty votes. We refer to this method as P_2^{Rand}.

2) select the voters with the minimal number of questions asked during the execution of P_2 among the voters with empty votes and choose one voter at random. We refer to this method as $P_2^{Rand-Min}$.

3) select the voters with the maximal number of questions asked during the execution of P_2 among the voters with empty votes and choose one voter at random. We refer to this method as $P_2^{Rand-Max}$.

Example 4.8. *Let $A = \{a, b, c, d, e, f\}$. At some step of the execution of P_2, some voters were asked to submit their next preferred, resulting in the profile $P = \{4 : c, 2 : b, 1 : e \succ c, 1 : e \succ b, 2 : a, 1 : d \succ a, 2 : f \succ b\}$. Giving P, candidate d is removed first, then f, e and a. The resulting incomplete plurality profile is $(s_\emptyset : 3, s_c : 5, s_b : 5)$.*

Now, we choose a voter to query her next preferred candidate among the voters who voted for candidate a. Three possible voters are available, namely: $(2 : a, 1 : d \succ a)$. Under P_2^{Rand}, we choose one voter at random among the available ones. Under $P_2^{Rand-Min}$, we choose one voter among the two whose most preferred candidate was a, namely: $2 : a$. Finally, under $P_2^{Rand-Max}$, the voter who voted for d then a (i.e. $1 : d \succ a$) will be the next voter to query.

4.5.3 Evaluation of the Protocols

This section evaluates the average communication complexity of P_1 and P_2 with the different randomized methods to select the voter with the empty vote. We discuss experiments using Mallows model and real data. Our objective is to determine the communication complexity, in terms of the number of questions voters need to answer on average. Note that, for plurality elections, this number is 1, while for (say) Borda elections, this number is m in the worst-case. We find that STV only asks 2 or 3 questions for each voter when $m = 7$, or in the Dublin election data. Note that in the experiments where we measure the average communication cost, we will report only on P_2 since the average number of questions voters need to answer is almost the same with different randomized methods.

4.5.3.1 Experiments using Mallows ϕ model

For each experiment, we draw 1000 random profiles. In the first set of experiments, we present simulation results with $m = 7$ and $n = 100$, and let ϕ vary. We count the number of questions asked by P_1 and P_2 with $\phi \in \{0.7, 0.8, 0.9, 1\}$. Figure 4.9 shows

the average communication cost of P_1, P_2 and compares it to P_{Worst}, the worst-case cost of P_1.

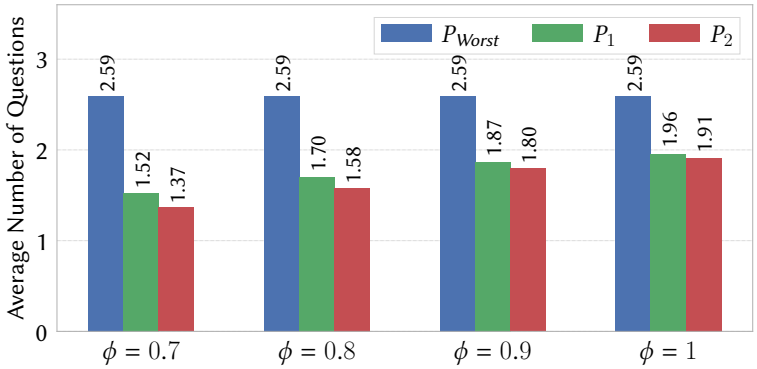


Figure 4.9: Average communication cost, Mallows.

Results suggest that in practice, both P_1 and P_2 ask fewer questions than in the worst case (P_{Worst}). P_2 asks fewer questions than P_1 for all parameter values, but the difference is more significant for lower ϕ. When $\phi = 1$, more information is needed from voters under both P_1 and P_2, and the savings of P_2 are smaller.

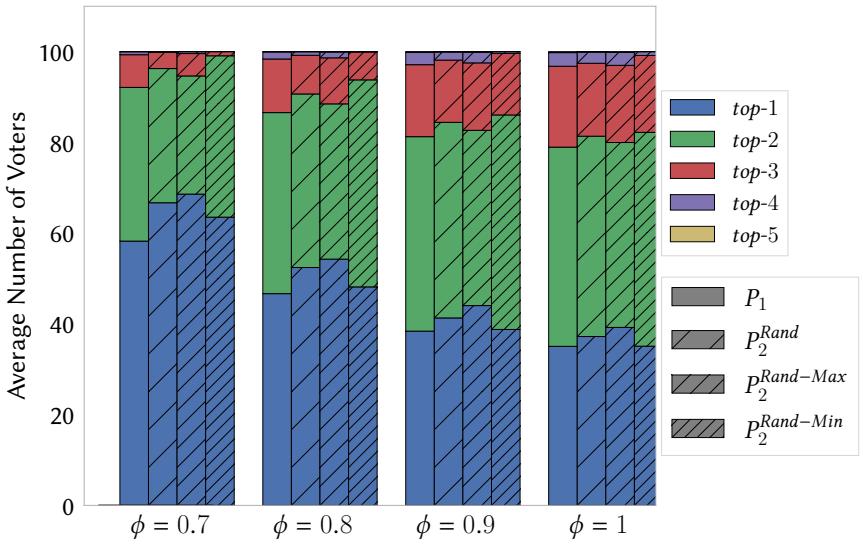


Figure 4.10: Average number of voters who were asked k questions when $m = 7$, $n = 100$ and $\phi \in \{0.7, 0.8, 0.9, 1\}$ under: P_1, P_2^{Rand}, $P_2^{Rand-Max}$ and $P_2^{Rand-Min}$; Mallows ϕ model.

To better understand the difference in the behavior of P_1 and P_2 and the advantage of using the different randomized methods, we now plot, for each k, how many voters

are needed to answer k questions using P_1, P_2^{Rand}, $P_2^{Rand-Max}$ and $P_2^{Rand-Min}$ when $n = 100$, $m = 7$ and $\phi \in \{0.7, 0.8, 0.9, 1\}$. Results are reported on Figure 4.10 taken over 1000 random samples. For instance, the first bar means that under P_1 and when $\phi = 0.7$, 58 (resp. 34, 7,1) voters out of 100 were asked one (resp. 2, 3, 4) question(s).

Results show that all randomized methods of P_2 outperform P_1 where P_2 often only needs to ask a voter for her top alternative, and then never asks another question. Indeed, asking voters for their most preferred candidate decreases as we increase ϕ with all protocols, e.g. Under P_1 and when $\phi = 0.7$ (resp. $\phi = 0.8$, $\phi = 0.9$, $\phi = 1$), 58 (resp. 46, 38, 35) voters were asked one query. Now, when comparing $P_2^{Rand-Min}$ and $P_2^{Rand-Max}$, we find that:

- under $P_2^{Rand-Min}$ all voters were asked no more than three questions with all values of ϕ (except when $\phi = 1$ only one voter was asked four questions). With small values of ϕ, $P_2^{Rand-Min}$ tends to ask one question to more than the half of the voters. As we increase ϕ, $P_2^{Rand-Min}$ often asks 2 questions to almost half of the voters.
  - for $\phi = 0.7$, 63.5% of voters were asked one query, 35.5% were asked two questions and 1 voter was asked three questions.
  - for $\phi = 1$, 35% of voters were asked one query, 47% were asked two questions, 17% voters were asked three questions and 1 voter was asked four questions.
- under $P_2^{Rand-Max}$ some voters were asked four questions especially when $\phi \geqslant 0.8$. Compared to the other protocols, $P_2^{Rand-Max}$ asks the highest number of voters for their top alternative and then never asks another question.
  - for $\phi = 0.7$, 68.6% of voters were asked one question, 26% were asked two questions, 5 voters were asked three questions.
  - for $\phi = 1$, 39% of voters were asked one question, 41% were asked two questions, 17% voters were asked three questions and 3 voters were asked four questions.

Table 4.2 summarizes results of the average number of voters (for different ϕ) asked to submit their *top-k* for $k \in \{1, \ldots, 4\}$ with $P_2^{Rand-Max}$ and $P_2^{Rand-Min}$.

From the results, $P_2^{Rand-Max}$ tends to promote some voters to others by: maximizing the number of voters asked for their *top*-1 candidate and minimizing those asked for their second preferred by allowing voters to submit their *top*-3 and *top*-4 candidates. Unlike $P_2^{Rand-Max}$, $P_2^{Rand-Min}$ tends to be egalitarian by minimizing the number of

	$P_2^{Rand-Max}$	$P_2^{Rand-Min}$
top-1	51,67%	46.39%
top-2	34.90%	44.11%
top-3	11.73%	9.4%
top-4	1.7%	0.1%

Table 4.2: Average number of voters asked to submit their *top*-k for $k \in \{1, 2, 3, 4\}$ when $\phi \in \{.7, .8, .9, 1\}$ with $P_2^{Rand-Max}$ and $P_2^{Rand-Min}$.

questions asked to all voters: first ask voters who have not said much for their most preferred candidate and then ask the maximum number of voters for their *top*-2 ballots in order to minimize the number of voters to query for their *top*-3.

4.5.3.2 Experiments using Real Data sets

We consider data sets from *Preflib* [58] to run our experiments. Most of the available data contain some incomplete ballots. Table 4.3 shows the number of alternatives, complete and partial ballots in each data sets. For example, Dublin North data contains 43942 votes: 3662 are complete ballots and the remaining votes are partial ones.

Data Set	m	All Ballots	Complete Ballots	Partial Ballots
Dublin North	12	43942	3662	40280
Dublin West	9	29988	3800	26188
Debian	7	504	327	177
Meath	14	64081	2490	61591
ERS	10	380	43	337
Glasgow	9	6900	548	6352
Sushi	10	5000	5000	5000

Table 4.3: Data sets from *Preflib*.

In order to evaluate our protocols with real data sets, we consider only votes with complete preferences. Figure 4.11 shows the average number of questions voters are asked by the protocols on real data sets. P_2 performs better than P_1 by asking 5–10% fewer questions. Both protocols are much better than the worst-case analysis suggests, by about 40%.

Figure 4.12 shows how many voters for different data sets were asked k questions under P_1, P_2^{Rand}, $P_2^{Rand-Max}$ and $P_2^{Rand-Min}$. From the results, the different randomized

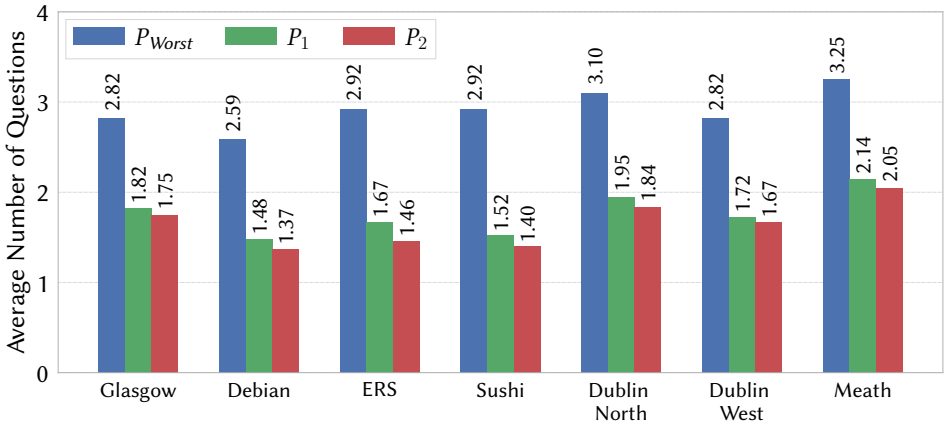


Figure 4.11: Average communication cost, real data sets.

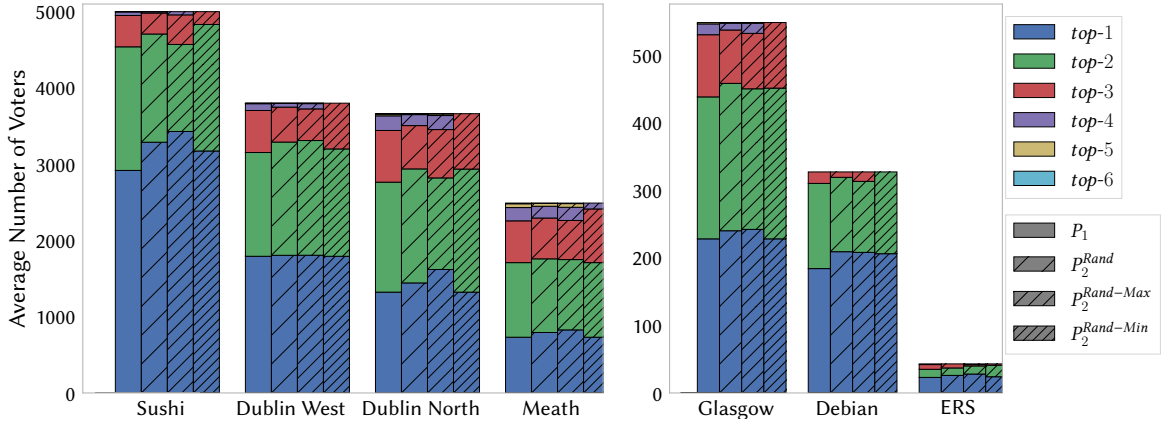


Figure 4.12: Number of voters who were asked k questions with different real data sets.

methods of P_2 ask more voters to submit their top preferred candidate compared to P_1. This behavior is more noticeable with Sushi, Dublin North and Debian than the other data sets. Consistently with the experiments in Figure 4.10, $P_2^{Rand-Min}$ asks fewer questions to all voters compared to $P_2^{Rand-Max}$. For instance, with Dublin North data having 3662 voters, $P_2^{Rand-Min}$ asks 1322 voters one question, 1614 voters two questions and 726 three questions while $P_2^{Rand-Max}$ asks 1619 voters one question, 1201 voters two questions, 634 three questions, 184 four questions and 23 voters six questions.

