Complexity of the Winner Determination Problem in Judgment Aggregation: Kemeny, Slater, Tideman, Young

Ulle Endriss
ILLC, University of Amsterdam
ulle.endriss@uva.nl

Ronald de Haan
Technische Universität Wien
dehaan@kr.tuwien.ac.at

ABSTRACT

Judgment aggregation is a collective decision making framework where the opinions of a group of agents is combined into a collective opinion. This can be done using many different judgment aggregation procedures. We study the computational complexity of computing the group opinion for several of the most prominent judgment aggregation procedures. In particular, we show that the complexity of this winner determination problem for analogues of the Kemeny rule, the Slater rule and the Young rule lies at the Θ_2^p-level of the Polynomial Hierarchy (PH). Moreover, we show that the problem has a complexity at the Δ_2^p-level of the PH for the analogue of Tideman's procedure with a fixed tie-breaking rule, and at the Σ_2^p-level of the PH for the analogue of Tideman's procedure without a fixed tie-breaking rule.

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms

Theory

Keywords

Judgment Aggregation; Winner Determination; Complexity Theory; Bounded Query Complexity

1. INTRODUCTION

Collective decision making is central in the area of multiagent systems [6]. Judgment aggregation is a collective decision making framework that can be used for many applications [12, 17, 23]. In judgment aggregation, the goal is to combine the opinions of a group of individuals (or agents) on a set of propositions into a collective opinion reflecting the views of the group as a whole. As such, judgment aggregation generalizes the setting of preference aggregation, where the opinions are restricted to preferences over a given domain of alternatives [9]. There are many ways (or procedures) to combine the individual opinions, and choosing between such procedures involves deciding between various desirable properties that such procedures can have. Some of the most salient desiderata for these procedures are that they are consistent (that is, the resulting group opinion is a tenable position) and complete (that is, for each proposition, the resulting group opinion takes a clear position).

Another important property of judgment aggregation procedures concerns their computational complexity, i.e., the amount of time it takes to compute the group opinion. In this paper, expanding on previous work [13, 21], we study the computational complexity of the winner determination problem for several of the most prominent judgment aggregation procedures that are complete and consistent. In particular, we study the judgment aggregation analogues of the Kemeny rule, the Slater rule and the Young rule, as well as two variants of Tideman's Ranked-Pairs rule (whose judgment aggregation analogue we call the Ranked-Agenda rule), all familiar from voting theory and preference aggregation [5].

Concretely, we study two different computational problems for the various procedures, one decision problem and one search problem. We consider the problem of deciding whether there exists an outcome that is deemed acceptable by the judgment aggregation procedure and that satisfies a number of additional requirements. In addition, we consider the computational task of producing one such acceptable outcome, if it exists. We argue that this latter formalization of the problem more adequately models the relevant computational properties of the various procedures. We show that the complexity (of both the decision and the search problem) for the Kemeny, Slater and Young rules lies at the Θ_2^p-level of the Polynomial Hierarchy (PH). Interestingly, these rules have exactly the same complexity in the setting of preference aggregation. In addition, we show that the complexity of Tideman's Ranked-Agenda procedure lies at the Δ_2^p-level of the PH, in case of a fixed tie-breaking rule, and lies at the Σ_2^p-level of the PH in case ties can be broken in arbitrary ways. It is interesting that both variants of the Ranked-Agenda rule are of higher computational complexity than the other three rules we consider in the setting of judgment aggregation, whereas in the case of preference aggregation both variants are of lower complexity than these rules. An overview of the complexity results that we obtain in this paper can be found in Table 1.

Understanding the complexity of the winner determination problem is a first step to making these important, but generally highly intractable, procedures amenable to practical use in multiagent systems and related fields requiring the consistent aggregation of information coming from several

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

autonomous agents, or more generally, independent sources of information. Determining the exact location of these problems in the PH is very useful for determining what algorithmic approaches are best-suited to solve the problems in practice. For problems at the Θ_2^p-level and at the Δ_2^p-level, for instance, the method of iterative SAT solving could be used, which is generally more efficient than solving methods for problems at the Σ_2^p-level.

Related Work.
The computational complexity of the winner determination problem has been studied before, by Endriss et al. [13] for several procedures including the Kemeny rule, and by Lang and Slavkovik [21] for several procedures, including the Slater rule, the Kemeny rule and the variant of the Ranked-Agenda procedure without a fixed tie-breaking rule. However, those previous works consider different formalizations of the computational task of winner determination. First, they study only decision problems (and no search problems) [13, 21]. Second, the problems studied by Lang and Slavkovik involve checking whether all outcomes satisfy a certain property, rather than some outcome [21]. The computational complexity of various other computational tasks in judgment aggregation has also been studied, including problems related to manipulation, bribery and control [1, 3, 4, 8, 13].

Structure of the Paper.
We begin, in Section 2, with reviewing notions from logic, judgment aggregation and complexity theory. Then, in Section 3, we formally define the computational problems that we use to capture the task of winner determination, and we analyze their computational complexity for the various judgment aggregation procedures. Finally, in Section 4, we conclude, and suggest directions for further research.

2. PRELIMINARIES
In this section we review relevant material on logic, judgment aggregation and complexity theory.

2.1 Propositional Logic
A *literal* is a propositional variable x or a negated variable $\neg x$. For literals $l \in \{x, \neg x\}$, we let $\text{Var}(l) = x$ denote the variable occurring in l. A *clause* is a finite set of literals, not containing a complementary pair x, $\neg x$, and is interpreted as the disjunction of these literals. We let $\bot$ denote the empty clause. A formula in *conjunctive normal form (CNF)* is a finite set of clauses, interpreted as the conjunction of these clauses. We define the *size* $||\varphi||$ of a CNF formula φ to be $\sum_{c \in \varphi} |c|$; the number of clauses of φ is denoted by $|\varphi|$. For a CNF formula φ, the set $\text{Var}(\varphi)$ denotes the set of all variables x such that some clause of φ contains x or $\neg x$. We use the standard notion of *(truth) assignments* $\alpha : \text{Var}(\varphi) \rightarrow \{0, 1\}$ for Boolean formulas and *truth* of a formula under such an assignment. We let SAT denote the problem of deciding whether a given propositional formula is satisfiable. For every propositional formula φ, we let $\sim\varphi$ denote the *complement* of φ, i.e., $\sim\varphi = \neg\varphi$ if φ is not of the form $\neg\psi$, and $\sim\varphi = \psi$ if φ is of the form $\neg\psi$.

2.2 Judgment Aggregation
An *agenda* is a finite nonempty set Φ of propositional formulas that does not contain any doubly-negated formulas and that is closed under complementation. Moreover, if $\Phi = \{\varphi_1, \dots, \varphi_m, \neg\varphi_1, \dots, \neg\varphi_m\}$ is an agenda, then we let $[\Phi] = \{\varphi_1, \dots, \varphi_m\}$ denote the *preagenda* associated with the agenda Φ. A *judgment set* J for an agenda Φ is a subset $J \subseteq \Phi$. We call a judgment set J *complete* if either $\varphi \in J$ or $\sim\varphi \in J$, for all $\varphi \in \Phi$; we call it *complement-free* if for all $\varphi \in \Phi$ it is not the case that both φ and $\sim\varphi$ are in J; and we call it *consistent* if there exists a truth assignment that makes all formulas in J true.

In addition, we associate with each agenda Φ an integrity constraint Γ, that can be used to explicitly represent logical dependencies between agenda issues. Integrity constraints for agendas have been considered in previous literature [10, 21], and the notion bears resemblance to the framework of binary aggregation with integrity constraints [15, 16]. Such an integrity constraint Γ consists of a single propositional formula. In the remainder of the paper, if no integrity constraint is specified, we implicitly assume that $\Gamma = \top$. We say that a judgment set J is Γ-*consistent* if there exists a truth assignment that simultaneously makes all formulas in J and Γ true. Let $\mathcal{J}(\Phi, \Gamma)$ denote the set of all complete and Γ-consistent subsets of Φ.

