Annals of Mathematics and Artificial Intelligence
https://doi.org/10.1007/s10472-023-09910-w

Collective combinatorial optimisation as judgment aggregation

Linus Boes[1] · Rachael Colley[2] · Umberto Grandi[3] · Jérôme Lang[4] · Arianna Novaro[5]

Accepted: 29 October 2023

Abstract
In many settings, a collective decision has to be made over a set of alternatives that has a combinatorial structure: important examples are multi-winner elections, participatory budgeting, collective scheduling, and collective network design. A further common point of these settings is that agents generally submit preferences over issues (e.g., projects to be funded), each having a cost, and the goal is to find a feasible solution maximising the agents' satisfaction under problem-specific constraints. We propose the use of judgment aggregation as a unifying framework to model these situations, which we refer to as collective combinatorial optimisation problems. Despite their shared underlying structure, collective combinatorial optimisation problems have so far been studied independently. Our formulation into judgment aggregation connects them, and we identify their shared structure via five case studies of well-known collective combinatorial optimisation problems, proving how popular rules independently defined for each problem actually coincide. We also chart the computational complexity gap that may arise when using a general judgment aggregation framework instead of a specific problem-dependent model.

✉ Rachael Colley
rachael.colley@glasgow.ac.uk

Linus Boes
linus.boes@uni-duesseldorf.de

Umberto Grandi
umberto.grandi@irit.fr

Jérôme Lang
jerome.lang@lamsade.dauphine.fr

Arianna Novaro
arianna.novaro@univ-paris1.fr

1 Heinrich-Heine-Universität Düsseldorf, Düsseldorf, Germany

2 School of Computing Science, University of Glasgow, Glasgow, Scotland

3 IRIT, Université Toulouse Capitole, Toulouse, France

4 CNRS, PSL, Paris, France

5 CES, Université Paris 1 Panthéon-Sorbonne, Paris, France

Published online: 15 December 2023

Keywords Computational social choice · Judgment aggregation · Combinatorial optimisation · Computational complexity

Mathematics Subject Classification (2010) 68R01 · 68T30

1 Introduction

Public decisions often have a combinatorial structure. Typical examples are participatory budgeting (choosing a set of projects to fund, given some budget constraints), collective scheduling (collectively deciding on the order some tasks will be executed), collective network design (collectively deciding on the edges of a network to connect multiple locations), multi-winner elections (find a committee of a fixed size), and many more—including problems which have not been given much attention, yet are worth investigating.

Many of these problems, including the four examples given above, have been studied separately while sharing many features. Their input mainly consists of individual preferences expressed *locally* (on projects, on the relative order of two projects, on edges between two nodes) rather than *globally* (on all sets of projects, all possible schedules or all possible networks).[1] The output is a feasible solution, i.e., a solution which abides by the constraints and that maximises an objective expressed via a score function. What distinguishes these problems is the nature of the constraints: the maximum budget should not be exceeded, a schedule should be a consistent ordering, a network should be a tree, etc.

The problems mentioned above are thus structurally close, and we may ask whether they could be seen as instances of a more general framework, consisting of a general language for expressing preferences and constraints, general aggregation functions, and general computational tools. If the answer is positive, each of these problems, their variants, as well as novel collective combinatorial optimisation (CCO) problems, could be expressed and solved in such a general framework without the need to study each of them separately.

We give a positive answer, and we do so by showing that *judgment aggregation* is such a suitably general framework. This framework has been initially developed to study the aggregation of binary judgments over logically interconnected issues in an agenda to gain a collective decision (cf. the survey by Endriss, [13]). Given the type of problems that we wish to study, we need a slight generalisation of standard judgment aggregation that can allow for *weighted agendas* and/or *asymmetric agendas*. We show that several aggregation rules used to solve specific problems are actually instances of existing and well-studied judgment aggregation rules, or of some of their variants (such as weighted generalisations).

However, it is worth first clarifying what our paper is not:

(*i*) *We do not define a brand-new framework*: most aspects of the unifying framework that we give are known from judgment aggregation; we do define weighted variants of some well-known rules that are useful to capture CCO problems.

(*ii*) *We do not study new problems*: instead, we restate several existing problems that have been studied independently, such that they can be analysed under a common umbrella framework. By doing so, we do however open the possibility to define and study new problems, such as what we will call *collective placement*.

[1] Since the set of solutions has a combinatorial structure, expressing preferences globally requires a high communication burden from the agents—however, using a local approach limits the types of preferences that can be expressed.

(*iii*) *We do not give better algorithms for specific problems*: in fact, general solvers can perform at best as well as specific solvers, and sometimes worse. Also, for several rules, the computational complexity of solving the outcome determination problem for the general case is higher than that of specific ones.

(*iv*) *We do not give any axiomatisation*: the axiomatisation of (judgment aggregation) rules has been investigated *per se*. The properties of the rules naturally carry over to the specific problems (although an axiomatisation of a general rule does not necessarily lead to that of its specific instantiation).

Our contribution establishes novel and fundamental connections between several lines of research and problems that have so far been studied separately. Identifying those connections may lead to meaningful insights across different areas of research. In doing so, we give an engineering flavour to judgment aggregation, a field that up until now has focused on impossibility results, axiomatisations, and computational complexity, but not yet on concrete applications to real-world problems.

Moreover, for some judgment aggregation rules the outcome can be computed via a translation into integer linear programming (ILP). Of course, such a translation could be done for each specific problem, but the existence of a general translation allows for a simpler process in comparing models, since translating a problem to judgment aggregation can be easier than translating it directly into an ILP.

There are two main streams of related literature. On the one hand, we have papers studying specific settings for collective combinatorial optimisation: these will be cited extensively when they are introduced formally in Section 3. On the other hand, we have papers adapting classical judgment aggregation to deal with weighted issues, which we discuss here below.

Rey et al. [36] provide efficient and exhaustive embeddings of participatory budgeting problems via DNNF circuits in non-weighted judgment aggregation, giving an initial axiomatic study of *asymmetric* additive rules extended from known judgment aggregation rules. Chingoma et al. [8] simulate multi-winner voting rules in judgment aggregation for both ordinal and approval-based preferences and study their complexity. Both of these works, however, do not generalise directly to other CCO settings. Nehring and Pivato [32] introduce and study a setting of judgment aggregation with weighted issues: we use their definitions to build our general framework, including the median rule of which they give an axiomatisation.

Our paper is structured as follows. In Section 2, we give an overview of judgment aggregation rules and their generalisations to weighted asymmetric agendas, and we also generalise the Chamberlin-Courant voting rule for approval ballots to judgment aggregation. In Section 3, we show how numerous collective combinatorial optimisation settings can be expressed in weighted judgment aggregation and we prove that some of the specific rules from these settings are in fact instances of judgment aggregation rules. In Section 4 we give a computational study of the rules, both from a theoretical and an experimental point of view. We conclude in Section 5.

2 Weighted asymmetric judgment aggregation

Judgment aggregation is a general framework to make collective decisions over a set of possibly interconnected issues linked by constraints. Nehring and Pivato [32] have considered a generalisation of judgment aggregation where each issue is associated with a numerical weight, while Rey et al. [36] have defined asymmetric judgment aggregation rules. We com-

bine both approaches and consider agendas for which each item and its negation have weights (which are not required to be equal).

2.1 Formal model

A set of n agents (or voters) $\mathcal{N} = \{1, \ldots, n\}$ have to take a collective decision on the acceptance of m items (projects, issues, etc.) in an agenda A. The agenda A is composed of two sets A^+ and A^-, which represent the set of positive and negative issues, respectively. Hence, $A = A^+ \cup A^-$, where $|A^+| = |A^-| = m/2$. This means that for every $a \in A^+$ there is an $\overline{a} \in A^-$ which represents the negation of a.