4.6 Conclusion

In this chapter, we have focused on Single Transferable Vote (STV) voting rule. In the first part, given *top-k* ballots, we have tested the resistance of STV_k and RP_k to

cloning where we have showed that the two truncated rules behave almost similarly to the introduction of similar candidates. Results on randomly generated profiles show that resistance to cloning increases rapidly with k and decreases with ϕ. Also, it significantly increases with the number of voters. With real data, resistance to cloning increases rapidly with k, even more rapidly than with randomly generated profiles. Then, we have studied empirically the probability that STV_k and plurality with runoff coincide. Our findings are that STV is closer to plurality with runoff than plurality especially with low values of ϕ and a large number of voters. Indeed, STV_k for $k \approx \frac{2}{5}m$ (resp. $k = 1$) coincides with plurality with runoff when $\phi = 0.8$ (resp. $\phi = 0.7$). For $\phi = 1$, STV is very far from plurality with runoff.

In the second part, we have investigated the computational complexity of the possible winner problem for *top-k* ballots under STV where we have showed that for $k = 1$, it can be solved in polynomial time, but is NP-complete when $k \geqslant 2$. While the problem is difficult in the worst case, we have showed that in practice there is a simple algorithm that is able to label almost all candidates as possible winners, necessary losers and candidates that we don't know if they are possible winners or not. Results show that the proposed algorithm succeeds to identify possible winners and necessary losers and to reduce the set of candidates that we cannot decide on.

In the third part, we have considered interactive communication protocols for STV. Building on a protocol proposed by Conitzer and Sandholm (2005), we have showed how we can reduce the amount of communication required in practice. We have then studied empirically the average communication complexity of these protocols, based on randomly generated profiles, and on real-world election data. Our results show that STV has low communication cost in practice when using the proposed protocol P_2.

Chapter 5

Search Methods for Vote Elicitation

Contents

In this chapter, we propose an incremental vote elicitation process using heuristic methods to guide the choice of the next voter to ask. Given top-k preferences, these heuristics will be used to choose at each round the voter to whom it is most relevant to ask to complete his k-vote by giving his next candidate (thus ranked k+1). For this purpose, first we propose to use the Monte Carlo Tree Search (MCTS) approach in a vote elicitation context to select the most prominent voter to ask in each round based on the Upper Confidence Bounds for Trees (UCT) evaluation function. Second, we propose a search heuristic able to select the voter (under certain conditions) for whom we want to reveal more information. Results on the evaluation of the average number of questions asked show the practical advantage of reducing the communication cost when using the proposed heuristics in an incremental elicitation process under Borda and Harmonic rules.

5.1 Introduction

In the previous sections we have considered exact methods for interactive vote elicitation to choose the voter to query in each round until the winner is known. In this chapter, we explore heuristic methods, based on evaluation functions that will guide the choice of the next voter to query. We still focus on top-k queries where one voter will be asked at a time and, if she is queried for the k^{th} time, she will be asked to provide her k^{th} preferred candidate (as in Section 4.5.2).

We are, in particular, interested by the use of *Monte Carlo Tree Search* (MCTS) which is a probabilistic and heuristic driven search algorithm that combines the classic tree search implementations alongside machine learning principles of reinforcement learning. The principle of MCTS is to analyze the most promising moves by incrementally expanding the search tree based on random sampling of the search space [20, 45] MCTS is essentially used in AI games, namely all the top AI solutions for Go (game).

In this chapter, our goal is to built an interactive vote elicitation process, based on evaluation functions, that will guide the choice of the next voter to query in each round. Given top-k ballots, these evaluation functions are able to predict the most prominent voter to ask to submit her next preferred (ranked k+1) candidate in order to determine the winner with the minimum number of questions asked. For this purpose, first we propose to use the MCTS's technique based on the *Upper Confidence Bounds for Trees* (UCT) [45] evaluation function in a vote elicitation context. Second, we propose a simple heuristic evaluation function which proceeds in two main steps: (1) we identify two key candidates for whom we want to gather more information, and then (2) if there is at least a voter whose current ballot contains neither of these two candidates, query such a voter; otherwise, query a voter whose current ballot contains one of these two candidates.

The use of these evaluation functions in a vote elicitation process will allow us to reveal the most relevant information about the voters thus; reduce the information communicated between them. Finally, we use the necessary winner (cf. Chapter 2) to check whether the information elicited is sufficient to output the winner or not. In this context, Xia and Conitzer [75] show that for any positional scoring rule and given a partial profile, it can be checked in polynomial time whether a given candidate is a necessary winner. We adapt their algorithm to the case of top-k ballots by considering two positional scoring rules, namely: Borda and Harmonic.

This chapter is organized as follows: In Section 5.2 we describe the basic steps of MCTS algorithm and the UCT evaluation function used to select the best move in a

game. In Section 5.3 we describe how to use the MCTS algorithm in a vote elicitation context where we consider the elicitation problem as a game between the voting center and the voters. Then we present the proposed vote elicitation process using MCTS with truncated ballots. In Section 5.4 we propose the search heuristic. Finally, Section 5.5 evaluates the proposed heuristics by measuring the number of questions asked to voters on average in order to determine the winner under Borda and Harmonic rules.

5.2 Monte Carlo Tree Search (MCTS)

The effectiveness of MCTS relies on selecting the most promising move to perform in a tree according to an evaluation function. Each node v in the tree represents a given position called a *state* of the game, denoted by $S(v)$, with the root node representing the current state. Each parent node is directly linked to child nodes representing *legal move* leading to subsequent states, i.e. it represents the move from one state (the parent) to another (the child). Note that any parent node may have as many children as there are legal moves from the state for which it represents.

Example 5.1. *Figure 5.1 shows the start game of TicTacToe. Assume that the computer will make the first move, then the root node (the parent) represents an empty* 3×3 *Tic-Tac-Toe board that can have nine children representing the possible moves (9 legal moves).*

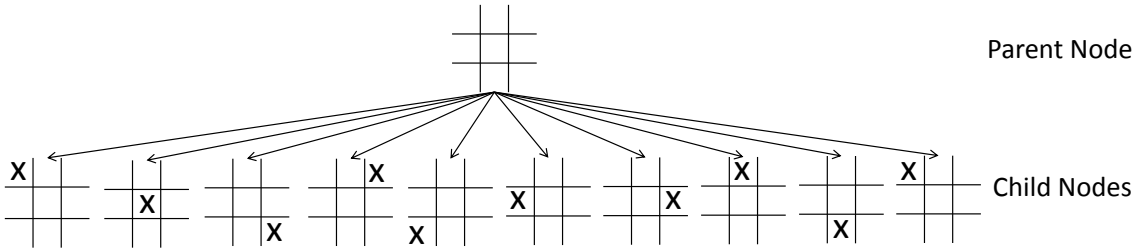


Figure 5.1: Example of a tree representing the start game of TicTacToa.

The MCTS is an iterative process based on four steps. An outline of these steps is depicted in Figure 5.2 for one iteration:

1) **Selection**: In the selection step, the tree is traversed starting at the root node where an evaluation function is applied recursively until the most promising expandable node is reached. A node is *expandable* if it represents (i) a *non-terminal* state (a specific condition has not been reached yet), and (ii) is not fully expanded (it is a leaf node i.e. no known children so far).

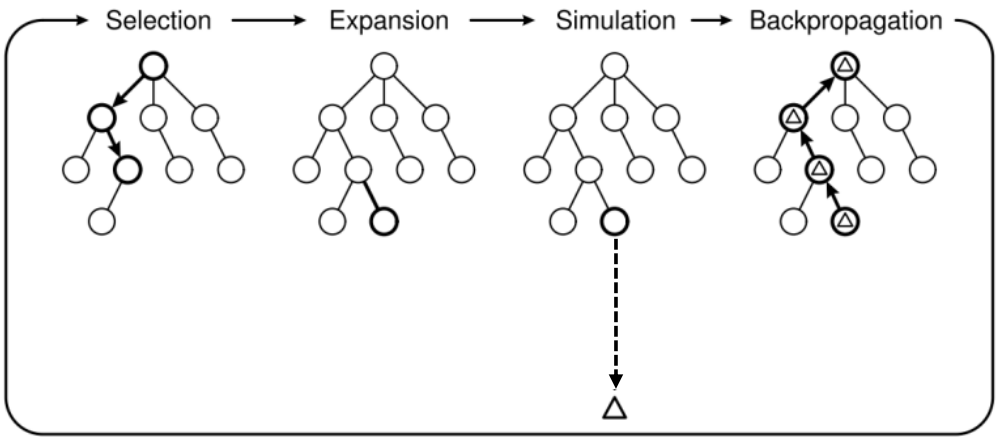


Figure 5.2: One iteration of the general MCTS approach.

2) **Expansion**: In the expansion step, for a given node from the selection step; we create one or more child nodes according to available actions at the current state.

3) **Simulation**: In this step, a simulation (aka. a playout) is run from the new node until a *terminal state* is reached. The simulation makes random moves and returns some value representing the result of the simulation on the new added node denoted by Δ.

4) **Backpropagation**: In the Backpropagation step, the simulation result is back-propagated through the previously traversed nodes to update their statistics, starting from the node on which we ran the simulation and ending at the root node. Finally, the move played by the program is the child of the root with the highest number of visits. This step ensures that each node's value reflect simulations performed in the subtrees that they are part of.

In the selection step, picking which node to visit is performed using a selection strategy. The most popular one is the *Upper Confidence Bounds for Trees* (UCT) introduced by Kocsis and Szepesvari [45] and expressed for each state $S(v)$ by:

$$UCT(S(v)) = \frac{Q(v)}{N(v)} + C \times \sqrt{\frac{ln\ N(p)}{N(v)}} \tag{5.1}$$

where $N(v)$ corresponds to the number of times v has been visited. $Q(v)$ is the total reward (score) of all payouts that was performed on v. p is the parent of v and C is the exploration constant parameter, generally; can be tuned experimentally.

UCT is based on the balancing of *exploitation* and *exploration* in games. Indeed, the first part of the UCT's equation tends to promote moves with the highest win value

which corresponds to the exploitation phase. While the second part corresponds to the exploration phase that tends to explore unexplored parts of the tree that currently may seem less than optimal but may be more effective in the long term. Note that when different nodes have the same UCT value, we choose one of them randomly.

Algorithm 5.1 outlines the MCTS steps (adapted from Browne *et al.* [17]).

Algorithm 5.1: Pseudocode for Monte-Carlo tree search.

create root node v_0 with state S_0
if v_0 *is terminal* **then**
    return game over
else
    while *within computational budget* **do**
        /* Selection and Expansion */
        $Test \leftarrow True$
        while v_0 *is non-terminal and Test* **do**
            if v_0 *is not fully expanded* **then**
                choose $a \in$ untried actions from $A(S(v_0))$
                add a new child v_l to v_0 with $S(v_l) = f(S(v_0), a)$ and $a(v_l) = a$
                $Test \leftarrow False$
            else
                $v_0 \leftarrow \underset{v' \in \text{ children of } v_0}{argmax} \frac{Q(v')}{N(v')} + C\sqrt{\frac{ln\, N(v_0)}{N(v')}}$
        /* Simulation */
        $s \leftarrow S(v_l)$
        while s *is non-terminal* **do**
            choose $a \in A(s)$ uniformly at random
            $s \leftarrow f(s, a)$
        $\Delta \leftarrow$ reward of s
        /* Backpropagation */
        while v_l *is not null do* **do**
            $N(v_l) \leftarrow N(v_l) + 1$
            $Q(v_l) \leftarrow Q(v_l) + \Delta$
            $v_l \leftarrow$ parent of v_l
    $BestChild \leftarrow \underset{v' \in \text{ children of } v_0}{argmax} \frac{Q(v')}{N(v')}$
    return $a(BestChild)$

Each node v in the tree is associated with four pieces of information: $S(v)$ corresponds to the associate state, $N(v)$ is the number of visits, $Q(v)$ is the total reward and $a(v)$

is the incoming action to v. $A(state)$ corresponds to the set of actions available on a specific state. The term $state' = f(state, action)$ corresponds to the next state $state'$ decided based on the current *state* and the chosen *action*. The MCTS's steps are repeated for a fixed number of iterations. After the process is finished, the *final move* is performed: one of the children of the root node is selected as best move (line 23). Different final move strategies exist to determine the best child [21] including: *Max Child* (select the child that maximizes $Q(v')$), *Robust Child* (select the child that maximizes $N(v')$) etc.