Let $\mathcal{N}$ be a finite set of *individuals* (or *agents*). A *judgment aggregation procedure (or rule)* for the agenda Φ and the set $\mathcal{N}$ of individuals is a function F that takes as input a profile $\boldsymbol{J} \in \mathcal{J}(\Phi, \Gamma)^n$, consisting of a sequence $(J_1, \dots, J_n)$ of $n = |\mathcal{N}|$ complete and Γ-consistent judgment sets, and that produces a non-empty set of non-empty judgment sets, i.e., it produces an element in $\mathcal{P}(\mathcal{P}(\Phi)\backslash\{\emptyset\})\backslash\{\emptyset\}$. We call a judgment aggregation procedure F *resolute* if for any profile $\boldsymbol{J}$ it returns a singleton, i.e., $|F(\boldsymbol{J})| = 1$; otherwise, we call F *irresolute*. An example of a resolute judgment aggregation procedure is the *strict majority rule* F^{maj}, where $\varphi \in F^{\text{maj}}(\boldsymbol{J})$ if and only if φ occurs in the strict majority of judgment sets in $\boldsymbol{J}$, for all $\varphi \in \Phi$. We call F *complete*, *complement-free* and *(Γ-)consistent*, if J is complete, complement-free and (Γ-)consistent, respectively, for every $\boldsymbol{J} \in \mathcal{J}(\Phi, \Gamma)^n$ and every $J \in F(\boldsymbol{J})$.

As an example, consider the following profile for a judgment aggregation problem with preagenda $[\Phi] = \{p, q, p \vee q\}$, integrity constraint $\Gamma = \neg(p \wedge q)$, and three individuals:

	p	q	$p \vee q$
individual 1	no	yes	yes
individual 2	yes	no	yes
individual 3	no	no	no
majority	no	no	yes

Thus, respecting majorities leads to an inconsistent outcome for this profile. This is an instance of the well-known *discursive dilemma* [22]. In this paper, we will consider several judgment aggregation procedures that resolve this dilemma by always picking from the set of all consistent judgment sets. There currently is no consistent naming convention for these aggregation procedures in the literature. Here, to identify the procedures that we consider, we use names from well-known voting rules that most closely resemble them [5]. The procedures that we consider have been studied as judgment aggregation procedures before. We consider procedures that resemble the Kemeny rule [24, 25], the Slater rule [24], the Young rule [20], and the Ranked-Agenda rule [20, 26].

The former three aggregation procedures are based on a notion of score. For $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$, we define Score_F as follows. Here J denotes a single consistent and complete judgment set, $\boldsymbol{J} \in \mathcal{J}(\Phi, \Gamma)^n$ denotes

F	WinDet(F)	FWinDet(F)	location in the PH
Kemeny	$\mathrm{P}^{\mathrm{NP}}[\log]$-complete	$\mathrm{FP}^{\mathrm{NP}}[\log,\mathrm{wit}]$-complete	Θ_2^{p}-level
Slater	$\mathrm{P}^{\mathrm{NP}}[\log]$-complete	$\mathrm{FP}^{\mathrm{NP}}[\log,\mathrm{wit}]$-complete	Θ_2^{p}-level
Young	$\mathrm{P}^{\mathrm{NP}}[\log]$-complete	$\mathrm{FP}^{\mathrm{NP}}[\log,\mathrm{wit}]$-complete	Θ_2^{p}-level
Tideman (fixed tie-breaking)	P^{NP}-complete	$\mathrm{FP}^{\mathrm{NP}}$-complete	Δ_2^{p}-level
Tideman	Σ_2^{p}-complete	$\mathrm{F}\Sigma_2^{\mathrm{p}}$-complete	Σ_2^{p}-level

Table 1: Overview of complexity results.

a profile, and $d(J, J')$ denotes the Hamming distance between two consistent and complete judgment sets J, J'. We let $\mathrm{Score}_{\mathrm{Kemeny}}(J, \boldsymbol{J}) = \sum_{1 \leq i \leq n} d(J, J_i)$. Moreover, we let $\mathrm{Score}_{\mathrm{Slater}}(J, \boldsymbol{J}) = d(J, F^{\mathrm{maj}}(\boldsymbol{J}))$, where $F^{\mathrm{maj}}(\boldsymbol{J})$ denotes the (possibly inconsistent) majority outcome of a profile $\boldsymbol{J}$. Finally, we let $\mathrm{Score}_{\mathrm{Young}}(J, \boldsymbol{J}) = \min\{\, k :$ there exists a subprofile $\boldsymbol{J}'$ of $\boldsymbol{J}$ containing $n-k$ judgment sets such that $J = F^{\mathrm{maj}}(\boldsymbol{J}')\,\}$. Using these scores, we can define the aggregation procedures $\mathrm{Winner}_{\mathrm{F},\Phi,\Gamma}$, for $\mathrm{F} \in \{$Kemeny, Slater, Young$\}$, as follows. We say that a Γ-consistent and complete judgment set J^* is in $\mathrm{Winner}_{\mathrm{F},\Phi,\Gamma}(\boldsymbol{J})$ if and only if there is no Γ-consistent and complete judgment set J such that $\mathrm{Score}_{\mathrm{F}}(J, \boldsymbol{J}) < \mathrm{Score}_{\mathrm{F}}(J^*, \boldsymbol{J})$.

Intuitively, the *Kemeny* rule selects those complete and consistent judgment sets that minimize the cumulative Hamming distance to the judgment sets in the profile. The *Slater* rule selects those complete and consistent judgment sets that minimize the Hamming distance to the majority outcome. The *Young* rule selects those complete and consistent judgment sets that are the majority outcome of a subprofile of maximal size with a consistent majority outcome.