Each item of the agenda A has an associated weight $w_a \in \mathbb{N}_0$ that will be used in the aggregation by the rules.[2] The weight vector $\boldsymbol{w}$ collects the weights of all the m items in the agenda A. While in the most general setting, each weight can be any natural number or 0, we are also interested in some specific restrictions on the weight vectors (which determine different types of agendas) that we describe here below.

First, we can have agendas where each positive item $a \in A^+$ and its corresponding negation $\overline{a} \in A^-$ have the same weight (what we call a *symmetric* agenda, i.e., $w_a = w_{\overline{a}}$ for each $a \in A^+$; although note that each positive-negative issue pair in the agenda may have a different weight), or agendas where the negated item has weight zero (what we call an *asymmetric* agenda, i.e., $w_{\overline{a}} = 0$ for each $\overline{a} \in A^-$). Second, we can have agendas where the items' weights are in the set $\{0, 1\}$ (what we call a *binary* agenda) or where they are in $\mathbb{N}_0$ (what we call a *weighted* agenda).

By combining the above cases, we derive the following four natural variants of judgment aggregation, defined by the corresponding restrictions on the weight vector of the agenda, which we will focus on in the rest of this paper. Note that an agenda may be neither symmetric nor asymmetric (i.e., $w_a, w_{\overline{a}} \in \mathbb{N}$, yet $w_a \neq w_{\overline{a}}$).

Standard judgment aggregation ($w_a = w_{\overline{a}} = 1$) In standard judgment aggregation, denoted by $\boldsymbol{w}_{bin}^{sym}$, all issues (and their negations) have equal weights. Formally, for any $a, a' \in A$ we have that $w_a = w_{a'}$. Without loss of generality, we can assume that all weights are equal to 1, thus yielding a *symmetric binary* agenda.

Asymmetric judgment aggregation ($w_a = 1$; $w_{\overline{a}} = 0$) For *binary asymmetric* agendas, which were studied by Rey et al. [36], and which we denote by $\boldsymbol{w}_{bin}^{asym}$, we assume that $w_a = 1$ for all $a \in A^+$ and $w_{\overline{a}} = 0$ for all $\overline{a} \in A^-$. Intuitively, when an item's weight is 0, the support for this item will be discarded when computing an outcome. Thus, in the setting by Rey et al. [36], asymmetric weights allow for rules to be biased towards the acceptance of positive agenda items.

Weighted judgment aggregation ($w_a = w_{\overline{a}} \in \mathbb{N}_0$) In *symmetric weighted* judgment aggregation, which we denote by $\boldsymbol{w}_{we}^{sym}$, different items of the agenda can have different weights, yet an item and its dual must have the same weight, $w_a = w_{\overline{a}} \in \mathbb{N}_0$. This was described by Nehring and Pivato [32].

Weighted asymmetric judgment aggregation ($w_a \in \mathbb{N}_0$; $w_{\overline{a}} = 0$) Finally, in the $\boldsymbol{w}_{we}^{asym}$ restriction we have $w_{\overline{a}} = 0$ for all $\overline{a} \in A^-$ and $w_a \in \mathbb{N}_0$ for all $a \in A^+$.

In addition to being able to capture standard judgment aggregation, the setting we provide is more general in two ways: the presence of weights and the possible asymmetry between

[2] Although Nehring and Pivato [32] assume real-valued weights, integer weights allow us to use compact languages for the constraints; a further generalisation to values in $\mathbb{R}$ (or to negative values) for the weights is also possible.

issues and their negations. The latter generalisation goes slightly in the direction of belief merging (see [28]); however, we assume that all agents report judgments on the same agenda, unlike belief merging.[3]

For an agenda A and an agent $i \in \mathcal{N}$, an agent's ballot is a vector $B_i \in \{0,1\}^m$, where each entry represents the agent's decision on a fixed agenda item. In this context, the notation $B_i(a)$ refers to the entry of vector $B_i \in \{0,1\}^m$ for issue $a \in A$. The collection of the agents' ballots is a profile $\boldsymbol{B} = (B_1, \ldots, B_n)$.

Constraints can be imposed on a weighted judgment aggregation problem, either on the collective outcome (e.g., abiding by a budget constraint) or on the individual ballots (e.g., approving a minimal number of items). Following Endriss [14] we call the former *feasibility* constraints (denoted by Γ_F) and the latter *rationality* constraints (denoted by Γ_R). Throughout the paper, we will express these constraints as sets of linear (in)equalities, allowing the constraints from many CCO settings that include numerical values to be formalised clearly.[4]

Moreover, $\mathcal{B}_R \subseteq \{0,1\}^m$ is the set of all ballots satisfying the rationality constraints, while $\mathcal{B}_F \subseteq \{0,1\}^m$ is the set of outcomes satisfying the feasibility constraints. Since agendas include an issue and its negation, we always assume that all ballots and all outcomes accept either an issue or its negation (no matter the issues' weights). Additionally, when imposing rationality and feasibility constraints, we require that any voter's ballot $B_i \in \{0,1,\}^m$ must be rational, i.e., $B_i \in \mathcal{B}_R$, and any outcome of an aggregation method $X \in \{0,1\}^m$ must be feasible, i.e., $X \in \mathcal{B}_F$.

An important remark to make here is that the weights will be used by our rules as a proxy for the agents' satisfaction during the aggregation step in order to identify the optimal outcomes. However, as we shall see for the case of participatory budgeting (in Section 3.2), in some CCO settings the weights will also determine the feasibility of the outcomes themselves: i.e., we could have a participatory budgeting scenario where the weights associated to projects correspond to their costs (thus, the satisfaction of an agent is the total cost of the accepted items they approve of), but they also appear in the constraints (i.e., the budget limit should not be exceeded).

In the following, we sometimes use $j \in [x, y]$ as a shorthand for $j \in \{x, \ldots, y\}$, and $j \in [x]$ as a shorthand for $j \in \{1, \ldots, x\}$.

2.2 Weighted asymmetric judgment aggregation rules

In this section we recap some well-studied rules from judgment aggregation, but we define them generally in the sense that the agenda items can have any weight vector (recall that, with a slight abuse of terminology, we refer to the *weighted* versions of the rules when the weights are not binary).

We call a *weighted asymmetric judgment aggregation rule* a function F that takes as input a rational profile $\boldsymbol{B} \in \mathcal{B}_R$, set of feasibility constraints Γ_F, and weight vector $\boldsymbol{w}$ for the items in an agenda A and it gives as output a set of feasible outcomes (i.e., where $X \in \mathcal{B}_F$ for each outcome X). Observe that rules are thus *irresolute*, in the sense that they may return a set of tied feasible outcomes.

[3] See [19] for a comparison of belief merging and judgment aggregation.

[4] Note that the constraints can be expressed in many ways: for instance, in the standard (symmetric) framework of binary judgment aggregation, they are usually expressed as formulas of propositional logic [23].

2.2.1 The (weighted) median rule

The *median rule* finds the outcomes that globally minimise the number of changes between the profile and the outcome or equivalently, maximise the number of agreements between the agents' ballots and the outcome. The *weighted median rule* extends the median rule by maximising the total sum of the *weights* of accepted items in the outcome and each of the agents' ballots.

Definition 1 The *(weighted) median rule* takes a rational profile $\boldsymbol{B}$, feasibility constraint Γ_F, and agenda weights $\boldsymbol{w}$ and gives a set of outcomes found by:

$$\underset{\{X|X\in\mathcal{B}_F\}}{\arg\max} \sum_{i\in\mathcal{N}} \sum_{a_j\in A} w_{a_j} \times B_i(a_j) \times X(a_j).$$

We may choose any weight vector paired with this rule, however, when paired with specific weight vectors, the function aligns with specific rules. When the agenda's weight vector is restricted to be unary (i.e., symmetric and binary, $\boldsymbol{w}^{sym}_{bin}$), this rule reflects the standard median rule (which we will refer to as *Med*); while under symmetric non-binary weights, $\boldsymbol{w}^{sym}_{we}$, we obtain the symmetric weighted median rule defined byNehring and Pivato [32] (which we will refer to as *WMed*).