Note that the complexity of MCTS is $O(\ell PI)$ where ℓ is the branching factor, P is the number of simulations of each child and I is the number of iterations.

The following example illustrates the MCTS algorithm. Each node v is labeled by $Q(v)/N(v)$. To alleviate notation, instead of $S(v_i)$ we will simply write S_i.

Example 5.2. *Figure 5.3 presents three iterations of the MCTS algorithm when* $C = \frac{1}{\sqrt{2}}$ *starting by an initial state* S_0 *with values 0/0 (a state with 0 reward and 0 simulation performed).*

- *In the first iteration, we start by an initial state* S_0*; a decision must be taken between two possible moves* S_1 *and* S_2*. Under the selection step, we compute the UCT value for* S_1 *and* S_2 *following Eq. 5.1:* $UCT(S_1) = UCT(S_2) = \infty$*. Since the two states have the same value, we select a state randomly. Assume we take* S_1 *which is not fully expanded. In the expansion step, a child node* S_3 *is added to* S_1 *and a playout is run from it. Suppose that the simulation step ends in* $\Delta = 20$*, thus; the value of the new node* S_3 *is 20/1. Finally, we backpropagate the result and update the statistics of* S_1 *and* S_0*.*

- *In the second iteration, we start by an initial state* S_0 *with the updated states' values from the first iteration. A decision must be taken between two possible moves* S_1 *and* S_2 *where* $UCT(S_1) = \frac{20}{1} + \frac{1}{\sqrt{2}}\sqrt{\frac{ln(1)}{1}} = 20$ *and* $UCT(S_2) = \infty$*. We select the child node with the highest UCT value which corresponds to state* S_2*. Now,* S_2 *is not fully expanded. In the expansion step, a child node* S_3 *is added to* S_2 *and a playout is run from it. Suppose that the simulation step ends in 10, thus; the value of the new node is 10/1. The ascendents of* S_2 *are updated. Now the reward of* S_0 *is 30 with 2 visits.*

- *In the third iteration, we again have to choose between* S_1 *and* S_2 *so we recompute the UCT value of these nodes:* $UCT(S_1) = \frac{20}{1} + \frac{1}{\sqrt{2}}\sqrt{\frac{ln(2)}{1}} = 20.58$ *and* $UCT(S_2) = \frac{10}{1} + \frac{1}{\sqrt{2}}\sqrt{\frac{ln(2)}{1}} = 10.58$*, then* S_1 *is the selected node. A child node is added to* S_1

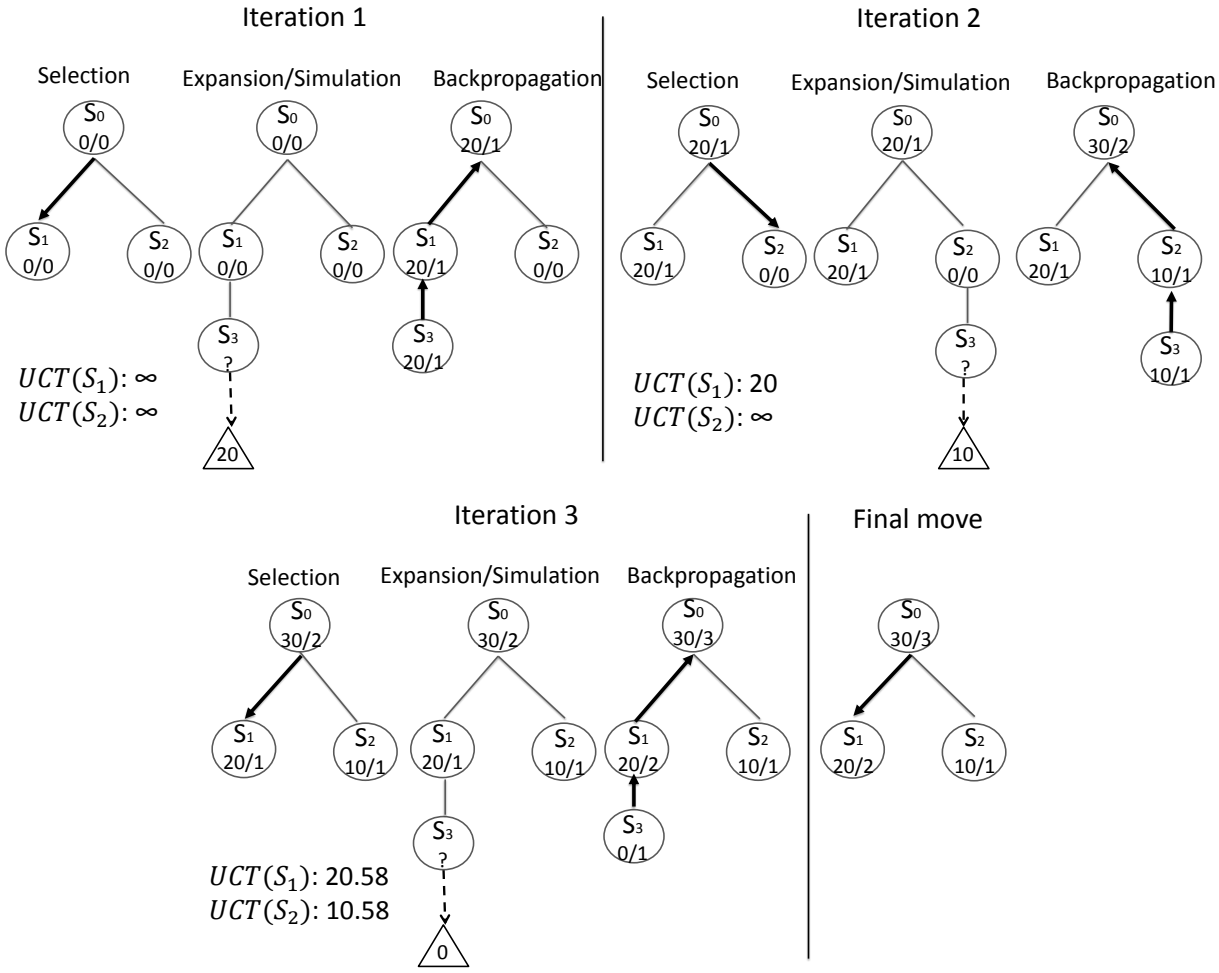


Figure 5.3: Example of 3 iterations using MCTS algorithm.

and a simulation is run that ends in 0 (i.e. the simulation results is a loss) and results in S_3 with value 0/1. We backpropagate the result and update the nodes' values.

Finally, given the generated (partial) tree from the third iteration, using either the Max Child or Robust Child strategies; the algorithm will return S_1 as final move.

5.3 Vote Elicitation Protocol Using MCTS

In this section we propose to use Monte Carlo Tree Search's technique in vote elicitation when considering top-k truncated ballots. Indeed, aggregating voters preferences can

be seen as a particular game with two players: the voting center and the set of voters, where 'the set of voters' is considered as a single player:

- *the voting center*: asks a specific voter to complete her top-k vote, and;
- *the set of voters*: the selected voter by the voting center submits her next preferred candidate to the voting center whenever he was asked to do so.

In this game, players do not play against each other; however, the goal is to make moves by selecting the relevant voter and ask him to complete his partial vote in the aim to determine the correct winner. To communicate information between the voting center and the voters, we consider queries in the form of *next-best-alternative* by asking the voter: *Who is your next preferred candidate ?* Note that the next best alternative of voter i and her previous votes correspond to answering a *top-k* query.

Each node v corresponds to a voter i that holds the top-k candidates of i represented by L– where L corresponds to the top-k vote and "–" means that the voting center has no information about how the remaining candidates are ranked. At the start of the game, the root node corresponds to the initial available knowledge held by the voting center where each voter gives her top-1 ballot. The voting center may ask any voter i to expand her vote by submitting her second preferred candidate. More generally, after different iterations, the voting center may ask any voter i to expand her k vote by submitting her $(k+1)^{th}$ preferred candidate. Thus, the root node will be linked to n children that correspond to possible legal moves (i.e. the different voters to ask) and the voting center will choose only one voter to query. Moving from one node to another corresponds to an elicitation process where a specific voter is selected and asked to complete her top-k ballot. The game ends when, given the voters' preferences, a *necessary winner* (cf. Chapter 2) is reached.

Our goal is to build an incremental vote elicitation protocol using the MCTS technique by minimizing the number of questions asked to voters in each round for a specific voting rule. To this end, the MCTS Algorithm 5.1 is used to guide the choice of the voter to be asked in each round by adapting its evaluation function, final move and terminal state as follows:

Evaluation function: In Eq. 5.1, the total reward $Q(v)$ corresponds to the total number of questions asked to voters in all playouts performed on v. In a vote elicitation context, the constant C used to adjust the degree of exploration of the algorithm is strongly related to the number of voters (n) and the number of candidates (m). Since n and m can be high; the higher the constant C, the more the algorithm will tend to

explore the less well rated moves. That is why we will assume that $C = \alpha(n \times m)$ where $\alpha \in [0, 1]$.

Final move: Since our aim is to ask the minimum number of questions to voters, in Algorithm 5.1 we select the best child (line 23) using $argmin\ Q(v')$ which corresponds to the child node with the minimal number of questions of all playouts that have been performed.

Terminal state: When using MCTS in a vote elicitation context, given the partial voters' preferences, the terminal state corresponds to reaching a *necessary winner* for a specific voting rule.

Fortunately, given a partial profile, for any positional scoring rule it can be checked in polynomial time whether a given candidate is a necessary winner (Xia and Conitzer [75]). In the following, we adapt their algorithm to the case of top-k ballots when considering a tie breaking over candidates for PSR. Let R be a partial profile containing the top-k votes of n voters. Each voter i reports a ranking list $\succ_i^k$ of k out of m candidates. For each candidate $a \in A$, we compute a minimal score and a maximal score (these two scores have already been considered by Konczak and Lang [46]):

- The minimal score of a candidate x is the lowest possible score x could obtain when considering all complete extensions of R. An unranked candidate x in $\succ_i^k$ will be ranked in the lowest position in the extension of $\succ_i^k$ denoted by $C_{min}(\succ_i^k)$. Let $C_{min}(x, R) = \{C_{min}(\succ_i^k), \ldots, C_{min}(\succ_n^k)\}$ denote the completion of the partial profile R given x. Then, $S_{min}(x, R)$ corresponds to the minimal score of candidate x when considering the completions $C_{min}(x, R)$ with respect to a given PSR.
- The maximal score of a candidate x is the highest possible score x could obtain when considering all complete extensions of R. An unranked candidate x in $\succ_i^k$ will be ranked in the highest position in the extension of $\succ_i^k$ denoted by $C_{max}(\succ_i^k)$ i.e. x is ranked in position $k+1$ in $C_{max}(\succ_i^k)$. Let $C_{max}(x, R) = \{C_{max}(\succ_i^k), \ldots, C_{max}(\succ_n^k)\}$ denote the complete completion of the partial profile R given x. Then, $S_{max}(x, R)$ corresponds to the maximal score of candidate x when considering the completions $C_{max}(x, R)$ with respect to a given PSR.

Example 5.3. *We consider a partial profile R of 4 voters having partial preference orders over 4 candidates $A = \{a, b, c, d\}$ with tie breaking $\rhd$: $a \rhd b \rhd c \rhd d$, where $R = \{a, c \succ d, a, b \succ a\}$. We consider Borda rule defined by a scoring vector $\vec{s}_{Borda} = (3, 2, 1, 0)$. The minimal and maximal scores of candidate c are:*

- *$S_{min}(c, R) = 3$ when considering a possible completion $C_{min}(c, R) = \{a \succ b \succ d \succ \boldsymbol{c}, c \succ d \succ b \succ a, a \succ d \succ b \succ \boldsymbol{c}, b \succ a \succ d \succ \boldsymbol{c}\}$*

- $S_{max}(c, R) = 8$ *when considering a possible completion* $C_{max}(c, R) = \{a \succ \boldsymbol{c} \succ b \succ d, c \succ d \succ b \succ a, a \succ \boldsymbol{c} \succ b \succ d, b \succ a \succ \boldsymbol{c} \succ d\}$.