Next, we consider the *Ranked-Agenda* aggregation procedure. Given a profile $\boldsymbol{J} = (J_1, \dots, J_n) \in \mathcal{J}(\Phi, \Gamma)^n$ and a formula $\varphi \in \Phi$, we let the *majority strength* $\mathrm{ms}(\boldsymbol{J}, \varphi)$ be the number of i's such that $\varphi \in J_i$, i.e., $\mathrm{ms}(\boldsymbol{J}, \varphi) = |\{\, 1 \leq i \leq n : \varphi \in J_i \,\}|$. We define the partial order $\leq_{\boldsymbol{J}} \subseteq \Phi \times \Phi$ as follows. Let $\varphi, \varphi' \in \Phi$ be formulas. If $\mathrm{ms}(\boldsymbol{J}, \varphi) < \mathrm{ms}(\boldsymbol{J}, \varphi')$, we let $\varphi \leq_{\boldsymbol{J}} \varphi'$; and if $\mathrm{ms}(\boldsymbol{J}, \varphi) = \mathrm{ms}(\boldsymbol{J}, \varphi')$, we let $\varphi \not\leq_{\boldsymbol{J}} \varphi'$. We say that a total order $<_{\boldsymbol{J}}$ is a *Ranked-Agenda order (for $\boldsymbol{J}$)* if it extends $\leq_{\boldsymbol{J}}$. Given a Ranked-Agenda order $<_{\boldsymbol{J}}$, we define the judgment set $\mathrm{RA}(<_{\boldsymbol{J}}, \Phi, \Gamma)$ as follows. Let $\Phi = \{\varphi_1, \dots, \varphi_m\}$ and assume that $\varphi_1 >_{\boldsymbol{J}} \dots >_{\boldsymbol{J}} \varphi_{2m}$. We let $J_0 = \emptyset$. For each $1 \leq i \leq 2m$, we let $J_i = J_{i-1} \cup \{\varphi_i\}$ if $J_{i-1} \cup \{\varphi_i\}$ is Γ-consistent; otherwise, we let $J_i = J_{i-1}$. By definition, each J_i is Γ-consistent. (Note that we could equivalently let $J_i = J_{i-1} \cup \{\neg\varphi_i\}$ if $J_{i-1} \cup \{\varphi_i\}$ is not Γ-consistent.) We then let $\mathrm{RA}(<_{\boldsymbol{J}}, \Phi, \Gamma) = J_{2m}$. Given any total order $<^A \subseteq \Phi \times \Phi$, and a partial order $\leq_{\boldsymbol{J}}$, we define the Ranked-Agenda order $<^A_{\boldsymbol{J}}$ to be the unique total order such that: (1) $\varphi <^A_{\boldsymbol{J}} \varphi'$ whenever $\varphi \leq_{\boldsymbol{J}} \varphi'$ and $\varphi' \not\leq_{\boldsymbol{J}} \varphi$; and (2) $\varphi <^A_{\boldsymbol{J}} \varphi'$ if and only if $\varphi <^A \varphi'$, whenever $\varphi =_{\boldsymbol{J}} \varphi'$. We now say that a Γ-consistent and complete judgment set $J^* \in \mathcal{J}(\Phi, \Gamma)$ is a *Ranked-Agenda winner*, denoted $J^* \in \mathrm{Winner}_{\mathrm{RA},\Phi,\Gamma}(\boldsymbol{J})$ if there is some Ranked-Agenda order $<_{\boldsymbol{J}}$ for $\boldsymbol{J}$ such that $J^* = \mathrm{RA}(<_{\boldsymbol{J}}, \Phi, \Gamma)$. We say that a tie-breaking rule A is defined by a total order $<^A \subseteq \Phi \times \Phi$ for each agenda Φ. Moreover, we say that a Γ-consistent and complete judgment set $J^* \in \mathcal{J}(\Phi, \Gamma)$ is a *Ranked-Agenda winner for the fixed tie-breaking rule A*, denoted $J^* \in \mathrm{Winner}_{\mathrm{RA}_A,\Phi,\Gamma}(\boldsymbol{J})$ if $J^* = \mathrm{RA}(<^A_{\boldsymbol{J}}, \Phi, \Gamma)$. In voting, it is pairs (of alternatives) rather than propositions that are being ranked, which is why the corresponding voting rule is called the Ranked-Pairs rule.

2.3 Complexity Theory

Readers familiar with seach problems and bounded query complexity may skip this section.

Search Problems.

In this paper, we will assume knowledge of to the well-known complexity classes P and NP, consisting of decision problems. In addition, we will consider search problems. Let Σ be an alphabet. A *search problem* is a binary relation R over strings in Σ^*. For any input string $x \in \Sigma^*$, we let $R(x) = \{\, y \in \Sigma^* : (x, y) \in R \,\}$ denote the set of *solutions* for x. We say that a Turing machine T *solves* R if on input $x \in \Sigma^*$ the following holds: if there exists at least one y such that $(x, y) \in R$, then T accepts x and outputs some y such that $(x, y) \in R$; otherwise, T rejects x. With any search problem R we associate a decision problem S_R, defined by $S_R = \{\, x \in \Sigma^* :$ there exists some $y \in \Sigma^*$ such that $(x, y) \in R \,\}$. We will use the following notion of reductions for search problems. A *polynomial-time Levin reduction* from one search problem R_1 to another search problem R_2 is a pair of polynomial-time computable functions (g_1, g_2) such that (1) the function g_1 is a many-one reduction from S_{R_1} to S_{R_2}, i.e., for every $x \in \Sigma^*$ it holds that $x \in S_{R_1}$ if and only if $g_1(x) \in S_{R_2}$; and (2) for every string $x \in S_{R_1}$ and every solution $y \in R_2(g_1(x))$ it holds that $(x, g_2(x, y)) \in R_1$. For more details, we refer to textbooks on the topic [14].

Complexity Classes.

The complexity class FP consists of those search problems that can be computed by a polynomial-time deterministic Turing machine, and the class FNP consists of those search problems that can be computed by a polynomial-time non-deterministic Turing machine.

Moreover, we will use complexity classes that are based on Turing machines that have access to an oracle. Let C be a complexity class with decision problems. A Turing machine T with access to a *yes-no C oracle* is a Turing machine with a dedicated *oracle tape* and dedicated states q_{oracle}, q_{yes} and q_{no}. Whenever T is in the state q_{oracle}, it does not proceed according to the transition relation, but instead it transitions into the state q_{yes} if the oracle tape contains a string x that is a yes-instance for the problem C, i.e., if $x \in C$, and it transitions into the state q_{no} if $x \notin C$. Let C be a complexity class with search problems. Similarly, a Turing machine with access to a *witness C oracle* has a dedicated oracle tape and dedicated states q_{oracle}, q_{yes} and q_{no}. Also, whenever T is in the state q_{oracle} it transitions into the state q_{yes} if the oracle tape contains a string x such that there exists some y such that $C(x, y)$, and in addition the contents of the oracle tape are replaced by (the encoding of) such an y; it transitions into the state q_{no} if there exists no y such that $C(x, y)$. Such transitions are called *oracle queries*.

We point out that the notion of algorithms that have access to witness FNP oracles accurately models algorithms that can call a SAT solver (modulo the running time of the SAT solver), as SAT solvers also return a satisfying assignment if it exists.

In this paper, we will consider the following complexity classes that are based on oracle machines. The class $\mathrm{P}^{\mathrm{NP}}[\log]$ consists of all decision problems that can be decided by a deterministic polynomial-time Turing machine that has access to a yes-no NP oracle, and on any input of length n queries the oracle at most $O(\log n)$ many times. This class coincides with the class $\mathrm{P}^{\mathrm{NP}}_{||}$, and is also known as Θ^{p}_2.

The class $\mathrm{FP}^{\mathrm{NP}}[\log,\mathrm{wit}]$ consists of all search problems that can be solved by a deterministic polynomial-time Turing machine that has access to a witness FNP oracle, and on any input of length n queries the oracle at most $O(\log n)$ many times. In a sense, it is the search variant of $\mathrm{P}^{\mathrm{NP}}[\log]$.

The class $\mathrm{F}\Sigma^{\mathrm{p}}_2$ consists of all search problems that can be computed by a nondeterministic polynomial-time Turing machine that has access to a witness FNP oracle. In a sense, it is the search variant of Σ^{p}_2.

We say that a decision problem P is complete for a complexity class C containing decision problems if any problem $Q \in C$ is polynomial-time (many-one) reducible to P. Similarly, we say that a search problem R is complete for a complexity class C containing search problems if for any problem $R' \in C$ there exists a polynomial-time Levin reduction from R' to R.

Complete Problems.

Next, we consider a number of search and decision problems that are complete for the various complexity classes that we consider in this paper. The decision problem SAT is NP-complete. This problem consists of deciding whether a given propositional formula φ has a satisfying assignment. The corresponding search problem, FSAT, consisting of all pairs (φ, α), where α is a satisfying assignment for φ, is FNP-complete. Next, consider the following problems.

X-Max-Model
Input: a formula φ and a subset $X \subseteq \mathrm{Var}(\varphi)$ of variables of φ.
Output: a model M of φ that sets a maximum number of variables in X to true, if such a model exists.

X-Max-Model-Parity
Instance: a formula φ, a subset $X \subseteq \mathrm{Var}(\varphi)$ of variables of φ, and a variable $x_0 \in X$.
Question: is there a model M of φ such that X-Max-Model(φ, M) and M sets x_0 to true?

The search problem X-Max-Model is complete for $\mathrm{FP}^{\mathrm{NP}}[\log,\mathrm{wit}]$ [7]. Moreover, the decision problem X-Max-Model-Parity is complete for $\mathrm{P}^{\mathrm{NP}}[\log]$. This follows from the fact that all predicates in $\mathrm{FP}^{\mathrm{NP}}[\log,\mathrm{wit}]$ are computable in $\mathrm{P}^{\mathrm{NP}}[\log]$ [18, Corollary 6.3.5].