Observe that *Med* and *WMed* differ in name, yet they are the same function with different weights being used for the aggregation. In the former, the weights of the items, for aggregation purposes, are binary. Hence, the satisfaction of the agents is based on the cardinality of the intersection between the outcome and voters' ballots only. Note that this does not stop the feasibility constraint to use the weights of the items when determining if an outcome is feasible. In the latter, i.e., *WMed*, the issues' weights represent the satisfaction of the voters: instead of getting one point for every item approved by both the outcome and the voters' ballots, we now get the sum of the items' weights which are approved by both.

2.2.2 The (weighted) egalitarian rule

The standard egalitarian rule outputs the outcomes that maximise the minimum number of projects approved by any agent in the outcome. When the weight vector is restricted to $\boldsymbol{w}^{sym}_{bin}$ it is called the d_H-max rule by Lang et al. [31] and the *MaxHam* rule by Botan et al. [4].

Definition 2 The (weighted) *egalitarian rule* takes a rational profile $\boldsymbol{B}$, feasibility constraint Γ_F, and agenda weights $\boldsymbol{w}$ and gives a set of outcomes found by:

$$\underset{\{X|X\in\mathcal{B}_F\}}{\arg\max} \min_{i\in\mathcal{N}} \sum_{a_j\in A} w_{a_j} \times B_i(a_j) \times X(a_j).$$

In general, we will refer to this rule as *Egal* when weights are binary and as *WEgal* when weights are non-binary ($\boldsymbol{w}^{asym}_{we}$ or $\boldsymbol{w}^{sym}_{we}$). Although egalitarian rules have also been studied in belief merging (see, e.g., [18]), to the best of our knowledge *WEgal* is new. Its motivation is natural in participatory budgeting instances with few agents, where we may want to ensure that each agent agrees with the funded projects to some minimum level.

2.2.3 The (weighted) ranked agenda rule

The ranked agenda rule was studied by Lang and Slavkovik [30] and its *leximax* refinement was studied by Nehring et al. [33]. The rule iteratively considers issues following the order

induced by the support received, rejecting an issue if its addition would break the feasibility constraint. There are two variants of the rule, depending on whether ties are broken immediately, using a tie-breaking rule, or if all tie-breaking possibilities are considered in parallel. For simplicity, we only consider the former variant.

Algorithm 1 The (weighted) ranked agenda rule.

1: **Input**: Γ_F, $\boldsymbol{B}$, A, $\boldsymbol{w}$
2: $\Theta := \{\Gamma_F\}$ and $X := \{0\}^{|A|}$
3: Order the issues of A w.r.t. $\sum_{i \in \mathcal{N}} B_i(a) \times w_a$ in descending order
4: **for** each issue a_j in the ordering **do**
5: **if** Θ and a_j is consistent **then**
6: $X(a_j) := 1$ and $\Theta := \Theta \cup \{a_j\}$
7: **end if**
8: **end for**
9: **return** X

The rule follows Algorithm 1. It has as input the constraint Γ_F, the profile of ballots $\boldsymbol{B}$, the agenda A, and the weight vector $\boldsymbol{w}$; the output is an outcome $X \in \mathcal{B}_F$. On line 2, Θ is initialised to be a set containing Γ_F and the outcome vector X sets a 0 for each item. It then orders the items in A with respect to their weighted support (using a linear tie-breaking rule when items have equal support). Following this order of items, in the for-loop on line 4, it first checks if the addition of this item breaks feasibility given the currently accepted items. If feasibility is respected, then the outcome for that item is set to 1 and the item is added to Θ (otherwise, its negation will be in the outcome). This algorithm can be altered to get other ranked rules, such as the greedy Chamberlin-Courant rule (see [41]).

For a non-binary agenda, we will refer to this rule as *WRank* and for a binary agenda we will refer to it as *Rank*.

2.2.4 The Chamberlin-Courant rule

The *Chamberlin-Courant voting rule* (CC) was originally introduced for ordinal preferences by Chamberlin and Courant [7] as a way to try to ensure that all voices are present in deliberation. We consider a variant for approval-based preferences, studied by Skowron and Faliszewski [37] for multi-winner elections and generalised to participatory budgeting by Talmon and Faliszewski [41].[5] For approval-based preferences, an outcome should maximise the number of agents who have at least one item in the outcome that they approve of. We define the rule here for general agendas: we say that an agent is *satisfied* by an outcome if there is at least one issue with non-zero weight approved by the agent and contained in the outcome; then, the rule outputs every outcome that maximises the number of satisfied agents.

Definition 3 The *Chamberlin-Courant* rule takes a rational profile $\boldsymbol{B}$, feasibility constraint Γ_F, and agenda weights $\boldsymbol{w}$ and gives a set of outcomes found by:

$$CC(\boldsymbol{B}, \Gamma_F, \boldsymbol{w}) = \underset{\{X | X \in \mathcal{B}_F\}}{\arg\max} \sum_{i \in \mathcal{N}} \min(1, \sum_{a_j \in A} X(a_j) \times B_i(a_j) \times w_{a_j}).$$

[5] In a related approach, Chingoma et al. [8] generalised the Chamberlin-Courant rule and the proportional approval voting rule (PAV) to judgment aggregation, relying on weak rankings to model either dichotomous or ordinal preferences.

First, note that changing the weight of an issue from some non-zero number to another non-zero number has no impact on the outcome: only changing non-zero weights to zero and vice-versa can affect the outcome (without loss of generality non-zero weights can be 1). Second, the rule makes particular sense for asymmetric agendas, such as those for participatory budgeting or multi-winner elections.

2.3 Examples

We now provide an example illustrating the four rules we just introduced, both in their weighted and binary versions.

Let $\mathcal{N}$ be a set of 8 voters and $A^+ = \{i, j, k, \ell, m, n\}$ be a set of 6 (positive) issues. Each positive issue has an associated cost, modelled by vector $c = (1, 2, 2, 2, 3, 4)$. In this example, costs are used to determine if an outcome is feasible. Moreover, when the weights are non-binary, they will correspond to the issues' costs. We will only consider an asymmetric agenda, thus, the weight vectors will be restricted to be of the form $\boldsymbol{w}_{bin}^{asym}$ or $\boldsymbol{w}_{we}^{asym}$. Hence, for each $\overline{a} \in A^-$, $w_{\overline{a}} = 0$.

The feasibility constraint states that the total cost of the collectively accepted issues is no greater than 5. We first study the profile $\boldsymbol{B}$ given in Table 1. The line labelled *sum* gives the total support for each of the issues in the profile, while the line labelled *weighted sum* gives the total support for that issue times its weight.

Binary asymmetric weights We first consider the aggregation weights to be binary and asymmetric. Thus, the aggregation weight vector $\boldsymbol{w}$ is such that $w_a = 1$ for $a \in A^+$ and $w_a = 0$, otherwise. Given $\boldsymbol{w}$, we compute the rules *Med*, *CC*, and *Rank*: each of them considers the amount of support each item of A^+ has received, as per the *sum* line in the table.

The unique outcome of *Med* is accepting only i, j, and k as it has the highest total support of 13 and all other feasible outcomes have smaller total support.

The *Rank* rule returns the outcome that accepts only i and m. It orders the issues with respect to how much support they have received: i.e., i, m, j, k, n, ℓ (breaking ties alphabetically); then, it accepts i and then m as their total cost is 4. None of the remaining issues can be added without exceeding the budget limit.

The *CC* rule returns the outcome that only accepts ℓ and m as this is the only feasible outcome in which all voters approve of at least one of the issues.