We consider the resolute version of positional scoring rules where whenever candidates are tied, we break ties using a predefined priority ordering over candidates $\rhd$. Then, given top-k ballots and $\rhd$, $x \in A$ is a necessary winner with respect to a given positional scoring rule, if and only if for each $y \in A, y \neq x$ either: (1) $S_{min}(x, R) > S_{max}(y, R)$, or (2) $S_{min}(x, R) = S_{max}(y, R)$ and $x \rhd y$.

To test whether a is a necessary winner for positional scoring rules, given $A = \{a, b, c, d\}$, Xia and Conitzer [75] test each of the alternatives $y \in \{b, c, d\}$ in turn, attempting to find a completion of the partial profile that maximizes the score difference $S(y) - S(a)$ by raising y as high as possible and lowering a as low as possible. Then, a is a necessary winner if $S(a) > S(y)$ for all $y \in \{b, c, d\}$. The characterization used by Xia and Conitzer corresponds exactly to the above result when considering truncated ballots with a one small detail is that we also consider ties between candidates.

Corollary 5.1. *Given top-k ballots and* $\rhd$, $x \in A$ *is a necessary winner with respect to a given positional scoring rule, if and only if for each* $y \in A, y \neq x$ *we have :*

1) $S_{min}(x, R) > S_{max}(y, R)$, *or*

2) $S_{min}(x, R) = S_{max}(y, R)$ *and* $x \rhd y$.

Proof.
(1 $\Rightarrow$ x is a necessary winner) proved by Xia and Conitzer [75]

(2 $\Rightarrow$ x is a necessary winner) Assume (2) holds then $S_{min}(x, R) = S_{max}(y, R)$. If $S_{max}(y, R)$ never reaches $S_{min}(x, R)$, then y will always have a score lower than x. If $S_{max}(y, R)$ reaches $S_{min}(x, R)$, then x will be always be the winner because $x \rhd y$. Therefore, x is a necessary winner.

If neither (1) nor (2) holds, then according to Xia and Conitzer x is not a necessary co-winner. Thus, regardless of the tie breaking used, x cannot be a necessary winner. □

Example 5.4. *Let us consider the same partial profile* R *in Example 5.3. Given* R, *the minimal and maximal scores of candidates under Borda rule are:*

- $S_{min}(a, R) = 8$, $S_{min}(b, R) = 3$, $S_{min}(c, R) = 3$ *and* $S_{min}(d, R) = 2$,
- $S_{max}(a, R) = 9$, $S_{max}(b, R) = 8$, $S_{max}(c, R) = 8$ *and* $S_{max}(d, R) = 7$,

Since $S_{min}(a, R) > S_{max}(d, R) = 7$ *and* $S_{min}(a, R) = S_{max}(b, R) = S_{max}(c, R)$ *and* $a \rhd b \rhd c$*, then* a *is a necessary winner.*

Now, we are ready to present the proposed elicitation protocol using MCTS which we call P_{MCTS} and proceeds as follows: Given each voter's top candidate in R, the MCTS Algorithm 5.1 is applied to determine the best voter to ask to complete her top-k vote. Let i be the selected voter. Then, we ask voter i to submit her next preferred candidate and we update the partial profile R by adding i's next preference. We repeat this process until we reach a necessary winner given R, for a specific voting rule. Note that the complexity of MCTS in a vote elicitation process turns out to be $O(Imn^2)$ for each move performed because: in the worst case we will have n children and the number of simulations performed on each child to find a necessary winner is $n(m-1)$ (since we already have the top preferred candidate of each voter).

Protocol 5.2: P_{MCTS}

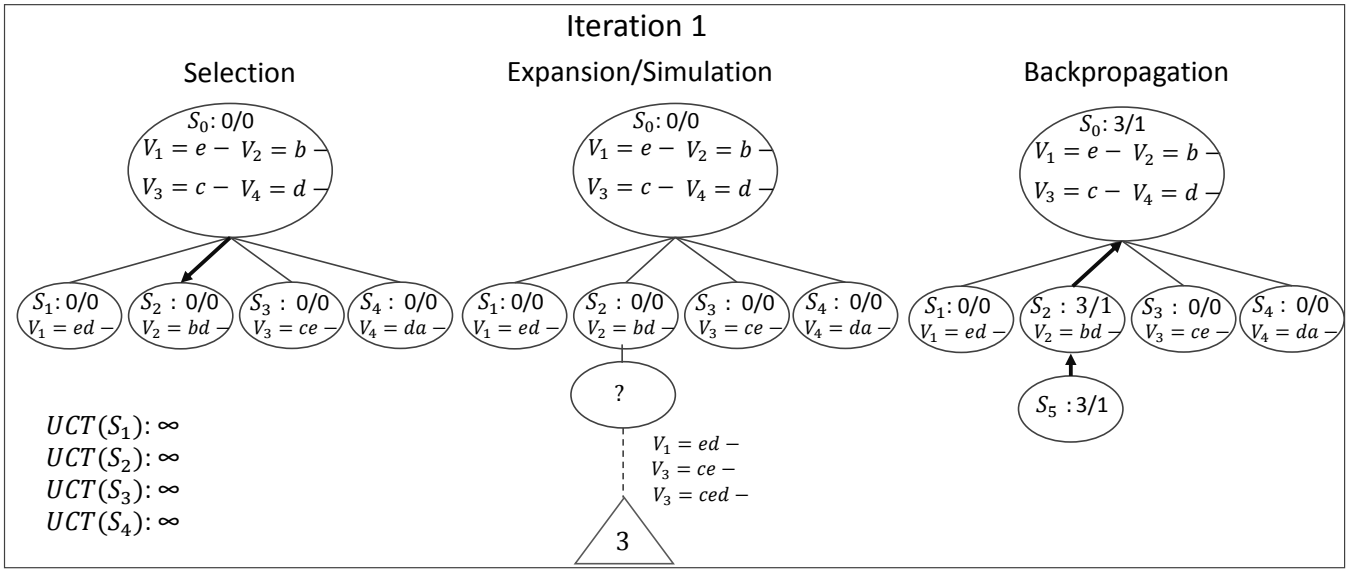
for each *voter* $i \in N$ **do**
  $R \leftarrow$Ask i to submit the name of her top candidate
repeat
  $i \leftarrow$ the selected voter returned using MCTS in Algorithm 5.1
  ask i to send the name of her next preferred candidate
  update R by adding the new vote of the selected voter i
until *there exists a necessary winner* c *given* R;
return c

Example 5.5. *Let us consider the setting of 4 voters with the following complete preferences over 5 candidates* $m = \{a, b, c, d, e\}$ *with* $\rhd$: $a \rhd b \rhd c \rhd d \rhd e$*:*

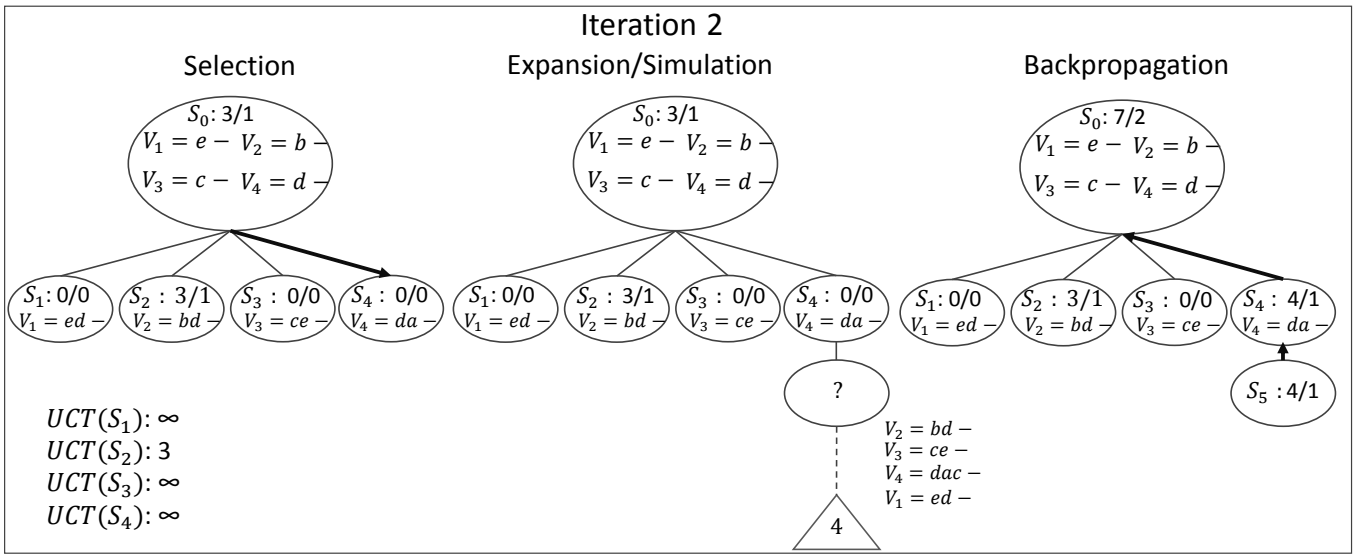
V_1	$e \succ d \succ a \succ b \succ c$
V_2	$b \succ d \succ a \succ c \succ e$
V_3	$c \succ e \succ d \succ b \succ a$
V_4	$d \succ a \succ c \succ b \succ e$

We consider Borda rule. Following Protocol 5.2, the voting center has the most preferred candidate of each voter, i.e. $R = \{e, b, c, d\}$*. We apply MCTS given the partial votes to determine the choice of the voter to query. Figures below describe the MCTS's four steps performed on four iterations in order to select the first best move given* R *when* $C = 0.3 \times n \times m = 6$*.*

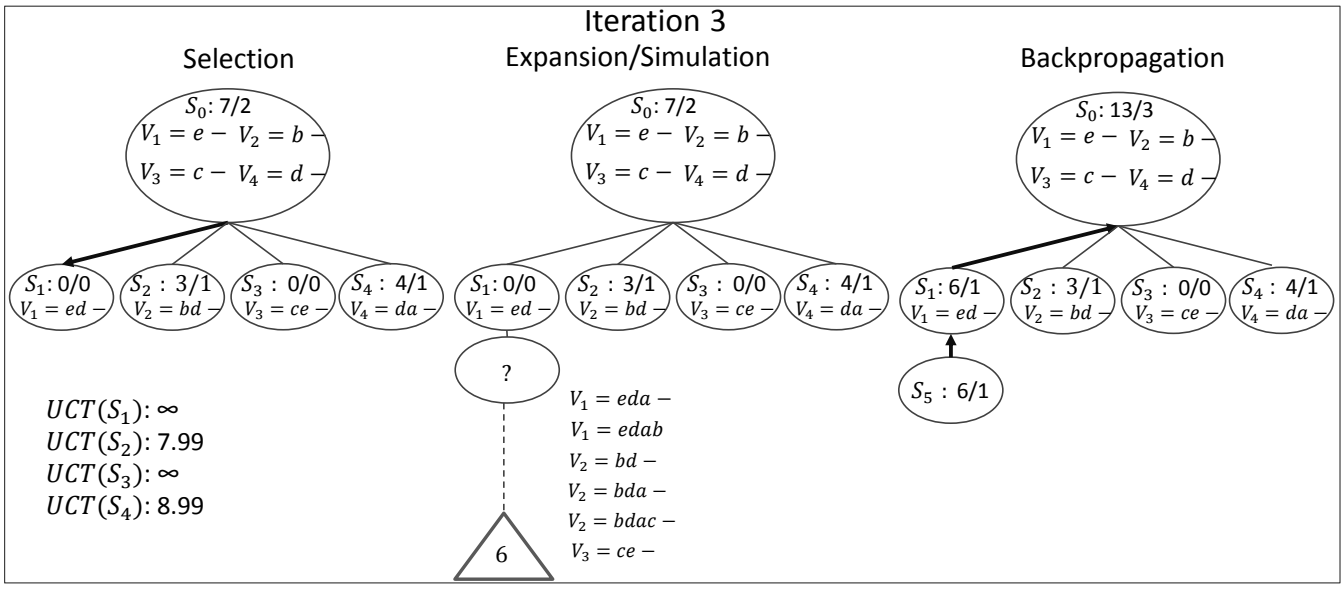
In the first iteration, the program starts at state S_0 containing R with value 0/0. Possible moves consist in asking the 4 voters to cast their second preferred choice with a winning possibilities of 0/0. We compute the UCT value for different moves which is equal to ∞ for all states. In this case, we choose a random move among the four available. Assume we select S_2. We are now at an unexpanded node (S_2). We expand S_2 by adding a node and a playout is run from the latter by asking voters randomly to report their next preferred candidate until a necessary winner is reached. A possible simulation consists in asking V_1 for her top-2 preferred candidate then V_3 for her second and third preferred, resulting in $R = \{e \succ d, b \succ d, c \succ e \succ d, d\}$ and d is a necessary winner. The simulation step ends in 3 (corresponds to the number of questions asked to voters). Finally, we backpropagate the result and update the nodes.