FLex-Max-Model
Input: a formula φ with $\mathrm{Var}(\varphi) = \{x_1, \dots, x_n\}$.
Output: a lexicographically maximal model M of φ, if such a model exists. (Here, the variables are ordered $x_1 < \cdots < x_n$.)

Lex-Max-Model
Instance: a formula φ and a variable $x_0 \in \mathrm{Var}(\varphi)$.
Question: is there a model M of φ such that FLex-Max-Model(φ, M) and M sets x_0 to true?

The search problem FLex-Max-Model is complete for $\mathrm{FP}^{\mathrm{NP}}$, and the decision problem Lex-Max-Model is complete for P^{NP} [19].

FQSat$_2$
Input: a quantified Boolean formula $\varphi = \exists X.\forall Y.\psi$, where ψ is quantifier-free.
Output: an assignment α to the variables in X, such that $\forall Y.\psi[\alpha]$ is true, if it exists.

The decision problem QSat$_2$ associated to FQSat$_2$ is well-known to be Σ^{p}_2-complete. The search problem FQSat$_2$ is complete for $\mathrm{F}\Sigma^{\mathrm{p}}_2$. This can be shown straightforwardly by modifying the proof of Σ^{p}_2-completeness for QSat$_2$ [27, 28], which can also be found in many textbooks (e.g., [2]).

3. COMPLEXITY RESULTS

For the various judgment aggregation procedures F, we will consider the following decision problem.

WinDet(F)
Instance: an agenda Φ with an integrity constraint Γ, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi,\Gamma)^n$ and a number of subsets $L_0, \dots, L_u \subseteq \Phi$ of the agenda, with $u \geq 0$.
Question: is there a judgment set $J^* \in \mathrm{Winner}_{\mathrm{F},\Phi,\Gamma}(\boldsymbol{J})$ such that $L_0 \subseteq J^*$ and $L_i \not\subseteq J^*$ for each $1 \leq i \leq u$?

In addition, we will consider the following search problem.

FWinDet(F)
Input: an agenda Φ with an integrity constraint Γ, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi,\Gamma)^n$ and a number of subsets $L_0, \dots, L_u \subseteq \Phi$ of the agenda, with $u \geq 0$.
Output: a judgment set $J^* \in \mathrm{Winner}_{\mathrm{F},\Phi,\Gamma}(\boldsymbol{J})$ such that $L_0 \subseteq J^*$ and $L_i \not\subseteq J^*$ for each $1 \leq i \leq u$, if it exists.

In these problems, the set L_0 allows us to specify judgments that the outcome must contain. Moreover, using the sets $L_1, \dots, L_u$, we can specify a list of forbidden (combinations of) judgments that the outcome may not contain. This allows us, for instance, to enumerate several outcomes by using previously found outcomes as the sets $L_1, \dots, L_u$.

We make several observations about this formalization of the computational task of winner determination for the judgment aggregation procedures. In previous work [13, 21], only decision problems have been studied. We argue that the search problem formulation is a more accurate formalization of the computational task related to the winner determination problem in judgment aggregation. In general, for any judgment aggregation procedure, given a multiagent setting and individual opinions of the agents, one would like to compute a group opinion, which could then be used for various purposes. Given access to an algorithm that performs the computational task captured by the decision problem, one needs a linear number of calls to produce the description of an outcome. However, it turns out that (under some common complexity-theoretic assumptions) any such (deterministic) algorithm invokes an NP oracle only a small (sublinear) number of times in the worst case. This was known for the case of the Kemeny rule [13], and we show that it is also the case for the Slater and Young rules. Therefore, it could be that an algorithm that produces an outcome by simply calling an algorithm for the decision problem a linear number of times exceeds the minimum number of calls to the

NP oracle that are needed. In fact, we show that producing an outcome can be done in deterministic polynomial time with only a logarithmic number of calls to an oracle that produces witnesses for NP problems. Formalizing the task as a search problem in the first place circumvents this issue.

Secondly, these computational problems involve the question whether there exists an outcome that satisfies certain properties, rather than the question whether all outcomes satisfy certain properties (as studied by Lang and Slavkovik [21]). We think the former question is the more natural of these two, as computing outcomes for judgment aggregation procedures is arguably the most central computational task in judgment aggregation.

The third observation concerns the condition that the subsets $L_1, \dots, L_u$, given as part of the input to the problems, are not subsets of the judgment set $J^* \in \text{Winner}_{\text{F},\Phi,\Gamma}(\boldsymbol{J})$. As can be seen in the proofs in the remainder of this section, this condition does not have an effect on the complexity analysis of the problem. However, it does allow us to use this formalization of the problem to devise an algorithm to enumerate outcomes of judgment aggregation procedures. In order to enumerate outcomes, one could use an algorithm that solves FWinDet(F) repeatedly, and rule out previously found outcomes by providing them as the subsets L_i (with $i > 0$).

We obtain the following complexity results for these two computational problems (summarized in Table 1).

Theorem 1. *For each* $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$*:*

a) WinDet(F) *is* $\text{P}^{\text{NP}}[\text{log}]$*-complete; and*

b) FWinDet(F) *is* $\text{FP}^{\text{NP}}[\text{log,wit}]$*-complete.*

Theorem 2. *For any tie-breaking rule A, it holds that* WinDet(RA_A) *is in* P^{NP} *and* FWinDet(RA_A) *is in* FP^{NP}*. Moreover, there is some tie-breaking rule A such that* WinDet(RA_A) *is* P^{NP}*-complete and* FWinDet(RA_A) *is* FP^{NP}*-complete.*

Theorem 3. WinDet(RA) *is* Σ_2^{p}*-complete and* FWinDet(RA) *is* $\text{F}\Sigma_2^{\text{p}}$*-complete.*

We would like to point out that the complexity result for WinDet(Kemeny) has been shown before [13]. We give an alternative proof of hardness for this problem. We prove hardness by giving a reduction directly from a canonical complete problem for $\text{P}^{\text{NP}}[\text{log}]$, which allows us to extend this hardness result to the case of the search problem FWinDet(Kemeny), which would not have been at all straightforward with the known $\text{P}^{\text{NP}}[\text{log}]$-hardness proof for WinDet(Kemeny) from the literature.

We begin by proving Theorem 1 in Sections 3.1–3.3 (Propositions 5–9 and Corollaries 6–10). Then, we prove Theorems 2 and 3 in Section 3.4 (Propositions 11–12 and Corollaries 13–14, and Propositions 15–16 and Corollary 17, respectively).

3.1 Membership for Kemeny, Slater, Young

We will show $\text{FP}^{\text{NP}}[\text{log,wit}]$-membership of the problem FWinDet(F), for each $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$ by giving a polynomial-time algorithm that uses an FNP witness oracle at most $O(\log n)$ many times. Since $\text{P}^{\text{NP}}[\text{log}]$ coincides with $\text{P}^{\text{NP}}[\text{log,wit}]$, this algorithm can then easily be adapted to show $\text{P}^{\text{NP}}[\text{log}]$-membership of WinDet(F). The approach that we use to show these membership results is similar to the approach taken in the known $\text{P}^{\text{NP}}[\text{log}]$-membership proof of WinDet(Kemeny) [13].

For each $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$, we will consider the following auxiliary problem.

FSetScore(F)
Input: an agenda Φ with an integrity constraint Γ, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi,\Gamma)$, a number of subsets $L_0, \dots, L_u \subseteq \Phi$, with $u \geq 0$, and a positive integer m (in unary).
Output: a consistent and complete judgment set J such that $L_0 \subseteq J$, $L_i \not\subseteq J$ for each $1 \leq i \leq u$, and $\text{Score}_{\text{F}}(J, \boldsymbol{J}) \leq m$, if it exists.