Weighted asymmetric weights We now consider a weighted asymmetric vector $\boldsymbol{w}'$, where $w'_a = c(a)$ for all $a \in A^+$, giving $\boldsymbol{w}'^+ = (1, 2, 2, 2, 3, 4)$, i.e., the weight of each issue is its cost, and each issue in A^- has a weight of 0. The outcomes of the rules *WMed* and *WRank* on $\boldsymbol{w}'$ are given in Table 1 (we do not consider the *CC* rule on non-binary weights).

The rule *WMed* returns two tied outcomes that maximise the total weighted support. The first outcome accepts only issues j, and m, while the second accepts only issues k and m: both have a total weighted support of 23.

The rule *WRank* first orders the issues with respect to their weighted support (the numbers in the *weighted sum* line), giving the order: n, m, j, k, ℓ, i (ties are broken alphabetically). Then, it accepts issue n with cost 4. It then must reject m, j, k, and then ℓ, in this order, as their acceptance would exceed the limit of 5 given by the constraint. Then issue i is added, giving the outcome where only n and i are accepted.

Egalitarian rules For the egalitarian rules, we consider a different profile $\boldsymbol{B}'$ given in Table 2 with $\mathcal{N} = [6]$, where we consider the same issues as before except j, i.e., $A^+ \backslash \{j\}$. Our weight

Table 1 Profile $\boldsymbol{B}$ from the example given in Section 2.3

	i	j	k	ℓ	m	n
cost	1	2	2	2	3	4
voter 1	1	0	1	0	1	0
voter 2	0	1	1	1	0	1
voter 3	1	1	1	0	1	1
voter 4	1	1	0	0	1	1
voter 5	1	1	0	1	0	1
voter 6	1	0	1	0	1	0
voter 7	0	0	0	1	0	0
voter 8	0	0	0	0	1	0
sum	5	4	4	3	5	4
Med	1	1	1	0	0	0
CC	0	0	0	1	1	0
Rank	1	0	0	0	1	0
weighted sum	5	8	8	6	15	16
WMed	0	1	0	0	1	0
	0	0	1	0	1	0
WRank	1	0	0	0	0	1

We give the total number of approvals for each issue (*sum*), the weighted number of approvals (*weighted sum*), and the outcome of the different rules (including possible ties, as seen for *WMed*)

vectors remain the same excluding the entry corresponding to j. The cost of each issue is given in $c' = (1, 2, 2, 3, 4)$, and the budget constraint remains at 5.

First consider *Egal*, where the weight vector has $w_a = 1$ if $a \in A^+\backslash\{j\}$, and $w_a = 0$ if $a \in A^-\backslash\{\overline{j}\}$. In this instance, the outcomes of *Egal* contain, for each voter, at least one issue

Table 2 Profile $\boldsymbol{B}'$ from Section 2.3 used to show the outcome(s) of *Egal* and *WEgal*

	i	k	ℓ	m	n
cost	1	2	2	3	4
voter 1	1	1	0	1	0
voter 2	0	1	1	0	0
voter 3	0	1	0	1	1
voter 4	1	0	0	1	1
voter 5	1	0	1	0	1
voter 6	1	1	0	1	0
Egal	1	1	1	0	0
	1	1	0	0	0
	0	0	1	1	0
WEgal	0	0	1	1	0

that they approve of. In particular, each of the 6 voters approves of either issue ℓ or m, so *Egal* returns the outcome approving these two issues (no more items can be added without exceeding the budget limit). Then, there is only one feasible outcome accepting three issues from the positive agenda: i, k and ℓ. Here the least satisfied voters (voters 3 and 4) approve exactly one issue in the outcome. Note that the outcome accepting only issues i and k is returned by *Egal*, as the least satisfied voter (voter 3) also approves of only one issue.

As *WEgal* is a weighted judgment aggregation rule (i.e., using a non-binary weight vector), we let the issues' weights be their cost. Thus, $w_a = c(a)$ for $a \in A^+\backslash\{j\}$ and $w_a = 0$ if $a \in A^-\backslash\{\overline{j}\}$. Then, only the outcome accepting issues ℓ and m is returned by *WEgal*, as every agent gets at least a minimum weight of 2 (out of 5) that they approve of.

3 Collective combinatorial optimisation: five problems

In this section, we present five examples of collective combinatorial optimisation (CCO) problems from the literature, and we model them in judgment aggregation. A CCO problem has the following characteristics. First, a collective decision must be taken to decide which discrete items should be accepted from a given finite set. In many examples of CCO settings, each of the items will have a cost, which may be used by the constraints. In general, the combinatorial nature of the problem then arises from the presence of constraints specifying what is a feasible outcome. CCO rules will then take a rational profile of ballots and return a feasible collective combinatorial outcome, optimising some metric of satisfaction. Note that rationality and feasibility constraints here correspond to the specifics of the CCO setting in question.

We show that specific rules, studied independently within each CCO setting, are instances of the rules defined in Section 2.2. Formally, a CCO rule $\mathcal{R}$ *is an instance of* a judgment aggregation rule $\mathcal{R}'$ if there is a translation from any profile $\boldsymbol{B}$ of this specific CCO setting into a judgment aggregation profile $\boldsymbol{B}'$ such that the outcome of $\mathcal{R}$ on $\boldsymbol{B}$ is equivalent to the outcome of $\mathcal{R}'$ on $\boldsymbol{B}'$, i.e., they correspond to the same collective decision.

3.1 Multi-winner elections

This well-studied framework models the collective selection problem of a set of candidates. The candidates have equal weight, which can be assumed to be unitary without a loss of generality. The agenda is $A = \{a, \overline{a} \mid a \text{ is a candidate}\}$. Furthermore, we mainly consider CCO rules in which the agents only vote on the acceptance of the candidates, the exception being the *minimax* approval rule (details are given in the following). Thus, the agenda is binary and asymmetric, i.e., for each pair $a, \overline{a} \in A$, $w_a = 1$ and $w_{\overline{a}} = 0$. We thus only consider the judgment aggregation rules with binary weights. The feasibility constraint requires that exactly a given number $k \in \mathbb{N}$ of candidates are elected, thus $\mathcal{B}_F = \{X \mid \sum_{a\in A} X(a) \times w_a = k\}$. The ballots represent the agents' approval of the candidates: they can approve as many candidates as they like, or exactly k candidates ($\mathcal{B}_R = \mathcal{B}_F$).

The following proposition shows the correspondence between rules in the literature on multi-winner voting and our general judgment aggregation rules. Definitions of multiwinner voting rules can be found in the work of Lackner and Skowron [29].

Proposition 1 *The following multi-winner rules are instances of their judgment aggregation counterparts:*

i) the standard multi-winner approval voting rule, that outputs the most approved k candidates, is an instance of both Med and Rank (modulo tie-breaking);
ii) the Chamberlin-Courant rule for approval ballots (see [37]) is an instance of CC;
iii) the minimax approval voting rule by Brams et al. [5], that outputs the outcomes minimising the maximum, over all agents i, of the Hamming distance between the outcome and i's ballot, is an instance of Egal with binary symmetric weights ($\boldsymbol{w}^{sym}_{bin}$).[6]

Proof For asymmetric agendas equipped with a multi-winner constraint, the median rule *Med* selects all subsets of candidates that maximise the (sum of the) overlap of voters' approved issues with the outcome. This is exactly what the *multi-winner approval voting rule* does. The ranked agenda rule *Rank* sorts the issues by voters' support first and then sequentially adds the best k issues to the outcome. Considering parallel-universe tie-breaking, we derive exactly those outcomes that are output by the median rule.

As the *Chamberlin-Courant rule for approval ballots* was adapted from Skowron and Faliszewski [37] into our judgment aggregation framework, the instantiation follows by design. Both in multi-winner elections and judgment aggregation (equipped with a multi-winner constraint), the *Chamberlin-Courant rule* selects those fixed-size subsets of issues that maximise the number of voters that approve of at least one (positive) issue.