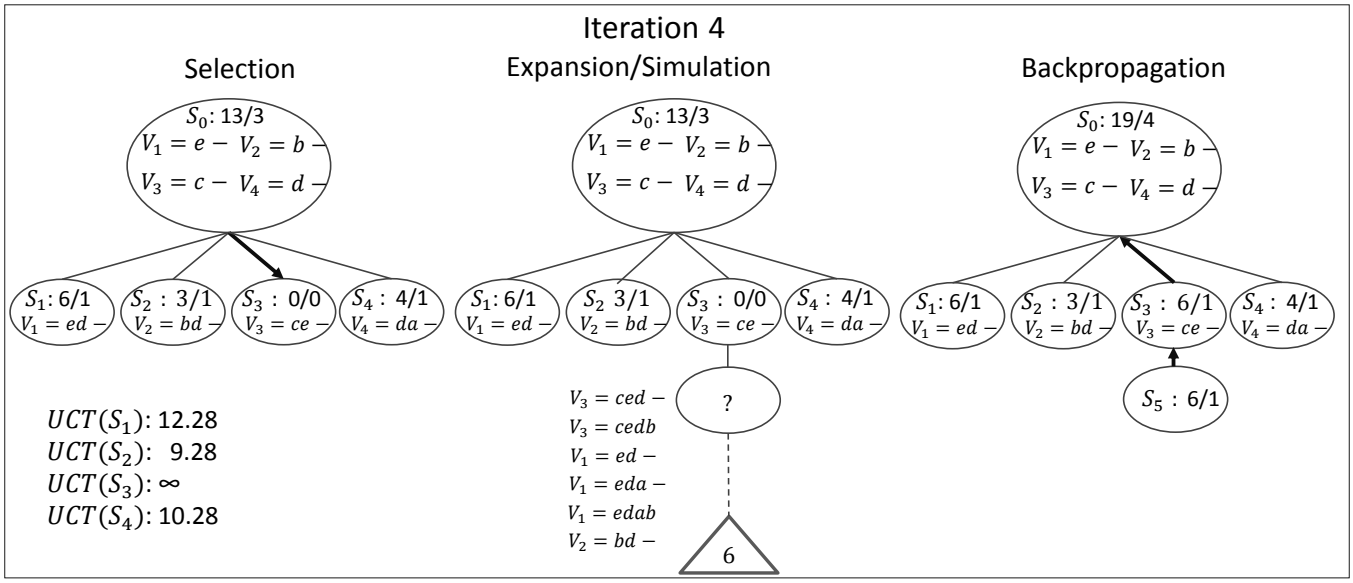
In the second iteration, we go back to the initial state with the updated values generated from the first iteration. Again we compute the UCT value for different states: $UCT(S_3) = \frac{3}{1} + 6\sqrt{\frac{ln(1)}{1}} = 3$ and ∞ for the other states; a node is chosen randomly from the latter. Assume we select S_4 and we run a simulation from it which results in 4 questions asked to voters (ask V_2 then V_3 for their second preferred candidate, V_4 for her third preferred and V_1 for her second preferred). This gives $R = \{e \succ d, b \succ d, c \succ e, d \succ a \succ c\}$. Then, we backpropagate the result to the root node. Now S_0 has a value of 7/2.



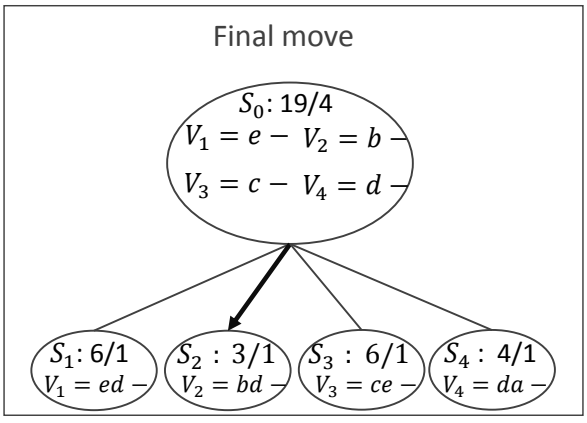
In the third iteration, giving the initial state, we recompute the UCT value which is equal to 7.99 and 8.99 for S_2 and S_4, respectively; and ∞ for S_1 and S_3. Let S_1 be the next move to perform, we ran a simulation on it that results in 6 questions asked that will be backpropagated to the root node to update its value.



In the fourth iteration, we select state S_3 with the highest UCT value and we run a playout on it that ends in 6 and we backpropagate the result. Now, S_0 has a value of 19/4.



Based on the result of the four iterations we select the final move with the minimal value which corresponds to S_2.



The described steps allow us to determine the first move to perform given the most preferred candidate of each voter. Following Protocol 5.2, we ask voter 2 to submit her next preferred candidate resulting in $R = \{e, b \succ d, c, d\}$. Since there is no necessary winner given R, in the next step S_2 will be the root node and, similarly we perform different iterations to select the next voter to ask. This process is repeated until a necessary winner is reached given the different moves performed. Now, assume that in the next moves we ask: V_1 for her second preferred candidate and V_3 for her second and third preferred, resulting in $R = \{e \succ d, b \succ d, c \succ e \succ d, d\}$ and d is a necessary winner. Thus, we stop eliciting preferences from voters and d is the winner.

5.4 Vote Elicitation Protocol Using a Heuristic Function

In this section, we propose another possible way to select the best voter during elicitation process. Now, we are no longer interested to select the voter using MCTS; however, we propose a simpler heuristic to choose at each round the voter to whom it is most relevant to ask to complete his k-vote. In the following, first we describe the proposed heuristic then we present the elicitation process using this latter.

The main idea of the search heuristic is to identify two key candidates for whom we want to gather more information. Then, if there is at least a voter whose current ballot contains neither of these two candidates, query such a voter; otherwise, query a voter whose current ballot contains one of these two candidates. Algorithm 5.3 presents the proposed search method where given an incomplete profile R, the algorithm proceeds as follows:

1) Compute the minimal score of each candidate $S_{min}(x, R)$ for all $x \in A$.

2) Let d be one of the candidates with the highest minimal score, i.e. $d \in argmax S_{min}(x, R)$ for $x \in A$.

3) Compute the maximal score of each candidate $S_{max}(x, R)$ for all $x \in A, x \neq d$.

4) Let y be one of the candidates with the highest maximal score, i.e. $y \in argmax S_{max}(x, R)$ for $x \in A, x \neq d$.

5) Let *Voters* denote the set of selected voters who meet certain criteria (detailed below) by checking whether or not candidates d and y are present in their vote. These voters are chosen if one of these conditions is met (in the following order):

   Condition 1: voters who did not rank neither candidate d nor candidate y. Select voters with the minimal number of questions asked.

   Condition 2: voters who did not rank candidate d and the number of questions previously asked is less or equal than $\frac{m}{2}$. Select voters with the minimal number of questions asked.

   Condition 3: voters who did not rank candidate y and the number of questions previously asked is less or equal than $\frac{m}{2}$. Select voters with the minimal number of questions asked.

   Condition 4: if none of conditions 1-3 has been satisfied, we select voters with the minimal number of questions previously asked.

6) Finally, among all the voters available in *Voters*, we select one voter at random.

Algorithm 5.3: Search Algorithm

```
1  compute S_min(x, R) for all x ∈ A
2  d ← candidate with the highest minimal score
3  compute S_max(x, R) for all x ∈ A
4  y ← candidate with the highest maximal score such that y ≠ d
5  Voters ← select voters according to Condition 1.
6  if Voters is empty then
7  |   Voters ← select voters according to Condition 2.
8  else if Voters is empty then
9  |   Voters ← select voters according to Condition 3.
10 else
11 |   Voters ← select voters according to Condition 4.
12 i ← select a voter randomly among voters in Voters
13 return i
```

In the following, we use the proposed search method in an incremental elicitation protocol which we call P_{Search}. Indeed, P_{Search} proceeds just like P_{MCTS} by replacing (in line 4) the selection of the next voter using Algorithm 5.1 by the search Algorithm 5.3.

Protocol 5.4: P_{Search}

```
1 for each voter i ∈ N do
2 |   R ←Ask i to submit the name of her top candidate
3 repeat
4 |   i ← the selected voter returned by the Search Algorithm 5.3
5 |   ask i to send the name of her next preferred candidate
6 |   update R by adding the new vote of the selected voter i
7 until there exists a necessary winner c given R;
8 return c
```

Example 5.6. *Let us consider the complete profile of four voters having preferences over five candidates* $A = \{a, b, c, d, e\}$*:*

V_1	$e \succ d \succ b \succ c \succ a$
V_2	$d \succ b \succ c \succ a \succ e$
V_3	$c \succ a \succ e \succ d \succ b$
V_4	$d \succ b \succ e \succ c \succ a$

In the following, in each round, we determine the voter to ask to complete his vote using Algorithm 5.3. Tables below present the minimal and maximal scores of candidates in each round. (To alleviate notation we will simply write $S_{min}(x)$ and $S_{max}(x)$ for different completions of R.) Values in bold correspond to the highest minimal and maximal scores.

In the first round, given the most preferred candidate of each voter, there is no necessary winner. After computing the different scores, we choose a voter who did not rank candidates d and c: namely, voter 1. Given the new voter's preference and since no necessary winner exists, in the second round, we ask the voter who did not rank candidates d and e which corresponds to voter 3.

Round 1					
$R = \{e, d, c, d\}$					
x	a	b	c	d	e
$S_{min}(x)$	0	0	4	**8**	4
$S_{max}(x)$	12	12	**13**	-	13
Condition 1: Ask voter 1					

Round 2					
$R = \{e \succ d, d, c, d\}$					
x	a	b	c	d	e
$S_{min}(x)$	0	0	4	**11**	4
$S_{max}(x)$	11	11	12	-	**13**
Condition 1: Ask voter 3					

Since there is no necessary winner, we continue with a third round: we choose voters according to condition 2: we select the voter who did not rank candidate d: namely, voter 3. In the fourth round, we select voters who did not rank candidate c and with the minimal number of questions previously asked. Thus, we choose between voters 2 and 4. Assume we select voter 2. Note that we cannot consider condition 2 (voters who did not rank candidate d) because voter 3 was already asked to submit 3 candidates.

Round 3					
$R = \{e \succ d, d, c \succ a, d\}$					
x	a	b	c	d	e
$S_{min}(x)$	3	0	4	**11**	4
$S_{max}(x)$	11	10	**12**	-	12
Condition 2: Ask voter 3					

Round 4					
$R = \{e \succ d, d, c \succ a \succ e, d\}$					
x	a	b	c	d	e
$S_{min}(x)$	3	0	4	**11**	6
$S_{max}(x)$	11	9	**12**	-	12
Condition 3: Ask voter 2 or 4					

In the fifth round, we select the voter who did not rank candidate c and with the minimal number of questions previously asked: namely, voter 4. Given the new voter's preference, candidate d is a necessary winner.

Round 5					
$R = \{e \succ d, d \succ b, c \succ a \succ e, d\}$					
x	a	b	c	d	e
$S_{min}(x)$	3	3	4	**11**	6
$S_{max}(x)$	10	9	**11**	-	11
Condition 3: Ask voter 4					

$R = \{e \succ d, d \succ b, c \succ a \succ e, d \succ b\}$					
x	a	b	c	d	e
$S_{min}(x)$	3	6	4	11	6
$S_{max}(x)$	9	9	10	12	10
d is a necessary winner					

5.5 Empirical Results

In this section we evaluate the average communication cost of P_{MCTS} and P_{Search}. We discuss experiments using the Mallows ϕ model for different values of ϕ. Our objective is to determine the number of questions voters need to answer, on average, in order to determine the winner under Borda and Harmonic rules. Note that for Borda and Harmonic elections, in the worst case this number is $m - 1$, while for (say) plurality elections, this worst case number is 1. We refer to this worst case number as P_{Worst}.

We compare our results to the most common way of eliciting preferences iteratively using top-k ballots, where voters are allowed to cast their preferences incrementally starting by top-1 ballots, then top-2, etc., until there is sufficient information for knowing the winner. More precisely, in each round, given top-k ballots, we check whether or not a necessary winner exists. If no necessary winner has been found, all voters are asked to submit their next top-k preferred candidate. This elicitation process was used by Kalech *et al.* [44] which we refer to as P_{Rounds}. For instance, in Example 5.6, in order to determine the necessary winner, 2 rounds are needed, i.e. each voter submits her top-2 preferred candidate.

For MCTS's experiments, we fix the constant C to $0.3 \times (n \times m)$ (0.3 was obtained experimentally), 10.000 playouts were conducted for each move. All experiments were performed on a 2.6GHz Intel dual Core i5 processor with 4GB of RAM memory.