Lemma 4. *For each* $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$*, it holds that* FSetScore(F) $\in$ FNP.

Proof (sketch). We describe a guess-and-check algorithm for FSetScore(F). The algorithm gets as input an agenda Φ with an integrity constraint Γ, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi,\Gamma)$, a number of subsets $L_0, \dots, L_u \subseteq \Phi$ of the agenda, with $u \geq 0$, and a positive integer m in unary. The algorithm guesses a complete judgment set J, and an interpretation M that witnesses that J is Γ-consistent. Additionally, if F = Young, the algorithm guesses $\ell \leq m$ many judgment sets $J_{i_1}, \dots, J_{i_\ell}$ in $\boldsymbol{J}$ to remove from $\boldsymbol{J}$. Then, the algorithm verifies (1) whether M satisfies both J and Γ, (2) whether $\text{Score}_{\text{F}}(J, \boldsymbol{J}) \leq m$, (3) whether $L_0 \subseteq J$, and (4) whether $L_i \not\subseteq J$ for each $1 \leq i \leq u$. If F = Young, the verification of (2) uses the guessed judgment sets $J_{i_1}, \dots, J_{i_\ell}$. It is straightforward to verify that these checks can be done in polynomial time. If J satisfies conditions (1)–(4), the algorithm accepts and outputs J; otherwise, the algorithm rejects. □

Proposition 5. *For each* $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$*,* FWinDet(F) $\in \text{FP}^{\text{NP}}[\text{log,wit}]$.

Proof (sketch). We describe a polynomial-time algorithm that computes FWinDet(F) by using an FSetScore(F) oracle. The algorithm firstly computes the minimal number m^* such that there is some Γ-consistent and complete judgment set J such that $\text{Score}_{\text{F}}(J, \boldsymbol{J}) \leq m^*$. For each $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$, we know that m^* is polynomially bounded by the size of (the encoding of) $\boldsymbol{J}$. Therefore, we can employ a binary search strategy to compute m^* by asking $O(\log |\boldsymbol{J}|)$ many oracle queries of the form $(\Phi, \Gamma, \boldsymbol{J}, \emptyset, m)$. Then, the algorithm returns the answer to the oracle query $(\Phi, \Gamma, \boldsymbol{J}, L_0, \dots, L_u, m^*)$. □

Corollary 6. *For each* $\text{F} \in \{\text{Kemeny}, \text{Slater}, \text{Young}\}$*,* WinDet(F) $\in \text{P}^{\text{NP}}[\text{log}]$.

3.2 Hardness for Kemeny and Slater

Proposition 7. *The problems* FWinDet(Kemeny) *and* FWinDet(Slater) *are* $\text{FP}^{\text{NP}}[\text{log,wit}]$*-hard, even with the restriction that* $\Gamma = \top$*, and* $u = 0$.

Proof (sketch). We give a polynomial-time Levin reduction from X-Max-Model. The same reduction works for both FWinDet(Kemeny) and FWinDet(Slater). Let (φ, X) be an input to X-Max-Model with $X = \{x_1, \dots, x_v\} \subseteq \text{Var}(\varphi)$. We construct an agenda Φ, an integrity constraint $\Gamma = \top$, an integer $u = 0$, a subset $L_0 \subseteq \Phi$, and a profile $\boldsymbol{J} \in \mathcal{J}(\Phi,\Gamma)^n$ as follows. We pick some w

such that $w > v^2$, e.g., $w = 2v^2$. For each $1 \leq i \leq v$ and $1 \leq j \leq w$, we introduce a fresh variable $z_{j,i}$. Moreover, we introduce a fresh variabe y. Then, we construct a propositional formula φ' that is true if and only if one of the following conditions holds: either (i) for some $1 \leq i \leq v$, x_i is set to false, y is set to false, and all $z_{j,i}$ for $1 \leq j \leq w$ are set to true, or (ii) all $z_{j,i}$ are set to false, y is set to true, and the assignment to the variables x_i satisfies φ. It is straightforward to construct in polynomial time some formula φ' that satisfies this property. Then we introduce w many syntactic copies φ'_j of φ'. We now define the agenda Φ by letting $[\Phi] = \{y\} \cup \{x_1, \dots, x_v\} \cup \{\, z_{j,i} : 1 \leq j \leq w, 1 \leq i \leq n \,\} \cup \{\varphi'_1, \dots, \varphi'_w\}$. Next, we construct the profile $\boldsymbol{J} = (J_1, \dots, J_v)$ as shown in Figure 1. Moreover, we let $L_0 = \{y\}$.

$\boldsymbol{J}$	J_1	J_2	$\dots$	J_{v-1}	J_v	$m(\boldsymbol{J})$
y	0	0	0	$\dots$	0	0
x_1	0	1	1	$\dots$	1	1
x_2	1	0	1	$\dots$	1	1
$\vdots$	$\vdots$		$\ddots$		$\vdots$	$\vdots$
x_{v-1}	1	$\dots$	1	0	1	1
x_v	1	$\dots$	1	1	0	1
$z_{1,1}$	1	0	0	$\dots$	0	0
$z_{1,2}$	0	1	0	$\dots$	0	0
$\vdots$	$\vdots$		$\ddots$		$\vdots$	$\vdots$
$z_{1,v-1}$	0	$\dots$	0	1	0	0
$z_{1,v}$	0	$\dots$	0	0	1	0
$\vdots$			$\vdots$			$\vdots$
$z_{w,1}$	1	0	0	$\dots$	0	0
$z_{w,2}$	0	1	0	$\dots$	0	0
$\vdots$	$\vdots$		$\ddots$		$\vdots$	$\vdots$
$z_{w,v-1}$	0	$\dots$	0	1	0	0
$z_{w,v}$	0	$\dots$	0	0	1	0
φ'_1	1	1	$\dots$		1	1
φ'_2	1	1	$\dots$		1	1
$\vdots$	$\vdots$		$\ddots$		$\vdots$	$\vdots$
φ'_{w-1}	1	$\dots$		1	1	1
φ'_w	1	$\dots$		1	1	1

Figure 1: construction of the profile in the proof of Proposition 7.

What remains is to specify a polynomial-time computable function g that takes some $J^* \in \text{Winner}_{\text{F},\Phi,\Gamma}(\boldsymbol{J})$ with $L_0 \subseteq J^*$, if it exists, and that produces a model M such that X-MAX-MODEL(φ, X, M). Let $J^* \in \text{Winner}_{\text{F},\Phi,\Gamma}(\boldsymbol{J})$ such that $L_0 \in J^*$. Then the function g outputs $J^* \cap \{x_1, \dots, x_v\}$.

The intuition behind this reduction is the following. The agenda and the profile are constructed in such a way that it is 'cheaper' to agree with the overall profile (respectively, with the majority outcome) on the formulas φ'_i and the formulas $z_{j,i}$, if possible. If φ is satisfiable, the cheapest way to agree with the overall profile (respectively, with the majority outcome) on the above formulas (in a consistent judgment set) is to set the variables x_i to a maximal assignment α to the variables in X that is extendable to a satisfying assignment to φ, and to set y to true. If φ is unsatisfiable, all consistent judgment sets that disagree with the majority outcome on a minimal number of formulas (e.g., setting $z_{j,i}$ to true for some $1 \leq i \leq v$ and all $1 \leq j \leq w$) do not contain y (and thus are not a superset of L_0). □

COROLLARY 8. *The problems* WINDET(KEMENY) *and* WINDET(SLATER) *are* $\text{P}^{\text{NP}}[\text{log}]$*-hard, even with the restriction that* $\Gamma = \top$*, and* $u = 0$*.*

PROOF (SKETCH). The reduction in the proof of Proposition 7 can be adapted to a many-one reduction from WINDET(KEMENY) (respectively, from WINDET(SLATER)) to X-MAX-MODEL-PARITY, by letting $L_0 = \{y, x_0\}$, where x_0 is given in the original input for the problem X-MAX-MODEL-PARITY. □