The *minimax approval voting rule* by Brams et al. [5] selects those outcomes that minimise the maximum Hamming distance to the voters' approval ballots. Botan et al. [4] point out that their judgment aggregation rule *MaxHam* generalises the minimax approval voting rule. Note that maximising the minimum support instead of minimising the maximum lack of support is only a difference in modelling. In particular, as we consider symmetric agendas, each agent supports exactly half of the issues (counting rejections). Thus, the Hamming distance between a voter's ballot and an outcome can be derived from counting the supported issues, subtracting $m/2$, and multiplying by -1 (and swapping minimisation and maximisation operators due to the sign change). □

The egalitarian multi-winner rule appears to be novel—although a recent paper on participatory budgeting proposed an asymmetric egalitarian rule [40]. The rule outputs the committees of k candidates that maximise the minimum number of committee members approved by any agent. It is close to the rules studied by Aziz et al. [2], who however consider ballots to be rankings over alternatives.

3.2 Participatory budgeting

Participatory budgeting (PB) is a class of collective selection problems, generalising multi-winner elections, where the agents approve projects to be funded by a limited resource (e.g., a monetary budget). A PB problem consists of a set of projects P, and each $p \in P$ has a cost c_p if implemented. The PB rule will return a set of selected projects with a total cost that must not exceed the budget limit $\ell \in \mathbb{N}$.

We will now rephrase the PB problem in terms of our judgment aggregation notation introduced in Section 2.1. The selection agenda A contains issues for each project $p \in P$ represented by a_p. The agenda has asymmetric weights, i.e., $w_{\overline{a}} = 0$ for every project $a \in A$. We focus on the case where the agenda weights are either binary ($w_p = 1$ for each $a_p \in A^+$) or a weighted agenda (where $w_p = c_p$), depending on the notion of satisfaction required by the rule.

[6] The link between minimax approval voting and judgment aggregation was discussed by Grossi and Pigozzi [24].

Note that in participatory budgeting, the feasibility of the outcome is always determined by the projects' costs. The weights, as mentioned in Section 2, are instead used in the aggregation to quantify the voters' satisfaction.

Regarding the feasibility of the outcomes, the cost of the collectively selected projects must not exceed the budget limit $\ell \in \mathbb{N}$. Hence,

$$\mathcal{B}_F = \{X \mid \sum_{a \in A^+} X(a) \times c_a \leq \ell\}$$

are the feasible outcomes, with A^+ the positive issues of A.

If there are no rationality constraints, $\mathcal{B}_R = \{0, 1\}^m$, agents can approve any number of items (regardless of their costs); if $\mathcal{B}_R = \mathcal{B}_F$, agents must submit their ideal allocation under the budget limit. Related problems include collective knapsack or knapsack voting [22], where rationality and feasibility constraints coincide, and weighted committee selection [27].

Proposition 2 *The following PB rules are instances of their judgment aggregation counterparts:*

i) *the max rule with cardinality satisfaction from Talmon and Faliszewski [41] is an instance of Med;*
ii) *the generalised approval-based Chamberlin-Courant rule by Talmon and Faliszewski [41] is an instance of CC;*
iii) *the max rule with cost satisfaction from Talmon and Faliszewski [41] is an instance of WMed;*
iv) *the greedy rule with cardinality satisfaction by Talmon and Faliszewski [41] is an instance of Rank;*
v) *the greedy rule with cost satisfaction from Talmon and Faliszewski [41] is an instance of WRank;*
vi) *the maxmin participatory budgeting rule from Sreedurga et al. [40] is an instance of WEgal;*
vii) *the individually best knapsack rule from Fluschnik et al. [20] is an instance of Med when their utilities are binary;*
viii) *the diverse knapsack rule from Fluschnik et al. [20] is an instance of CC when their utilities are binary.*

Proof We begin with the rules by Talmon and Faliszewski [41]. The *max rule with cardinality satisfaction* and the *generalised approval-based Chamberlin-Courant* respectively generalise the multi-winner rules *approval voting rule* and *Chamberlin-Courant rule for approval ballots*, with the only difference that feasibility is determined by a participatory budgeting constraint. Following Proposition 1, the rules are instances of *Med* and *CC*.

In participatory budgeting, a voter's cost-based satisfaction corresponds to the funds spent on projects the voter approves of. The *max rule with cost satisfaction* selects those feasible bundles that maximise the (sum of) voters' cost-based satisfaction. This is exactly what *WMed* for weighted asymmetric agendas does when the weights model the respective projects' costs.

The *greedy rule with cardinality (resp. cost) satisfaction* constructs an outcome sequentially. The issues are first ranked either by the sum of cardinality satisfaction (i.e., the number of supporting voters) or the sum of cost satisfaction, then projects are selected by descending total satisfaction (with some tie-breaking rule), skipping projects that would break the feasibility constraint. Translated to judgment aggregation with asymmetric agendas, the voters' satisfaction with each issue is preserved. Assuming the same tie-breaking scheme is used, *(W)Rank* ranks the projects in the same way as the *greedy rule with cardinality (cost)*

satisfaction and issues are added to the outcome sequentially, skipping issues that break feasibility.

The *maxmin participatory budgeting rule* from Sreedurga et al. [40] also assumes cost-based voters' satisfaction. The rule selects those outcomes that maximise satisfaction for the least satisfied voter, analogously to *WEgal*.

For the remaining two rules by Fluschnik et al. [20], the voters assign each item a non-negative integer utility. Assuming these utilities are binary, each item has an individual voter's utility of zero or one. Given a participatory budgeting constraint, the *individually best knapsack rule* selects a subset of items that maximises the (sum of the) voters' utilities in an analogous way to *Med*. In contrast, the *diverse knapsack rule* maximises the number of voters that have a non-zero utility (for non-binary utilities every voter is represented by the utility of her most preferred item in the outcome). For binary utilities, this corresponds to *CC*. □

3.3 Collective networking

In the problem of collective networking, the agents have to design a common network—whether the network consists of water pipelines, internet services, or travel connections between countries. The agents specify which links they approve of, and the goal is to find a spanning tree from such input, i.e., an undirected acyclic graph that includes all nodes, maximising the satisfaction of the agents. This problem has been introduced and studied by Darmann et al. [10, 11].

Given an undirected network $G = (V, E)$ a networking agenda is the set of items $A = \{a_{ij}, \overline{a_{ij}} \mid (i, j) \in E\}$, where $w_{\overline{a_{ij}}} = 0$ for all $\overline{a_{ij}} \in A^-$ (i.e., the agenda has asymmetric weights). Then, c_{ij} is the cost of adding edge (i, j) to the outcome network. Darmann et al. [11] consider edges with costs but no budget limit determining what is a feasible outcome, as they assume that some central authority will fund any outcome. As for participatory budgeting, we can consider either $w_{a_{ij}} = c_{ij}$ or $w_{a_{ij}} = 1$, depending on how the rule is going to model the agents' satisfaction for building such a connection.

The set of accepted edges must form a spanning tree (i.e., acyclic and connected tree), this is reflected in the feasibility constraints—and a budget limit can also be imposed. These constraints can be formulated as linear inequalities in many ways.[7] We here focus on the single commodity flow model by Abdelmaguid [1], where we first move from undirected to directed graphs, and we then forget the direction of the edges to obtain the collective spanning tree. We have $|E|$ variables a_{ij} stating whether (i, j) is in the collective spanning tree, and $2|E|$ variables y_{ij} and y_{ji} in set Y for the two directions of each edge in E. Each $y_{ij} \in \mathbb{N}$ describes the flow going from node i to node j.

We also have $|V|$ constraints as follows, for $j \in V$:

$$\sum_{i:(i,j)\in E} (y_{ij} - y_{ji}) = \begin{cases} 1 - |V|, & \text{if } j = 1 \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

The first case accounts for the (artificial) root of the tree $j = 1$, having no in-flowing edges. Thus, $y_{i1} = 0$ for all $(i, 1) \in E$ and the out-flowing edges have a total weight of $|V| - 1$. The second case ensures that in a spanning tree, the in-flowing weight exceeds the out-flowing by one.