For each experiment, we draw 1000 random profiles. We present simulation results with $m = 7$ and $n = 10$ as we vary the value $\phi \in \{0.7, 0.8, 0.9, 1\}$. Figure 5.4 (resp. Figure 5.5) shows obtained results for Borda (resp. Harmonic) with P_{MCTS}, P_{Search} and P_{Rounds}, and we also show P_{Worst} for comparison. Results show that in practice, P_{MCTS}, P_{Search} and P_{Rounds} ask much fewer questions than the worst case (P_{Worst}). For Borda rule, best results are depicted with P_{MCTS} which is able to save up to 48% of communication cost followed by P_{Search} that reduces communication by 46%, finally, P_{Rounds} where the savings present only 31%, when $\phi = 0.7$. When $\phi = 1$, more information is needed from voters under P_{MCTS}, P_{Search} and P_{Rounds}; however,

the savings of P_{MCTS} and P_{Search} are still notable compared to P_{Rounds}: P_{MCTS} saves up to 33% of voters preferences against 29% for P_{Search} and 17% for P_{Rounds}.

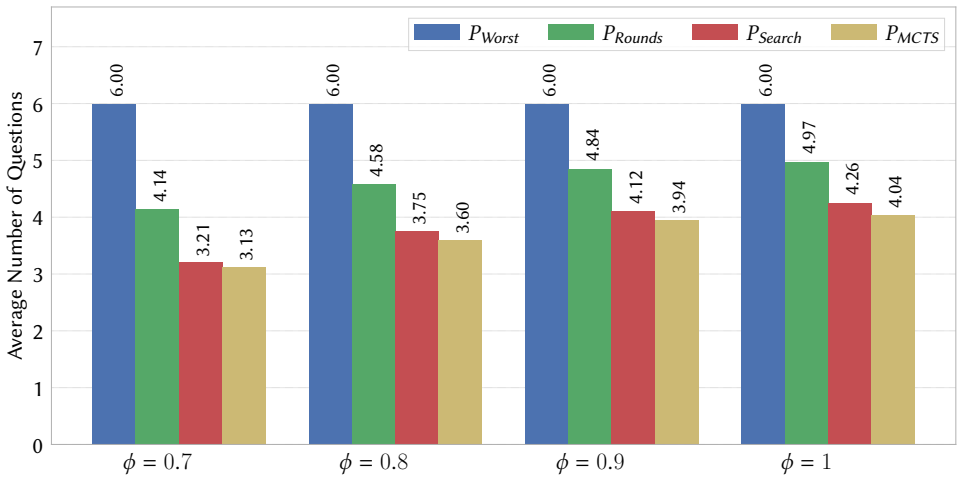


Figure 5.4: Average communication cost of Borda rule with Mallows ϕ model when: $m = 7$, $n = 10$ and $\phi \in \{.7, .8, .9, 1\}$.

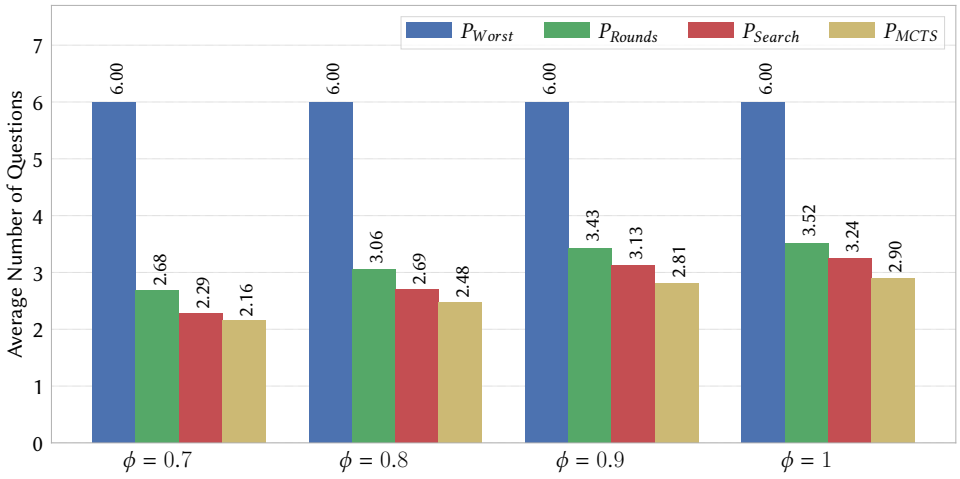


Figure 5.5: Average communication cost of Harmonic rule with Mallows ϕ model when: $m = 7$, $n = 10$ and $\phi \in \{.7, .8, .9, 1\}$.

A similar behavior occurs when considering Harmonic rule: P_{MCTS} outperforms P_{Search} and P_{Rounds}. From the results in Figure 5.5, we can notice that Harmonic requires less communication cost compared to Borda where the savings of Harmonic with different values of ϕ are very important. Indeed, under Harmonic rule when $\phi = 0.7$, P_{MCTS} saves up to 64% of voters' preferences against 61% for P_{Search} and 55% for P_{Rounds}. As we increase ϕ ($\phi \in \{0.8, 0.9\}$), the savings decrease by 5% (resp.

7%) when $\phi = 0.8$ and 10% (resp. 14%) when $\phi = 0.9$ under P_{MCTS} (resp. P_{Search}) compared to the results when $\phi = 0.7$. When $\phi = 1$, P_{MCTS} still outperforms the other protocols, i.e. P_{MCTS} needs only 48% of voters' preferences against 54% under P_{Search} and 58% under P_{Rounds}.

Figures 5.4 and 5.5 show that P_{MCTS} has lower communication cost under both Borda and Harmonic rules compared to the other protocols. For P_{MCTS}, we limit our experiments on small elections since evaluating the efficiency of MCTS with large elections is challenging: it is computationally expensive especially when the number of voters is high since it impacts the branching factor which prevents a deep look-ahead and hinders the construction of an effective evaluation function. Even though our conclusions are driven from experiments on small elections, we guess that the same behavior will be seen with large (real) elections.

In the next experiments, dedicated to P_{Search} and P_{Rounds}, we consider real data sets from *Preflib* presented in Table 4.3 (cf. Section 4.5.3.2) by excluding all partial ballots from the data and use only complete ones. Figure 5.6 (resp. Figure 5.7) shows the average number of questions voters asked by P_{Search}, P_{Rounds} and P_{Worst} on real data sets under Borda (resp. Harmonic) rule.

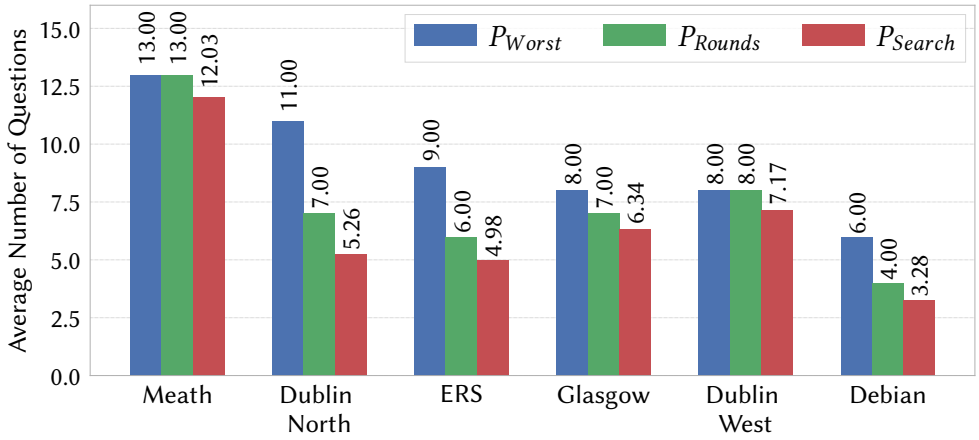


Figure 5.6: Average communication cost of Borda rule with real data sets under P_{Search}, P_{Rounds} and P_{Worst}.

Consistently with the above experiments, results show that P_{Search} and P_{Rounds} significantly perform better than P_{Worst} especially under Harmonic rule. For instance, for *Dublin North* data only 47% (resp. 35%) of the voters' preferences are needed for P_{Search} against 63% (resp. 45%) for P_{Rounds} under Borda (resp. Harmonic). Also, results suggest that under Borda rule, P_{Search} outperforms P_{Rounds} by asking 7–16% fewer questions; however, under Harmonic, the savings of P_{Search} are smaller compared to P_{Rounds}, i.e. P_{Search} asks 1–10% fewer questions than P_{Rounds}.

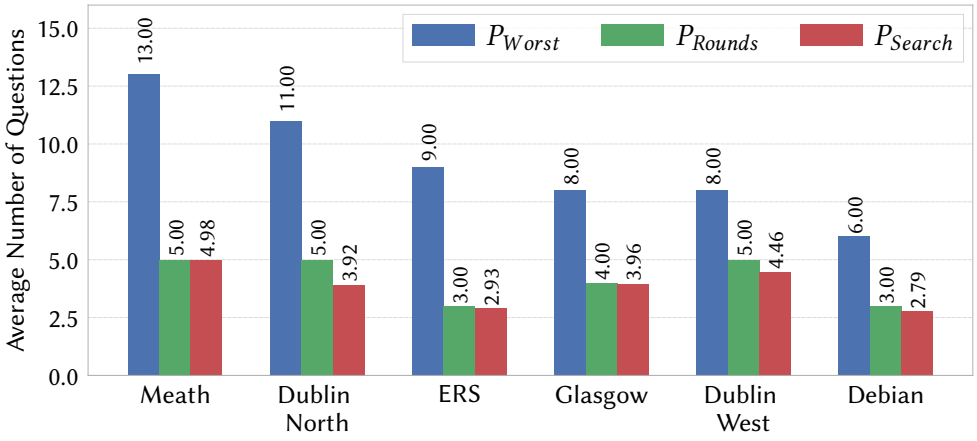


Figure 5.7: Average communication cost of Harmonic rule with real data sets under P_{Search}, P_{Rounds} and P_{Worst}.

Both protocols are much better than the worst-case analysis except for Meath and Dublin West with Borda: all voters preferences are needed to determine the winner when using P_{Rounds} (13 with Meath and 8 with Dublin West). This can be explained by the vote's distribution in these two data sets where the real winner does not appear in the first places (top-1, top 2, top 3) in a massive way unlike other candidates (for example in Dublin West the winner (b) appears 14% (resp. 17%, 16%) times in top-1 (resp. top-2, top-3) while the candidates (d) and (e) appear 22% and 25% in top-1, respectively. Indeed, this prevents the necessary winner to be determined in early rounds since his score will be very low compared to other candidates.

Results in Figure 5.7 show that the savings of Harmonic rule using P_{Search} are smaller compared to P_{Rounds}. We may wonder whether the communication cost of Harmonic could be improved if in P_{Search} instead of starting with the top-1 preferred candidate of each voter, we ask the voters for their top-2 preferred candidate at the first round. We denote such protocol by P^2_{Search}. Figure 5.8 reports on results of the average number of questions asked under P_{Search}, P^2_{Search}, P_{Rounds} and P_{Worst} with different real data sets. Depicted results show that P^2_{Search} asks 2–5% fewer questions than P_{Search}. Consequently, the savings of P^2_{Search} increase compared to P_{Rounds} by asking 3–12% fewer questions. Since Harmonic rule has a significantly lower communication cost compared to Borda, starting with top-2 ballots of each voter will allow the voting center to choose the "right" voter to query among the available ones which will accelerate the process of determining the necessary winner.

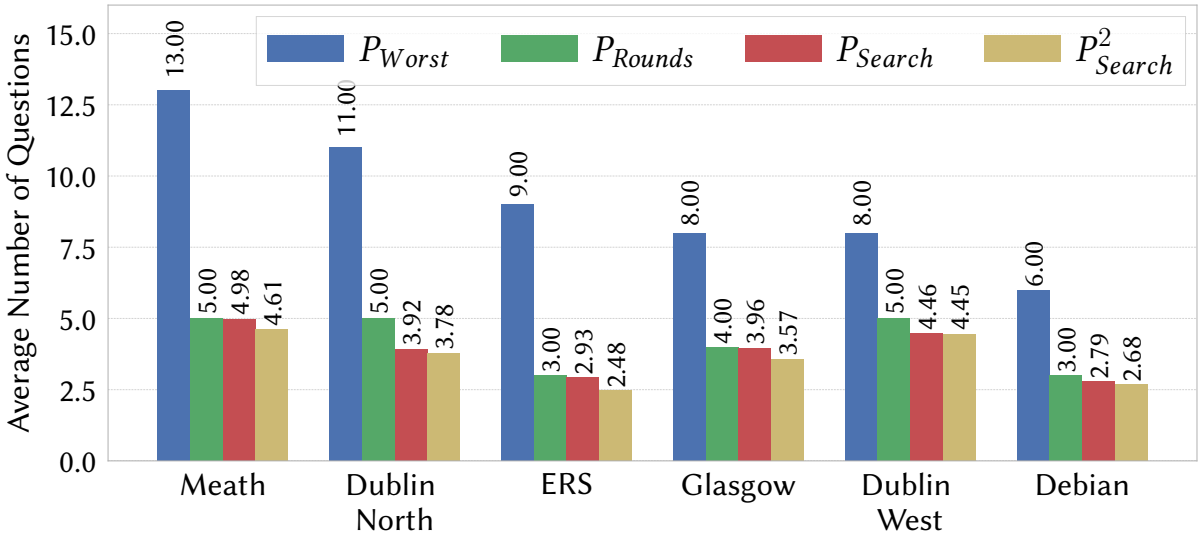


Figure 5.8: Average communication cost of Harmonic rule with real data sets under P_{Search}, P^2_{Search}, P_{Rounds} and P_{Worst}.