3.3 Hardness for Young

PROPOSITION 9. FWINDET(YOUNG) *is* $\text{FP}^{\text{NP}}[\text{log,wit}]$*-hard, even with the restriction that* $\Gamma = \top$*, and* $u = 0$*.*

PROOF (SKETCH). We give a polynomial-time Levin reduction from X-MAX-MODEL. Let φ be an input to X-MAX-MODEL with $\text{Var}(\varphi) = \{x_1, \dots, x_v\}$. We may assume that any satisfying assignment (if there exists any) sets at least one variable in X to true, and that setting all variables in X to true does not satisfy φ. We construct an agenda Φ, an integrity constraint $\Gamma = \top$, an integer $u = 0$, a subset $L_0 \subseteq \Phi$, and a profile $\boldsymbol{J} \in \mathcal{J}(\Phi, \Gamma)^n$ as follows. We let $w = 2v + 1$. Then, we construct a propositional formula φ' that is satisfiable if and only if one of the following conditions holds: either (i) for some $1 \leq j \leq w$, y_j is set to true, or (ii) all y_j are set to false and the assignment to the variables x_i reduces φ to a satisfiable formula. It is straightforward to construct in polynomial time some formula φ' that satisfies this property. We now define the agenda Φ by letting $[\Phi] = \{x_1, \dots, x_v\} \cup \{\, y_j : 1 \leq j \leq w \,\} \cup \{\varphi'\}$. Next, we construct the profile $\boldsymbol{J} = (J_1, \dots, J_w)$ as shown in Figure 2. Moreover, we let $L_0 = \{\neg y_1, \dots, \neg y_w\}$.

What remains is to specify a polynomial-time computable function g that takes some $J^* \in \text{Winner}_{\text{Young},\Phi,\Gamma}(\boldsymbol{J})$ with $L_0 \subseteq J^*$, if it exists, and that produces a model M such that X-MAX-MODEL(φ, M). Let $J^* \in \text{Winner}_{\text{Young},\Phi,\Gamma}(\boldsymbol{J})$. Then the function g outputs $J^* \cap \{x_1, \dots, x_v\}$.

The intuition behind this reduction is the following. Since the profile is unanimous on φ', any winner must include φ'. Satisfying φ' can be done by either setting some y_i to true, or by satisfying the original formula φ. In the case that φ is satisfiable, a maximal assignment α to the variables in X that is extendable to a satisfying assignment for φ corresponds to the minimum number of judgment sets that need to be removed from the profile to get a consistent majority outcome (namely, remove exactly those J_i for which α sets x_i to false, and remove an equal number of judgment sets J_j for $v + 1 \leq j \leq w$). In the case that φ is unsatisfiable, the only subprofiles that have a consistent majority outcome are subprofiles consisting of a single judgment set J, for which all holds that $L_0 \not\subseteq J$. □

COROLLARY 10. WINDET(YOUNG) *is* $\text{P}^{\text{NP}}[\text{log}]$*-hard, even with the restriction that* $\Gamma = \top$*, and* $u = 0$*.*

PROOF (SKETCH). The reduction in the proof of Proposition 9 can be adapted to a many-one reduction from WINDET(YOUNG) to X-MAX-MODEL-PARITY, by letting $L_0 = \{x_0, x_1\}$, where x_0 is given in the original input for the problem X-MAX-MODEL-PARITY. □

$\boldsymbol{J}$	J_1	J_2	J_3	$\dots$	J_v	J_{v+1}	J_{v+2}	$\dots$	J_w
φ'	1	1	1	$\dots$	1	1	1	$\dots$	1
x_1	1	0	0	$\dots$	0	1	1	$\dots$	1
x_2	0	1	0	$\dots$	0	1	1	$\dots$	1
$\vdots$	$\vdots$		$\ddots$		$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
x_{v-1}	0	$\dots$	0	1	0	1	1	$\dots$	1
x_v	0	$\dots$	0	0	1	1	1	$\dots$	1
y_1	1	0	$\dots$				$\dots$	0	0
y_2	0	1	$\dots$				$\dots$	0	0
$\vdots$	$\vdots$				$\ddots$				$\vdots$
y_{w-1}	0	0	$\dots$				$\dots$	1	0
y_w	0	0	$\dots$				$\dots$	0	1

Figure 2: construction of the profile in the proof of Proposition 9.

3.4 Completeness for Tideman

Proposition 11. *Let $<^A$ be some tie-breaking rule. Then* $\text{FWinDet}(\text{RA}_A)$ *is in* FP^{NP}.

Proof (sketch). We describe a deterministic polynomial-time algorithm with access to a (yes-no) SAT oracle that solves $\text{FWinDet}(\text{RA}_A)$. The algorithm takes as input an agenda Φ, an integrity constraint Γ, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi,\Gamma)^n$, and a number of subsets $L_0,\dots,L_u$ of the agenda with $u \geq 0$. In polynomial time, the algorithm computes $\leq_{\boldsymbol{J}}$, and computes the Ranked-Agenda order $<^A_{\boldsymbol{J}}$, using the tie-breaking rule $<^A$. Let $\Phi = \{\varphi_1,\dots,\varphi_{2m}\}$, where $\varphi_1 >^A_{\boldsymbol{J}} \cdots >^A_{\boldsymbol{J}} \varphi_{2m}$. We compute $\text{RA}(<^A_{\boldsymbol{J}},\Phi,\Gamma) = J_{2m}$ by iteratively computing J_i, for all $0 \leq i \leq 2m$. Let $J_0 = \emptyset$. Given J_i, we compute J_{i+1} as follows, by querying the SAT oracle. We query the SAT solver whether $\psi_{i+1} = \bigwedge_{\varphi \in J_i} \varphi \wedge \varphi_{i+1}$ is Γ-consistent. If ψ_{i+1} is Γ-consistent, we let $J_{i+1} = J_i \cup \{\varphi_{i+1}\}$; otherwise, we let $J_{i+1} = J_i$. Clearly, this requires $2m = |\Phi|$ calls to the SAT oracle. Then, let $J^* = J_{2m} = \text{RA}(<^A_{\boldsymbol{J}},\Phi,\Gamma)$. Finally, the algorithm verifies whether $L_0 \subseteq J^*$ and $L_i \not\subseteq J^*$ for all $1 \leq i \leq u$. If these checks do not succeed, the algorithm rejects the input; otherwise, it returns J^*. □

Proposition 12. $\text{FWinDet}(\text{RA}_A)$ *is* FP^{NP}*-hard for some (fixed) tie-breaking rule A, even with the restriction that $\Gamma = \top$, and $u = 0$.*

Proof (sketch). We give a polynomial-time Levin reduction from FLex-Max-Model. Let φ be an input for FLex-Max-Model, with $\text{Var}(\varphi) = \{x_1,\dots,x_v\}$. We construct an agenda Φ, an integrity constraint $\Gamma = \top$, an integer $u = 0$, a linear order $<^A \subseteq \Phi \times \Phi$, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi,\Gamma)^n$, and a subset $L_0 \subseteq \Phi$, as follows. We pick some even w such that $w > v$, e.g., $w = 2v$. We introduce auxiliary variables $y_1,\dots,y_w$. Then, we construct a propositional formula φ' that is true if and only if one of the following conditions holds: either (i) for some $1 \leq j \leq w$, y_j is set to true, or (ii) the assignment to the variables x_i satisfies φ. We define the agenda Φ by letting $[\Phi] = \{\varphi'\} \cup \{y_1,\dots,y_w\} \cup \{x_1,\dots,x_v\}$. Then, we define the total order $<^A$ by letting $\varphi' >^A \neg y_1 >^A \cdots >^A \neg y_w >^A x_1 >^A \neg x_1 >^A \cdots >^A x_v >^A \neg x_v >^A y_1 >^A \cdots >^A y_w >^A \neg\varphi'$. Next, we construct the profile $\boldsymbol{J} = (J_1,\dots,J_w)$ as shown in Figure 3. Note that the Ranked-Agenda order $<^A_{\boldsymbol{J}}$ coincides with the order $<^A$. Finally, we let $L_0 = \{\neg y_w\}$.