[7] Note that without the use of linear inequalities, constraints can still be expressed compactly (only adding a polynomial number of variables) when weights are unary (see [25]).

Next, we ensure that directed edges correspond to the undirected edges:

$$y_{ij} \leq (|V| - 1)x_{ij} \quad \text{and} \quad y_{ji} \leq (|V| - 1)x_{ij}. \tag{2}$$

For every $(i, j) \in E$, the constraints impose that each of y_{ij} and y_{ji} can carry flow only when x_{ij} is in the spanning tree. Finally, the tree must have $|V| - 1$ edges:

$$\sum_{(i,j)\in E} x_{ij} = |V| - 1. \tag{3}$$

Example 1 Four cities want to improve their train connections (illustrated in Fig. 1) and need to decide which rails should become high-speed. Let $G = (V, E)$ represent the cities and candidate rails, where $V = \{h, i, j, k\}$ and $E = \{(h, i), (h, k), (i, j), (i, k), (j, k)\}$. Thus, the agenda is $A^+ = \{a_{hi}, a_{hk}, a_{ij}, a_{ik}, a_{jk}\}$, and $(1, 2, 4, 3, 2)$ are the respective costs to build such connections, in millions of euros. Assuming the cities themselves may submit a preference on which connections to upgrade, h may want a faster connection with cities i and k, where many of its citizens work, thus submitting ballot $B_h = (1, 1, 0, 0, 0)$. A solution would then need to choose which connections to improve, ensuring that all cities are connected by high-speed rails (possibly abiding by a budget if previously specified).

We now show that the specific rules introduced for the collective network problem are all instances of judgment aggregation rules.

Proposition 3 *The following collective network rules are instances of their judgment aggregation counterparts:*

i) *the maximum collective spanning tree from Darmann et al. [10] is an instance of Med;*
ii) *the maximin voter satisfaction problem for approval voting for spanning trees from Darmann et al. [11] is an instance of Egal;*
iii) *the greedy algorithm for the maximum spanning tree problem from Escoffier et al. [17] is an instance of Rank, when restricted to approval ballots.*

Proof We now show that the procedure to find the maximum collective spanning tree from Darmann et al. [10] is an instance of *Med*. Finding the maximum collective spanning tree equates to finding the spanning tree that maximises the total support. Support here is determined by the number of agents who vote on an edge in the initial graph which is included in the spanning tree. We see that *Med* gives the same solution, as it returns a feasible outcome (in the case of spanning trees, with respect to (1), (2) and (3)) such that the total support of the issues (i.e., total support on the edges) is maximised. Thus, the maximum collective spanning tree from Darmann et al. [10] is an instance of *Med*.

We then show that the maximin voter satisfaction problem for approval voting as defined by Darmann et al. [11] is an instance of *Egal*. Its scoring function for approval voting assigns

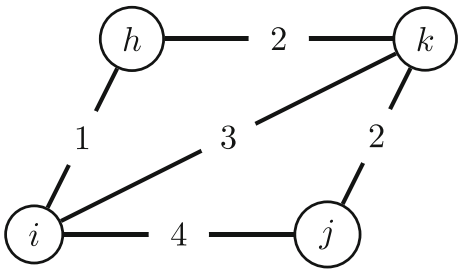

Fig. 1 Graph given in Example 1. Each node represents a city, each edge is an existing train connection, and an edge's weight is the cost to upgrade a high-speed connection

a point for each edge a voter approves of in a given spanning tree. Their egalitarian operator finds the spanning trees which maximise the minimum approval score of any agent. This coincides with the formula given in Definition 2 with binary asymmetric weights.

We next show that the greedy algorithm for the maximum spanning tree problem from Escoffier et al. [17] is an instance of *Rank*, when restricted to approval ballots. The algorithm provided by Escoffier et al. [17] orders the items by their approval level by the scoring function used. The items are added in this order only if they do not break feasibility. When the setting is restricted to approval voting, i.e., the voters can give only a valuation of 1 or 0 to every edge, this corresponds to the number of agents in support of an edge. Therefore, given a profile of votes, both the maximum spanning tree problem from Escoffier et al. [17] and *Rank* create the same ordering of the items (provided that they follow the same tie-breaking rule). As in Algorithm 1, the items are added with respect to this ordering and are only rejected when their addition would entail that the resulting graph is not a spanning tree (as per (1), (2) and (3)). Hence, the greedy algorithm for the maximum spanning tree problem from Escoffier et al. [17] is an instance of *Rank*. □

3.4 Collective scheduling

Let $P = \{p_1, \ldots, p_m\}$ be a set of (at least two) jobs to be performed on a single machine, with execution time t_x for job $p_x \in P$. The agents submit transitive and asymmetric orderings over P, indicating their preferred order of execution of the jobs, which is then decided by a collective rule. Pascual et al. [35] assume that the output schedule has no gaps and is complete (hence, $\mathcal{B}_R = \mathcal{B}_F$): the setting is thus equivalent to the aggregation of orderings of alternatives, where the alternative can have different durations (similarly to costs for participatory budgeting).[8]

Let A with $A^+ = \{a_{x \prec y}, a_{y \prec x} \mid p_x, p_y \in P\}$ be a scheduling agenda, where $a_{x \prec y}$ being accepts represents the support of p_x being scheduled before p_y, whereas $a_{y \prec x}$ represents support of p_y being scheduled before p_x. For an agent i, who provides a complete ranking over the projects, it holds that $B_i(a_{y \prec x}) = 1 - B_i(a_{x \prec y})$, and similarly for their negations. The agenda items are either weighted or binary and are usually asymmetric. We focus here on the binary asymmetric setting, where the weights of all of the items in A^+ are set to 1, and those in A^- are 0. In the weighted setting, many different asymmetric weight vectors could be considered, e.g., weights corresponding to the jobs' durations (see the discussion at the end of this section).

Agents submit complete rankings, so their ballots must approve exactly half of the positive agenda items in A^+, i.e., either $a_{x \prec y}$ or $a_{y \prec x}$ for all $\{p_x, p_y\} \subseteq P$. For the full agenda A, each agent approves half of the items (i.e., an item or its negation), even in case we want to model a voter submitting an incomplete schedule.

The outcome X of the collective scheduling problem must be a linear order of the jobs: thus, the feasibility constraints must impose transitivity and asymmetry of scheduled jobs. These can be easily formulated as linear inequalities.

Example 2 A faculty is scheduling the mandatory courses $P = \{p_1, p_2, p_3, p_4\}$ for the first-year students, and the faculty members have to decide on their ordering. Then the positive agenda is given by $A^+ = \{a_{1 \prec 2}, a_{2 \prec 1}, a_{1 \prec 3}, a_{3 \prec 1}, a_{1 \prec 4}, \ldots\}$. Professor i thinks that p_2 should come first, then p_1 second, p_3 third, and that p_4 should come last. Hence, i approves of all

[8] The connection between the aggregation of (preference) orderings over alternatives without durations and judgment aggregation is well-known, and thus known how the rules in both settings correspond to each other Endriss [13].

items in $\{a_{2\prec x}, a_{\overline{x\prec 2}} \mid p_x \in \{p_1, p_3, p_4\}\} \cup \{a_{1\prec x}, a_{\overline{x\prec 1}} \mid p_x \in \{p_3, p_4\}\} \cup \{a_{3\prec 4}, a_{\overline{4\prec 3}}\}$, and reject the remaining items.

Proposition 4 *The following collective scheduling rules are instances of their judgment aggregation counterparts:*

i) *the utilitarian aggregation rule with swap distance from Pascual et al. [35] is an instance of Med;*

ii) *the egalitarian aggregation rule with swap distance from Pascual et al. [35] is an instance of Egal.*

Proof These rules consider the weight vector to be binary and asymmetric for the items in A.