5.6 Conclusion

In this chapter we have proposed an interactive elicitation process using two heuristic methods to choose the voter to query in each round. The first heuristic uses the UCT selection function of MCTS technique while the second one considers a simple search heuristic able to select the voter (under certain conditions) for whom we want to reveal more information. Our empirical study shows that these two heuristics are efficient to reduce the number of questions asked to voters under Borda and Harmonic rules. From the results, both rules have low communication cost, either with P_{MCTS} (Section 5.3) or P_{Search} (Section 5.4). Although P_{MCTS} outperforms P_{Search}; however, under P_{MCTS} the computation and communication costs are not comparable. Indeed, with large elections the computation cost will be too heavy a burden to be supported by the voters, unless they may have to wait longer before being asked the next question.

The proposed elicitation protocols can be generalized to other voting rules. More precisely,

- P_{MCTS} can be adapted to any voting rule; however, it will nevertheless be necessary to have an algorithm to determine whether x is a necessary winner, which can sometimes be NP-complete (For instance the problem is coNP-complete for Copeland and ranked pairs, while it is polynomial for Maximin and Bucklin [75].)
- P_{Search} only works for rules based on score maximization, such as: positional

scoring rules (beyond Borda and Harmonic), Copeland and Maximin. Using this class of rules allows us to determine the minimum and maximum scores and identify the two key candidates for whom we want to gather more information.

Conclusion and perspectives

In this dissertation we have taken a comprehensive look at the amount of information needed to accurately decide the election's outcome given truncated ballots under different voting rules. We have considered successively two contexts and their associated questions:

First, we have assumed that the information from the voters is communicated in a single shot where each voter submits her k preferred candidates (for a fixed k). So using the *top-k* ballot, we were interested to approximate the true winner for different voting rules (Borda, Harmonic, Copeland, Maximin, ranked pairs and STV) by proposing "k-truncated approximations". The key question that we were interested in is: *How often will these approximations lead a mistake and output a different winner than we would have obtained with complete ballots?* To answer this question we have measured theoretically, for rules whose definition is based on score maximization (Borda, Harmonic, Copeland and Maximin), the score ratio between the true winner and the winner of the k-truncated rule. The obtained worst-case bounds are rather negative: very negative for Copeland (∞) and Maximin ($m-k$), less so for Borda ($\Theta(\frac{m}{k})$), and even less so for Harmonic ($\Theta(\frac{m}{k \log k})$). Experimental results on random Mallows model as well as real data suggest that a very small value of k work very well in practice which largely contrasts the obtained theoretical bounds. Best prediction is always obtained when considering Harmonic rule. We have also measured empirically the probability that the approximate rule selects the same winner as the original rule. Results confirm the conclusion obtained with the score ratio and suggest that small values of k work very well in practice. For instance with real data, eliciting $k = \frac{1}{4}m$ is a good predictor of the correct winner.

In the same context (i.e. *top-k* communicated in a single shot), we have paid a specific attention to STV rule which is of particular interest in voting since it is hard to manipulate and it enjoys a very important normative property: clone-proofness. In fact, several questions arise: *To which (quantitative) extent clone-proofness is preserved when replacing complete ballots by k-truncated ones ? What is the connection*

between STV_k (the STV version with k-truncated ballots) and the plurality with runoff rule? Knowing that deciding whether a candidate is possible winner for STV given partial profile is NP-complete; what is the complexity of this problem when considering "uniform truncated ballots" (fixed k)? Our conclusions about these questions are, respectively, as follows:

- For the question of clone-proofness, results using randomly generated profiles (on Mallows model by varying ϕ) show that resistance to cloning increases rapidly with k and decreases with ϕ. Also, it significantly increases with the number of voters. With real data, resistance to cloning increases rapidly with k, even more rapidly than with randomly generated profiles. The same result holds with the k-truncated version of the ranked pairs (RP) rule, namely RP_k.
- About the connection between STV_k and the plurality with runoff, we have studied the question empirically (on Mallows model by varying ϕ) and we conclude that STV_k for $k \approx \frac{2}{5}m$ (resp. $k = 1$) coincides with plurality with runoff when $\phi = 0.8$ (resp. $\phi = 0.7$). For $\phi = 1$, STV is very far from plurality with runoff.
- Results about complexity are very interesting, in fact we have proved that with $k = 1$ determining the possible winner for STV can be solved in polynomial time, but the problem remains NP-complete when $k \geqslant 2$ mainly due to the number of completions. While the problem is difficult in the worst case, we have showed that, in practice, there is a simple algorithm ($PW_{top\text{-}k}$ Algorithm 4.1) that is able to classify candidates into three classes: possible winners, necessary losers and candidates that we cannot decide on. Results on randomly generated profiles show that the proposed algorithm significantly reduces the set of candidates that we cannot decide on, thus, the number of completions. Results on Mallows model by varying ϕ show that when $\phi = 0.7$ only one candidate out of 7 is neither possible winner nor necessary loser given top-2 ballots. For $\phi = 0.8$, the set of candidates that we cannot decide on is present only with a probability of 2.7% and it decreases to 1.6% (resp. 1.08%) when $\phi = 0.9$ (resp. $\phi = 1$).

Second, we were interested in computing the real winner, rather than an approximate one, by allowing interactive communication. We have studied possible ways to minimize the required amount of communication. More precisely, the question here is: *What is the sufficient number of bits (on average) that must be communicated by the voters to the voting center so that the winner is determined ?* To answer this question, we first proposed exact methods for interactive vote elicitation to choose the voter to query in each round until the winner is known. Then we have explored innovative heuristic

methods, based on evaluation functions that will guide the choice of the next voter to query. We still focus on *top-k* queries where one voter will be asked at a time and, if she is queried for the k^{th} time, she will be asked to provide her k^{th} preferred candidate

- For exact methods, we have focused on STV rule. Building on a protocol proposed by Conitzer and Sandholm [25] which we called P_1, we have proposed a new protocol (P_2) able to reduce the amount of communication required in practice for STV. Then we have empirically studied the average communication complexity of P_1 and P_2, based on randomly generated profiles and on real-world election data. Our results show that STV has low communication cost in practice when using the proposed protocol P_2 by asking 5–10% fewer questions compared to P_1. When using P_2, most of the voters are asked to submit only their top preferred candidate. In P_1, it is more common to be asked to submit two preferences. Thus, the low communication cost of STV might be a reason to prefer it to other traditional voting rules, such as Borda.
- For heuristic methods, first we have adapted the Monte Carlo Tree Search (MCTS) approach to vote elicitation in order to select the most prominent voter to ask in each round based on its evaluation function Upper Confidence Bounds for Trees (UCT). We called such protocol P_{MCTS}. Second, we have proposed a simpler interactive protocol so called P_{Search} that uses an alternative heuristic method able to select the voter (under certain conditions) for whom we want to reveal more information. Results on randomly generated profiles and real data show that P_{MCTS} and P_{Search} are efficient to reduce the number of questions asked to voters under Borda and Harmonic rules. From the results, P_{MCTS} has low communication cost under both Borda and Harmonic rules.

Though we present a significant body of research in this dissertation, there are a number of interesting directions for future research:

- For the k-truncated approximations detailed in Chapter 3, we would like to consider approximations of a voting rule (here, *top-k* approximations) as a genuine rule, and to study their normative properties as doing by Caragiannis *et al.* [18].
- Complete our study on STV detailed in Chapter 4, by considering STV in another incomplete information context, namely *vote streams*, where voters arrive one at a time [10, 31]. The key practical question is to decide when we have enough information to eliminate one more candidate, so that the next voters will have less information to communicate.

- Complete our results on the use of heuristics on vote elicitation (in Chapter 5) by testing the performance of P_{MCTS} and P_{Search} with other voting rules (beyond Borda and Harmonic).
- Many interesting extensions can be considered when using MCTS (Chapter 5) in vote elicitation:
  - In the simulation step, instead of playing random moves from all possible legal moves, it would be interesting to use a predefined simulation strategy to choose moves from a subset of available moves. This subset may help to reach an end game faster and increase the chance to win. For instance, using the output of the search heuristic (of Section 5.4) to select the next voter when running the simulation, would be a very good starting point for testing the MCTS's performance with a predefined simulation strategy.
  - To reduce the computational cost of the simulation step, instead of simulating the game until the end, an evaluation cutoff strategy could be used to assess the game prematurely. An alternative is the *fixed-length cutoff strategy* [52] where the simulation stops after a fixed number of d moves have been executed in the playout.
  - To reduce the computational cost of MCTS in a vote elicitation context, MCTS has to be parallelized. Three possible ways exist to accomplish this task: (1) *leaf parallelization* [19] where independent simulated games are played in parallel for each leaf node, (2) *root parallelization* [19] where separate MCTS trees are created and when some number of iterations are finished, trees are combined to give a better statistic, and (3) *tree parallelization* [36] where one shared tree is created from which several simultaneous games are played.

References

[1] Kenneth J. Arrow. A difficulty in the concept of social welfare. *Journal of Political Economy*, 58(4):328–346, 1950.

[2] Kenneth J. Arrow, Amartya Sen, and Kotaro Suzumura. *Handbook of social choice and welfare*, volume 1. Gulf Professional Publishing, 2002.

[3] Kenneth J. Arrow, Amartya Sen, and Kotaro Suzumura. *Handbook of social choice and welfare*, volume 2. Elsevier, 2010.

[4] Manel Ayadi, Nahla Ben Amor, and Jérôme Lang. The communication burden of single transferable vote, in practice. In *Proceedings of the Algorithmic Game Theory - 11th International Symposium (SAGT), Beijing, China, September 11-14, 2018*, pages 251–255, 2018.

[5] Manel Ayadi, Nahla Ben Amor, and Jérôme Lang. The communication burden of single transferable vote, in practice. In *Proceedings of the 5th International Workshop on Computational Social Choice (COMSOC), Troy, NY, USA, June 25–27, 2018*, 2018.

[6] Manel Ayadi, Nahla Ben Amor, Jérôme Lang, and Dominik Peters. Single transferable vote: Incomplete knowledge and communication issues. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), Montreal, QC, Canada, May 13-17, 2019*, pages 1288–1296, 2019.

[7] Yoram Bachrach, Nadja Betzler, and Piotr Faliszewski. Probabilistic possible winner determination. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, Atlanta, Georgia, USA, July 11-15, 2010*, 2010.

[8] John J. Bartholdi and James B. Orlin. Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8(4):341–354, 1991.

[9] Dorothea Baumeister, Piotr Faliszewski, Jérôme Lang, and Jörg Rothe. Campaigns for lazy voters: truncated ballots. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Valencia, Spain, June 4-8, 2012 (3 Volumes)*, pages 577–584, 2012.

[10] Arnab Bhattacharyya and Palash Dey. Fishing out winners from vote streams. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:135, 2015.

[11] Duncan Black. On the rationale of group decision-making. *Journal of Political Economy*, 56(1):23–34, 1948.

[12] Craig Boutilier, Ioannis Caragiannis, Simi Haber, Tyler Lu, Ariel D. Procaccia, and Or Sheffet. Optimal social choice functions: A utilitarian view. *Artificial Intelligence*, 227:190–213, 2015.

[13] Craig Boutilier and Jeffrey S. Rosenschein. Incomplete information and communication in voting. In *Handbook of Computational Social Choice*, pages 223–258. 2016.

[14] Simina Brânzei, Ioannis Caragiannis, Jamie Morgenstern, and Ariel D. Procaccia. How bad is selfish voting? In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA.*, pages 138–144, 2013.

[15] Markus Brill and Felix A. Fischer. The price of neutrality for the ranked pairs method. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.*, pages 1299–1305, 2012.

[16] Dirk Briskorn, Gábor Erdélyi, and Christian Reger. Bribery under partial information. In *Proceedings of the 2nd Workshop on Exploring Beyond the Worst Case in Computational Social Choice (EXPLORE)*, 2015.

[17] Cameron B. Browne, Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[18] Ioannis Caragiannis, Christos Kaklamanis, Nikos Karanikolas, and Ariel D. Procaccia. Socially desirable approximations for dodgson's voting rule. *ACM Transactions on Algorithms (TALG)*, 10(2):6, 2014.