$\boldsymbol{J}$	J_1	J_2	J_3	J_4	$\dots$	J_{w-1}	J_w
φ'	1	1	1	1	$\dots$	1	1
x_1	1	0	1	0	$\dots$	1	0
x_2	1	0	1	0	$\dots$	1	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
x_{v-1}	1	0	1	0	$\dots$	1	0
x_v	1	0	1	0	$\dots$	1	0
y_1	1	0	0	$\dots$			0
y_2	0	1	0	$\dots$			0
$\vdots$	$\vdots$			$\ddots$			$\vdots$
y_{w-1}	0	$\dots$			0	1	0
y_w	0	$\dots$			0	0	1

Figure 3: construction of the profile in the proof of Proposition 12.

What remains is to specify a polynomial-time computable function g that takes some $J^* \in \text{Winner}_{\text{RA}_A,\Phi,\Gamma}(\boldsymbol{J})$ with $L_0 \subseteq J^*$, and that produces a model M such that $\text{FLex-Max-Model}(\varphi, M)$. The function g outputs $J^* \cap \{x_1,\dots,x_v\}$.

Note that for each instance φ of FLex-Max-Model, this reduction produces a different agenda Φ. Therefore, there is a single tie-breaking rule A that produces the correct order $<^A$ for each agenda Φ that is generated by this reduction (and moreover, this order $<^A$ clearly is computable in polynomial time, given φ).

The intuition behind this reduction is the following. If φ is satisfiable, the Ranked-Agenda winner (for the tie-breaking rule A) will contain φ' and $\neg y_1,\dots,\neg y_w$, because these are consistent (and are the first formulas in the order $>^A_{\boldsymbol{J}}$, and if φ is unsatisfiable, the winner will not contain $\neg y_w$, because φ' and $\neg y_1,\dots,\neg y_{w-1}$ are not consistent with $\neg y_w$. Moreover, if φ is satisfiable, the Ranked-Agenda order $>^A_{\boldsymbol{J}}$ forces the winner to contain the lexicographically maximal model of φ. □

Corollary 13. *Let $<^A$ be some tie-breaking rule. Then* $\text{WinDet}(\text{RA}_A)$ *is in* P^{NP}.

Corollary 14. $\text{WinDet}(\text{RA}_A)$ *is* P^{NP}*-hard for some (fixed) tie-breaking rule A, even with the restriction that $\Gamma = \top$, and $u = 0$.*

Proof (sketch). The reduction in the proof of Proposition 12 can be adapted to a many-one reduction from

WinDet(RA$_A$) to Lex-Max-Model, by letting $L_0 = \{\neg y_w, x_0\}$, where x_0 is given in the original input for the problem Lex-Max-Model. $\square$

Because the winner determination problem is hard for the Δ_2^p-level of the PH for some tie-breaking rule, we know that one cannot obtain lower complexity results without exploiting the structure of tie-breaking rules. In other words, any lower complexity results cannot hold for all tie-breaking rules, and their proofs will have to take into account some specific properties of any tie-breaking rules for which the results hold.

Proposition 15. FWinDet(RA) *is in* $\mathrm{F}\Sigma_2^p$.

Proof (sketch). We describe a nondeterministic polynomial-time algorithm with access to a (yes-no) SAT oracle that solves FWinDet(RA). The algorithm takes as input an agenda Φ, an integrity constraint Γ, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi, \Gamma)^n$, and a number of subsets $L_0, \dots, L_u \subseteq \Phi$ of the agenda. It is straightforward to compute $\leq_{\boldsymbol{J}}$ in polynomial time. Then, the algorithm guesses some total order $<^A_{\boldsymbol{J}} \subseteq \Phi \times \Phi$. It is straightforward to verify in polynomial time that $<^A_{\boldsymbol{J}}$ extends the partial order $\leq_{\boldsymbol{J}}$. (If $<^A_{\boldsymbol{J}}$ does not extend $\leq_{\boldsymbol{J}}$, the algorithm rejects.) Then, the algorithm proceeds similarly to the algorithm described in the proof of Proposition 11 to compute $J^* = \mathrm{RA}(<^A_{\boldsymbol{J}}, \Phi, \Gamma)$. Also, the algorithm verifies whether $L_0 \subseteq J^*$ and $L_i \not\subseteq J^*$ for all $1 \leq i \leq u$. If these checks do not succeed, the algorithm rejects; otherwise, it returns J^*. $\square$

Proposition 16. FWinDet(RA) *is* $\mathrm{F}\Sigma_2^p$*-hard, even with the restriction that* $\Gamma = \top$*, and* $u = 0$.

Proof (sketch). We give a polynomial-time Levin reduction from FQSat$_2$. Let $\varphi = \exists X.\forall Y.\psi$ be an instance of FQSat$_2$, where $X = \{x_1, \dots, x_v\}$. We may assume without loss of generality that ψ is not a tautology. We construct an agenda Φ, an integrity constraint $\Gamma = \top$, an integer $u = 0$, a profile $\boldsymbol{J} \in \mathcal{J}(\Phi, \Gamma)^n$, and a subset $L_0 \subseteq \Phi$ of the agenda, as follows. We pick some w such that $w > v$, e.g., $w = 2v$. We introduce auxiliary variables $y_1, \dots, y_w$. Also, we introduce auxiliary variables z_l for each $l \in \{ x_i, \neg x_i : 1 \leq i \leq v \}$. Then we construct a propositional formula χ that is true if and only if one of the following two conditions holds: either (i) some y_i is true, or (ii) for each $l \in \{ x_i, \neg x_i : 1 \leq i \leq v \}$ it holds that l is true if and only if z_l is true. Moreover, we construct a propositional formula ψ' that is true if and only if both of the following two conditions holds: both (i) no y_i with $1 \leq i \leq 3$ is true, and (ii) either (ii.a) some y_i is true, or (ii.b) the formula $\neg\psi$ is satisfiable. It is straightforward to construct these formulas χ and ψ' in polynomial time. We now define the agenda Φ by letting $[\Phi] = \{\chi, \psi'\} \cup \{ z_{x_i}, z_{\neg x_i} : 1 \leq i \leq v \} \cup \{y_1, \dots, y_w\}$. Next, we construct the profile $\boldsymbol{J} = (J_1, \dots, J_w)$ as shown in Figure 4. Note that $m(\boldsymbol{J}, \chi) = w$; $m(\boldsymbol{J}, \psi') = w - 3$; for each $1 \leq i \leq v$, $m(\boldsymbol{J}, z_{x_i}) = m(\boldsymbol{J}, z_{\neg x_i}) = w - 2$; for each $1 \leq j \leq w$, $m(\boldsymbol{J}, \neg y_j) = w - 1$. Therefore $\{\chi\} \leq_{\boldsymbol{J}} \{ \neg y_j : 1 \leq j \leq w \} \leq_{\boldsymbol{J}} \{ z_{x_i}, z_{\neg x_i} : 1 \leq i \leq v \} \leq_{\boldsymbol{J}} \{\psi'\}$. Moreover, we let $L_0 = \{\neg\psi'\}$.

What remains is to specify a polynomial-time computable function g that takes some $J^* \in \mathrm{Winner}_{\mathrm{RA},\Phi,\Gamma}(\boldsymbol{J})$ with $L_0 \subseteq J^*$, and that produces a model M such that FQSat$_2(\varphi, M)$. Let $J^* \in \mathrm{Winner}_{\mathrm{RA},\Phi,\Gamma}(\boldsymbol{J})$. The function g outputs $\{ x_i : 1 \leq i \leq v, z_{x_i} \in J^* \}$.