Pascual et al. [35] already make the connection between the Kemeny rule and their utilitarian aggregation rule with swap distance, which is known to be equivalent to *Med*.

As for the egalitarian aggregation function with swap distance, this rule gives a score for each agent $i \in \mathcal{N}$ and possible outcome, which is one negative point for each item that an agent wanted and is not accepted. By a slight abuse of notation, we can define for agent $i \in \mathcal{N}$ this score as $\sum_{a_j \in A} -B_i(a_j) \times (1 - X(a_j))$, where $B_i(a_j) \in \{0, 1\}$ represents if i accepts issue a_j or not, and $X(a_j)$ represents if a_j is accepted or not in possible outcome X.

The rule returns those outcomes such that the minimum value Z of this score for any agent (i.e., the value such that all agents have at least this score) is maximal. All agents approve the same number of items in collective scheduling, i.e., $k = \sum_{a \in A} B_i(a)$ for every $i \in \mathcal{N}$. The value of Z can be at worst the number of approvals in any ballot and at best 0. Since every voter approves exactly k items of the agenda, we can add k to every agent's summation, $k + \sum_{a_j \in A} -B_i(a_j) \times (1 - X(a_j)) \geq Z + k = Z'$, for $Z' \in [0, k]$. By rearranging this inequality, we see that this is equivalent to $\sum_{a_j \in A} X(a_j) \times B_i(a_j)$ for $i \in \mathcal{N}$, which is equivalent to maximising the least number of agreements between each agent's ballot and the outcome, which is exactly what *Egal* does. □

Pascual et al. [35] also study, among others, a *tardy measure* of satisfaction, which paired with their utilitarian or egalitarian rules resemble *WMed* and *WEgal*, respectively. However, they are not instances of our judgment aggregation rules, as they rely on information specific to each agent: namely, a *due date* which may be different for each agent (thus requiring an individual weight function). Although these kinds of rules cannot be modelled in weighted judgment aggregation, and since each item of a scheduling agenda refers to *a pair* of jobs (to their preferred ordering), one way to incorporate non-binary weights would be to assign the duration of the second job as the weight of the item: i.e., the weight of $w(a_{x\prec y}) = t_y$ and $w(a_{\overline{x\prec y}}) = 0$.[9]

Moreover, if agents are allowed to submit partial schedules, we could integrate this into our model by letting them disapprove both the four elements of a pair of projects $a_{x\prec y}, a_{y\prec x}, a_{\overline{x\prec y}}, a_{\overline{y\prec x}}$. This could allow for the modelling of being indifferent between the ordering of p_x and p_y.

3.5 Collective placement

Not only does our general framework allow us to give a unifying treatment to many known problems, but it can also be used to tackle novel applications. Due to the applications being

[9] Note that this would also require adding extra items to the agenda representing the starting job in the ordering, e.g., $a_{0\prec x}, \overline{a_{0\prec x}}$ for each project $p_x \in P$, in order to take into account the duration of this starting job as well.

novel, they have not been studied independently and thus do not currently have a specific framework. Consider a situation in which agents collectively decide both which projects to accept and also how a given resource will fund them: e.g., selecting events and the rooms where they can take place, pieces of furniture to purchase and where to place them in a room, which tasks to accomplish and by which worker, and so on. The items in a *placement agenda* thus represent whether a project is funded and by which part of the resource.

Formally, let $P = \{p_1, \ldots, p_m\}$ be a set of projects (e.g., events, tasks), where $p_i \in P$ requires $c_i \in \mathbb{N}^+$ parts of a given resource (e.g., rooms, workers) to be implemented. Given a resource divisible into ℓ separate parts, a placement agenda contains a total of $\ell \cdot \sum_{p_i \in P} c_i$ elements, where each item $a^k_{i,j} \in A^+$ can be read as funding project p_i's j^{th} part with the k^{th} part of the resource. Thus, for each $p_i \in P$, for all $k \in [\ell]$ and each $j \in [c_i]$ we have $a^k_{i,j} \in A^+$, where $w_{\overline{a}} = 0$ for all $\overline{a} \in A^-$.

Feasibility constraints impose that a resource part is only used once in an outcome X (5), that a project part is only funded once (4), and either every part of the project is funded or none of it is (6).

Formally, for each $p_i \in P$ and $j \in [c_i]$ we introduce:

$$\sum_{k=1}^{\ell} X(a^k_{i,j}) \leq 1 \tag{4}$$

For each $k \in [\ell]$ we have:

$$\sum_{p_i \in P} \sum_{j=1}^{c_i} X(a^k_{i,j}) \leq 1 \tag{5}$$

Furthermore, we introduce binary variables p_i that evaluate to one when project p_i has been accepted and zero otherwise. Therefore, for each $p_i \in P$:

$$\sum_{j=1}^{c_i} \sum_{k=1}^{\ell} X(a^k_{i,j}) = c_i \cdot p_i \tag{6}$$

These constraints define a basic setup where projects may not be funded by consecutive parts of the resource.

Example 3 A company is refurbishing the floor of one of its buildings. Ten rooms will be built with the following constraints: 6 rooms for office space, at most two toilets, at most one common room and at most 5 meeting rooms (but possibly none). The employees are asked to vote on possible refurbishment plans. The resource is thus composed of 10 discrete blocks (the rooms), and the corresponding placement agenda items are $a^k_{office,j}$ for $j \in [6]$, $a^k_{common,j}$ for $j = 1$, $a^k_{toilets,j}$ for $j \in [2]$, $a^k_{meeting,j}$ for $j \in [5]$, with $k \in [10]$. An employee whose favourite floor plan is to put the six offices together first, then a toilet, and then 3 meeting rooms will thus vote $a^k_{office,k} = 1$ for $k \in [6]$, $a^7_{toilets,1} = 1$, $a^{7+j}_{meeting,j} = 1$ for $j \in [3]$, and also vote 0 on all other issues (and 1 on their negations), signalling also that they are not interested in having a common room on the floor. Of course, a nice user interface could help to visualise a vote and translate it into a ballot for this placement agenda.

4 Identifying computational complexity gaps

In the previous section, we saw that our general judgment aggregation framework is expressive enough to capture a variety of different settings. Therefore, it is a good candidate as a declarative language for collective combinatorial optimisation problems because it describes the problems in logical terms, and their algorithmic resolution is taken care of by a general solver. Moreover, we showed that many domain-specific CCO rules across the literature can be simulated by known judgment aggregation rules—either directly or with a slight generalisation to weighted and/or asymmetric agendas.

Shifting to a general framework allows for the use of additional constraints on the output to study variants of classical problems without modelling a new framework. As an example, Rey et al. [36] showed that by modelling participatory budgeting in judgment aggregation they can consider multiple resources, project dependencies or quotas on project types. Yet, this shift may come with a computational complexity gap. When there is such a gap, large instances may not become easily solvable when expressed directly in the general language using a general solver (yet, they could still be solved approximately; and small instances can be solved easily in spite of the gap). On the other hand, if there is no such gap, then generality comes for free.

This section is devoted to comparing the complexity of outcome determination for judgment aggregation rules and domain-specific CCO rules. Endriss et al. [16] give a complexity overview for standard (binary, symmetric) rules. We complement their results by classifying the complexity of the weighted counterparts of these rules and identifying the presence or absence of complexity gaps to outcome determination for multi-winner elections and participatory budgeting. We do not consider collective scheduling and collective network design here, as few complexity results are known.

We focus our study of outcome determination on the credulous (i.e., existentially quantified) version of the decision problem—the results for the skeptical (i.e., universally quantified) variants are similar, replacing classes by their co-class.

F-Credulous-Outcome-Determination (F-Cred)	
Given:	An agenda A with associated weight vector $\boldsymbol{w}$, a set of rationality and feasibility constraints modelling $\mathcal{B}_R$ and $\mathcal{B}_F$, a profile $\boldsymbol{B} \in {\mathcal{B}_R}^n$, and a distinct issue from the agenda $a^* \in A$.
Question:	Is there a feasible outcome $B \in F(\boldsymbol{B}, \Gamma_F, \boldsymbol{w})$, such that $B(a^*) = 1$?