[19] Tristan Cazenave and Nicolas Jouandeau. On the parallelization of uct. In *Proceedings of the Computer Games Workshop*, pages 93–101. Citeseer, 2007.

[20] Guillaume M.J-B. Chaslot, Jahn-Takeshi Saito, Bruno Bouzy, Jos W.H.M. Uiterwijk, and H. Jaap Van Den Herik. Monte-Carlo strategies for computer go. In *Proceedings of the 18th BeNeLux Conference on Artificial Intelligence, Namur, Belgium*, pages 83–91, 2006.

[21] Guillaume M.J-B. Chaslot, Mark H.M. Winands, H. Jaap Van Den Herik, Jos W.H.M. Uiterwijk, and Bruno Bouzy. Progressive strategies for Monte-Carlo tree search. *New Mathematics and Natural Computation*, 4(03):343–357, 2008.

[22] Gal Cohensius, Omer Ben-Porat, Reshef Meir, and Ofra Amir. Efficient crowdsourcing via proxy voting. *CoRR*, abs/1806.06257, 2018.

[23] Vincent Conitzer, Matthew Rognlie, and Lirong Xia. Preference functions that score rankings and maximum likelihood estimation. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI), Pasadena, California, USA, July 11-17, 2009*, pages 109–115, 2009.

[24] Vincent Conitzer and Tuomas Sandholm. Vote elicitation: Complexity and strategy-proofness. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI), July 28 - August 1, 2002, Edmonton, Alberta, Canada.*, pages 392–397, 2002.

[25] Vincent Conitzer and Tuomas Sandholm. Communication complexity of common voting rules. In *Proceedings 6th ACM Conference on Electronic Commerce, EC-2005, Vancouver, BC, Canada, June 5-8, 2005*, pages 78–87, 2005.

[26] Vincent Conitzer, Tuomas Sandholm, and Jérôme Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM (JACM)*, 54(3):14, 2007.

[27] John Cullinan, Samuel K. Hsiao, and David Polett. A borda count for partially ordered ballots. *Social Choice and Welfare*, 42(4):913–926, 2014.

[28] Claude d'Aspremont and Louis Gevers. Social welfare functionals and interpersonal comparability. *Handbook of Social Choice and Welfare*, 1:459–541, 2002.

[29] Lihi Dery, Svetlana Obraztsova, Zinovi Rabinovich, and Meir Kalech. Lie on the fly: Strategic voting in an iterative preference elicitation process. *CoRR*, abs/1905.04933, 2019.

[30] Lihi Naamani Dery, Meir Kalech, Lior Rokach, and Bracha Shapira. Reaching a joint decision with minimal elicitation of voter preferences. *Information Sciences*, 278:466–487, 2014.

[31] Palash Dey, Nimrod Talmon, and Otniel van Handel. Proportional representation in vote streams. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 15–23, 2017.

[32] Jean-Paul Doignon, Aleksandar Pekeč, and Michel Regenwetter. The repeated insertion model for rankings: Missing link between two subset choice models. *Psychometrika*, 69(1):33–54, 2004.

[33] Edith Elkind, Martin Lackner, and Dominik Peters. Structured preferences. *Trends in Computational Social Choice, Chapter 10*, pages 187–207, 2017.

[34] Peter Emerson. The original borda count and partial voting. *Social Choice and Welfare*, 40(2):353–358, 2013.

[35] Peter J. Emerson. *The politics of consensus: For the resolution of conflict and reform of majority rule*. 1994.

[36] Markus Enzenberger and Martin Müller. A lock-free multithreaded monte-carlo tree search algorithm. In *Proceedings of the 12th Advances in Computer Games (ACG), Pamplona, Spain, May 11-13, 2009.*, pages 14–20, 2009.

[37] Yuval Filmus and Joel Oren. Efficient voting via the top-k elicitation scheme: a probabilistic approach. In *Proceedings of the fifteenth ACM Conference on Economics and Computation*, pages 295–312. ACM, 2014.

[38] Rupert Freeman, Markus Brill, and Vincent Conitzer. On the axiomatic characterization of runoff voting rules. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 675–681, 2014.

[39] Allan Gibbard. Manipulation of voting schemes: a general result. *Econometrica*, 41(4):587–601, 1973.

[40] Umberto Grandi, Andrea Loreggia, Francesca Rossi, and Vijay Saraswat. A borda count for collective sentiment analysis. *Annals of Mathematics and Artificial Intelligence*, 77(3-4):281–302, August 2016.

[41] G.-T. Guilbaud. Les théories de l'intérêt général et le problème logique de l'agrégation. *Economie appliquée*, 5(4):501–584, 1952.

[42] Noam Hazon, Yonatan Aumann, Sarit Kraus, and Michael Wooldridge. On the evaluation of election outcomes under uncertainty. *Artificial Intelligence*, 189:1–18, 2012.

[43] Dongling Huang and Lan Luo. Consumer preference elicitation of complex products using fuzzy support vector machine active learning. *Marketing Science*, 35(3):445–464, 2016.

[44] Meir Kalech, Sarit Kraus, Gal A. Kaminka, and Claudia V. Goldman. Practical voting rules with partial information. *Autonomous Agents and Multi-Agent Systems*, 22(1):151–182, 2011.

[45] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European Conference on Machine Learning*, pages 282–293. Springer, 2006.

[46] Kathrin Konczak and Jérôme Lang. Voting procedures with incomplete preferences. In *Proceedings of the Multidisciplinary IJCAI Workshop on Advances in Preference Handling*, volume 20, pages 124–129. Citeseer, 2005.

[47] Kiyoshi Kuga and Hiroaki Nagatani. Voter antagonism and the paradox of voting. *Econometrica: Journal of the Econometric Society*, 42:1045–1067, 1974.

[48] Martin Lackner. Incomplete preferences in single-peaked electorates. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 742–748, 2014.

[49] Jérôme Lang, Maria Silvia Pini, Francesca Rossi, Domenico Salvagnin, Kristen Brent Venable, and Toby Walsh. Winner determination in voting trees with incomplete preferences and weighted votes. *Autonomous Agents and Multi-Agent Systems*, 25(1):130–157, 2012.

[50] Jean-François Laslier. Heuristic voting under the Alternative Vote: the efficiency of 'sour grapes' behavior. *Homo Oeconomicus*, 33(1):57–76, 2016.

[51] Benedikt Loepp, Tim Hussein, and Jüergen Ziegler. Choice-based preference elicitation for collaborative filtering recommender systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3085–3094. ACM, 2014.

[52] Richard J Lorentz. Amazons discover monte-carlo. In *Proceedings of the International Conference on Computers and Games*, pages 13–24. Springer, 2008.

[53] Tyler Lu and Craig Boutilier. Learning mallows models with pairwise preferences. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 145–152, 2011.

[54] Tyler Lu and Craig Boutilier. Robust approximation and incremental elicitation in voting protocols. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 287–293, 2011.

[55] Tyler Lu and Craig Boutilier. Vote elicitation with probabilistic preference models: Empirical estimation and cost tradeoffs. In *Proceedings of the International Conference on Algorithmic Decision Theory (ADT), Piscataway, NJ, USA, October 26-28, 2011.*, pages 135–149, 2011.

[56] Colin L. Mallows. Non-null ranking models. *Biometrika*, 44:114–130, 1957.

[57] Nicholas Mattei. Empirical evaluation of voting rules with strictly ordered preference data. In *Proceedings of the International Conference on Algorithmic Decision Theory (ADT)*, pages 165–177. Springer, 2011.

[58] Nicholas Mattei and Toby Walsh. Preflib: A library for preferences http://www.preflib. org. In *Proceedings of the International Conference on Algorithmic Decision Theory (ADT)*, pages 259–270. Springer, 2013.

[59] Nina Narodytska and Toby Walsh. The computational impact of partial votes on strategic voting. In *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI), 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems PAIS*, pages 657–662, 2014.

[60] Jean Antoine Nicolas. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. Par M. le marquis de Condorcet....* de l'Imprimerie Royale, 1785.

[61] Joel Oren, Yuval Filmus, and Craig Boutilier. Efficient vote elicitation under candidate uncertainty. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI), Beijing, China, August 3-9, 2013*, pages 309–316, 2013.

[62] Maria Silvia Pini, Francesca Rossi, Kristen Brent Venable, and Toby Walsh. Incompleteness and incomparability in preference aggregation: Complexity results. *Artificial Intelligence*, 175(7-8):1272–1289, 2011.

[63] Michel Regenwetter, Bernard Grofman, A. A. J. Marley, and Ilia Tsetlin. *Behavioral Social Choice - Probabilistic Models, Statistical Inference, and Applications.* Cambridge University Press, 2006.

[64] Mark Allen Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10(2):187–217, 1975.

[65] Markus Schulze. A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method. *Social Choice and Welfare*, 36(2):267–303, 2011.

[66] Travis C. Service and Julie A. Adams. Communication complexity of approximating voting rules. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Valencia, Spain, June 4-8, 2012 (3 Volumes)*, pages 593–602, 2012.

[67] Piotr Skowron, Piotr Faliszewski, and Arkadii Slinko. Achieving fully proportional representation: Approximability results. *Artificial Intelligence*, 222:67–103, 2015.

[68] T. N. Tideman. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3):185–206, September 1987.

[69] Toby Walsh. Uncertainty in preference elicitation and aggregation. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, volume 7, pages 3–8, 2007.

[70] Toby Walsh. Complexity of terminating preference elicitation. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Estoril, Portugal, May 12-16, 2008, Volume 2*, pages 967–974, 2008.

[71] Toby Walsh. An empirical study of the manipulability of single transferable voting. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI), Lisbon, Portugal, August 16-20, 2010*, pages 257–262, 2010.

[72] Toby Walsh. Generating single peaked votes. *CoRR*, abs/1503.02766, 2015.

[73] Jun Wang, Sujoy Sikdar, Tyler Shepherd, Zhibing Zhao, Chunheng Jiang, and Lirong Xia. Practical algorithms for STV and ranked pairs with parallel universes tiebreaking. *CoRR*, abs/1805.06992, 2018.

[74] Marieke G.M. Weernink, Sarah I. M. Janus, Janine A. Van Til, Dennis W. Raisch, Jeannette G. Van Manen, and Maarten J. IJzerman. A systematic review to identify the use of preference elicitation methods in healthcare decision making. *Pharmaceutical Medicine*, 28(4):175–185, 2014.

[75] Lirong Xia and Vincent Conitzer. Determining possible and necessary winners under common voting rules given partial orders. *Journal of Artificial Intelligence Research*, 41:25–67, 2011.

[76] H Peyton Young. Social choice scoring functions. *SIAM Journal on Applied Mathematics*, 28(4):824–838, 1975.

[77] Zhibing Zhao, Haoming Li, Junming Wang, Jeffrey O. Kephart, Nicholas Mattei, Hui Su, and Lirong Xia. A cost-effective framework for preference elicitation and aggregation. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI), Monterey, California, USA, August 6-10, 2018*, pages 446–456, 2018.

RÉSUMÉ

Les règles de vote classiques supposent que les bulletins de vote des électeurs sont des ordres de préférence complets sur les candidats. Cependant, lorsque le nombre de candidats est suffisamment élevé, il est trop coûteux de demander aux électeurs de classer tous les candidats. Il y a donc un compromis à faire entre l'efficacité d'une méthode d'agrégation des préférences et la charge de communication qu'elle fait peser sur les électeurs.

Dans cette thèse, nous abordons ce problème en suggérant de demander aux électeurs de ne classer que leurs k candidats préférés (où k peut varier selon les électeurs et/ou au cours du processus). On dit que de tels votes sont k-tronqués. Nous étudions la quantité d'information nécessaire pour déterminer le résultat de l'élection (de manière exacte ou approchée) à partir de bulletins tronqués selon différentes règles de vote et nous proposons et analysons différentes méthodes permettant un compromis entre précision du résultat et quantité de communication requise; certaines ne requièrent qu'une seule phase de communication, alors que d'autres sont dynamiques.

MOTS CLÉS

Choix social computationnel, Approximation, Bulletins tronqués, Protocole de communication

ABSTRACT

Classical voting rules assume that voters' ballots are complete preference orders over candidates. However, when the number of candidates is large enough, it is too costly to ask the voters to rank all candidates. There is therefore a trade-off between the efficiency of an aggregation method and the communication burden it places on voters.

In this thesis, we address this problem by suggesting to ask voters to report only their k preferred candidates (where k may vary depending on the voters and/or during the process). The obtained ballots are then said to be k-truncated. We study the amount of information needed to determine the outcome of the election (exact or approximate) from truncated ballots with respect to different voting rules and we propose and analyze different methods allowing a compromise between the accuracy of the result and the amount of communication required; some require only one round of communication, while others are interactive.

KEYWORDS

Computational social choice, Approximation, Truncated ballots, Communication protocol