$\boldsymbol{J}$	J_1	J_2	J_3	J_4	$\dots$	J_w
χ	1	1	1	1	$\dots$	1
ψ'	0	0	0	1	$\dots$	1
z_{x_1}	0	0	1	$\dots$		1
$z_{\neg x_1}$	0	0	1	$\dots$		1
$\vdots$	$\vdots$	$\vdots$	$\vdots$			$\vdots$
z_{x_v}	0	0	1	$\dots$		1
$z_{\neg x_v}$	0	0	1	$\dots$		1
y_1	1	0	0	$\dots$		0
y_2	0	1	0	$\dots$		0
$\vdots$	$\vdots$		$\ddots$			$\vdots$
				$\ddots$		
y_{w-1}	0	$\dots$		0	1	0
y_w	0	$\dots$		0	0	1

Figure 4: construction of the profile in the proof of Proposition 16.

The intuition behind this reduction is the following. The profile is constructed in such a way that every Ranked-Agenda winner contains χ. If there is an assignment α to the variables $\{x_1, \dots, x_v\}$ such that $\forall Y.\psi[\alpha]$ is true, then there exists a winner containing all literals $\neg y_j$, the variables z_l for those literals l that are satisfied by α, and the formula $\neg\psi'$, because any set containing χ, the literals $\neg y_j$, and those variables z_l would be inconsistent with ψ', since such a set implies ψ (and thus forces $\neg\psi$ to be unsatisfiable). Moreover, any winner containing $\neg\psi'$ is of such a form, and thus corresponds to an assignment α with the property that $\forall Y.\psi[\alpha]$ is true. $\square$

Corollary 17. WinDet(RA) *is* Σ_2^p*-complete. Hardness holds even with the restriction that* $\Gamma = \top$*, and* $u = 0$.

4. CONCLUSION

We studied the computational complexity of the winner determination problem for (the judgment aggregation analogues of) the Kemeny rule, the Slater rule and the Young rule, as well as two variants of the Ranked-Agenda rule. These computational tasks we formalized in the form of a search problem and a decision problem. We showed that the complexity for the Kemeny, Slater and Young rules lies at the Θ_2^p-level of the PH, and we showed that the complexity of the Ranked-Agenda rule, lies at the Δ_2^p-level of the PH, in case of a fixed tie-breaking rule, and lies at the Σ_2^p-level of the PH in case ties can be broken in arbitrary ways.

Future research should include investigating the computational complexity of the winner determination problem for other complete and consistent judgment aggregation procedures. One example of such a procedure is one based on the distance measure introduced by Duddy and Piggins [11]. Another direction for further research is to investigate the contribution of various aspects of the problem to its computational complexity, by analyzing the problems using the framework of parameterized complexity theory.

Acknowledgments

Ronald de Haan is supported by the European Research Council (ERC), project 239962, and the Austrian Science Fund (FWF), project P26200.

REFERENCES

[1] N. Alon, D. Falik, R. Meir, and M. Tennenholtz. Bundling attacks in judgment aggregation. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI 2013)*. AAAI Press, 2013.

[2] S. Arora and B. Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009.

[3] D. Baumeister, G. Erdélyi, O. J. Erdélyi, and J. Rothe. Computational aspects of manipulation and control in judgment aggregation. In P. Perny, M. Pirlot, and A. Tsoukiàs, editors, *Proceedings of the Third International Conference on Algorithmic Decision Theory (ADT 2013), Brussels, Belgium, November 13–15, 2013*, volume 8176 of *Lecture Notes in Computer Science*, pages 71–85. Springer Verlag, 2013.

[4] D. Baumeister, G. Erdélyi, and J. Rothe. How hard is it to bribe the judges? A study of the complexity of bribery in judgment aggregation. In *Proceedings of the Second International Conference on Algorithmic Decision Theory (ADT 2011), Piscataway, NJ, USA, October 26–28, 2011*, volume 6992 of *Lecture Notes in Computer Science*, pages 1–15. Springer Verlag, 2011.

[5] S. J. Brams and P. C. Fishburn. Voting procedures. In K. J. Arrow, A. K. Sen, and K. Suzumura, editors, *Handbook of Social Choice and Welfare*, volume 1, chapter 4, pages 173–236. Elsevier Science Publishers, North-Holland, 2002.

[6] F. Brandt, V. Conitzer, and U. Endriss. Computational social choice. In G. Weiss, editor, *Multiagent Systems*, pages 213–283. MIT Press, 2013.

[7] Z.-Z. Chen and S. Toda. The complexity of selecting maximal solutions. *Information and Computation*, 119:231–239, June 1995.

[8] V. Conitzer, J. Lang, and L. Xia. How hard is it to control sequential elections via the agenda? In C. Boutilier, editor, *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, pages 103–108, 2009.

[9] F. Dietrich and C. List. Arrow's Theorem in judgment aggregation. *Social Choice and Welfare*, 29(1):19–33, 2007.

[10] F. Dietrich and C. List. Judgment aggregation under constraints. In T. Boylan and R. Gekker, editors, *Economics, Rational Choice and Normative Philosophy*, Routledge Frontiers of Political Economy. Taylor & Francis, 2008.

[11] C. Duddy and A. Piggins. A measure of distance between judgment sets. *Social Choice and Welfare*, 39(4):855–867, 2012.

[12] U. Endriss. Judgment aggregation. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia, editors, *Handbook of Computational Social Choice*. Cambridge University Press, 2015. Forthcoming.

[13] U. Endriss, U. Grandi, and D. Porello. Complexity of judgment aggregation. *J. Artif. Intell. Res.*, 45:481–514, 2012.

[14] O. Goldreich. *P, NP, and NP-Completeness: The Basics of Complexity Theory*. Cambridge University Press, 2010.

[15] U. Grandi. *Binary Aggregation with Integrity Constraints*. PhD thesis, University of Amsterdam, 2012.

[16] U. Grandi and U. Endriss. Lifting integrity constraints in binary aggregation. *Artificial Intelligence*, 199–200:45–66, 2013.

[17] D. Grossi and G. Pigozzi. *Judgment Aggregation: A Primer*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool, 2014.

[18] J. Krajicek. *Bounded arithmetic, propositional logic and complexity theory*. Cambridge University Press, 1995.

[19] M. W. Krentel. Generalizations of OptP to the Polynomial Hierarchy. *Theoretical Computer Science*, 97(2):183–198, 1992.

[20] J. Lang, G. Pigozzi, M. Slavkovik, and L. van der Torre. Judgment aggregation rules based on minimization. In *Proceedings of the 13th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-2011)*, 2011.

[21] J. Lang and M. Slavkovik. How hard is it to compute majority-preserving judgment aggregation rules? In *21st European Conference on Artificial Intelligence (ECAI 2014)*. IOS Press, 2014.

[22] C. List and P. Pettit. Aggregating sets of judgments: An impossibility result. *Economics and Philosophy*, 18(1):89–110, 2002.

[23] C. List and C. Puppe. Judgment aggregation: A survey. In P. Anand, P. Pattanaik, and C. Puppe, editors, *Handbook of Rational and Social Choice*. Oxford University Press, 2009.

[24] M. K. Miller and D. Osherson. Methods for distance-based judgment aggregation. *Social Choice and Welfare*, 32(4):575–601, 2009.

[25] G. Pigozzi. Belief merging and the discursive dilemma: An argument-based account of paradoxes of judgment aggregation. *Synthese*, 152(2):285–298, 2006.

[26] D. Porello and U. Endriss. Ontology merging as social choice. In *Proceedings of the 12th International Workshop on Computational Logic in Multiagent Systems (CLIMA-2011)*, volume 6814 of *Lecture Notes in Artificial Intelligence*, pages 157–170. Springer Verlag, July 2011.

[27] L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.

[28] C. Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):23–33, 1976.