We are now set to discuss the complexity of computing the outcome of the rules we introduced in Section 2.2, both in the general case and in two restricted cases of participatory budgeting and multi-winner elections. Table 3 summarises our results. We do not consider the complexity in the weighted setting for multi-winner elections since the weights for candidates are binary (by definition of the setting). Our findings show that for *WMed*, *WEgal* and *CC* (and we conjecture also for *Egal*), there is no increase in complexity when moving from specific CCO problems to our general formulation. This is not true for *Med* and *Rank*, whose application to participatory budgeting or multi-winner elections can be run in polynomial time. Interestingly, the restriction of the median rule *Med* to constraints whose consistency

Table 3 In this table we compare each of the rules used throughout the paper with respect to different settings: namely, judgment aggregation, multi-winner elections, and participatory budgeting

Rule	Judgment Aggregation with arbitrary weights in $\mathbb{N}_0$	Multi-winner Elections	Participatory Budgeting
Med	Θ_2^{p}-c$^{\blacklozenge}$	P	P$^{\spadesuit,\blacktriangle}$
WMed	Δ_2^{p}-c$^{\blacktriangle\oplus}$	-	Δ_2^{P}-c$^{\blacktriangle}$
Egal	Θ_2^{p}-c$^{\blacklozenge}$	coNP-h$^{\ominus}$ $\in \Theta_2^{\mathrm{P}\ominus}$	coNP-h$^{\ominus}$ $\in \Theta_2^{\mathrm{P}\ominus}$
WEgal	Δ_2^{P}-c$^{\blacktriangle\oplus\ominus}$	-	Δ_2^{P}-c$^{\blacktriangle\oplus\ominus}$
(W)Rank	Δ_2^{P}-c$^{\clubsuit}$	P	P
CC	Θ_2^{P}-c$^{\heartsuit}$	Θ_2^{P}-c$^{\heartsuit}$	Θ_2^{P}-c$^{\heartsuit}$

For each rule and setting, we give the computational complexity of the (credulous) outcome determination decision problem. $^{\clubsuit}$[15]; $^{\blacklozenge}$[16]; $^{\spadesuit}$[41]; $^{\heartsuit}$[39]; $^{\blacktriangle}$[3]—hardness proof for *WMed* in PB holds for one voter, the extension of this result to other settings can be seen in $^{\oplus}$Proposition 5 and $^{\ominus}$Proposition 6; unmarked results are obviously polynomial-time computable

can be checked in polynomial-time is hard,[10] but its application to participatory budgeting and multi-winner elections is easy.

Proposition 5 *For the weighted asymmetric median rule (considering $\boldsymbol{w}_{we}^{asym}$), WMed-*CRED *is Δ_2^{p}-complete. For the binary restriction ($\boldsymbol{w}_{bin}^{sym}$ or $\boldsymbol{w}_{bin}^{asym}$), Med-*CRED *is Θ_2^{p}-complete. For participatory budgeting constraints, WMed-*CRED *is Δ_2^{p}-complete for weighted agendas ($\boldsymbol{w}_{we}^{asym}$) and in* p *for binary agendas ($\boldsymbol{w}_{bin}^{asym}$); it is in* p *for multi-winner elections.*

Proof For the upper Δ_2^{p} and Θ_2^{p} bounds, the proofs are routine. The median rule outputs judgments maximising a given value. We first identify the optimal score k^* using binary search, which needs a polynomial (resp. logarithmic) number of NP-oracle calls for weighted (resp. binary) agendas. In a final query, we may ask whether this maximal value is reached for some feasible outcome $X \in \mathcal{B}_F$ where $X(a^*) = 1$.

For participatory budgeting (and its unit-weight variant multi-winner elections), Talmon and Faliszewski [41] show that computing an outcome with the median rule (see Proposition 2) can be done in polynomial time; the result in an irresolute variant for CRED follows [3].

For the lower bounds, for the (weighted, asymmetric) participatory budgeting agenda, Baumeister et al. [3] showed that *WMed*-CRED for the weighted median rule is Δ_2^{p}-hard, even for settings with a single voter. We described in Section 3.2 how participatory budgeting can be encoded into our model. Hence, we can use the same reduction as Baumeister et al. [3], yielding Δ_2^{p}-hardness. Finally, *Med*-CRED for the binary median rule is known to be Θ_2^{p}-hard even for symmetric agendas (see [16]). The hardness result extends to asymmetric agendas since we can simulate a symmetric agenda with an asymmetric agenda (for each issue in the negative agenda we add a corresponding issue to the positive agenda, whose values must be kept consistent with the rationality and feasibility constraints). □

Proposition 6 *For the weighted asymmetric agenda ($\boldsymbol{w}_{we}^{asym}$), WEgal-*CRED *is Δ_2^{p}-complete. For the binary restriction ($\boldsymbol{w}_{bin}^{asym}$ or $\boldsymbol{w}_{bin}^{sym}$), Egal-*CRED *is Θ_2^{p}-complete.*

[10] de Haan [25] showed that hardness already holds for Horn formula constraints for binary symmetric agendas.

*For participatory budgeting, WEgal-*CRED *is* Δ_2^{p}*-complete for weighted agendas (*$\boldsymbol{w}_{we}^{asym}$*); for binary agendas (*$\boldsymbol{w}_{bin}^{asym}$*) Egal-*CRED *is in* Θ_2^{p} *and* coNP*-hard for participatory budgeting and multi-winner election constraints.*

Proof Some of the proofs for membership and hardness are similar to the proof of Proposition 5 for the (weighted) median rule. For the upper bound, we can also optimise a value which is bounded upwards to decide credulous outcome determination. Analogously, we need a polynomial number of NP-queries for weighted agendas and a logarithmic number of NP-queries for binary agendas. For the lower bounds, we begin with weighted agendas. Note that when the profile consists of a single voter $\boldsymbol{B} = (B_1)$, then for any constraint Γ_F and any weight vector $\boldsymbol{w}$, it holds that $\mathit{WMed}(\boldsymbol{B}, \Gamma_F, \boldsymbol{w}) = \mathit{WEgal}(\boldsymbol{B}, \Gamma_F, \boldsymbol{w})$. Hence, for single-voter profiles, we can reduce *WMed*-CRED for the weighted median rule to *WEgal*-CRED for the egalitarian rule. In the proof of Proposition 5, the reduction for the weighted median rule only uses a single voter; thus, we can use the same reduction, resulting in Δ_2^{p}-hardness. The result holds in particular for participatory budgeting constraints.

For the binary restriction, *Egal*-CRED is Θ_2^{p}-complete (see [16]), which for symmetric agendas still holds even if no feasibility constraint is given (see [25]). Hardness transfers to asymmetric agendas, as we can simulate symmetric agendas with asymmetric agendas. For multi-winner elections and participatory budgeting, Θ_2^{p} membership follows the same structure, while coNP-hardness comes from a straightforward reduction of the complement of the NP-complete problem EXACT COVER BY 3-SETS (see [21]). We assume that we are deciding whether a finite set of elements cannot be covered exactly (i.e., each element once) by a distinct selection of k 3-element subsets. We can reduce each element to a voter and each 3-element subset to a candidate, approved by the voters representing the contained elements. If we add another candidate, not approved by any voter, this candidate is in a winning committee of size k if and only if there is no exact cover. □

We also classify the weighted version of the ranked agenda rule: the results follow mainly from the literature.

Proposition 7 *For the rank rule (with immediate tie-breaking), WRank-*CRED *is* Δ_2^{p}*-complete, even for symmetric weights,* $\boldsymbol{w}_{bin}^{sym}$ *or* $\boldsymbol{w}_{we}^{sym}$*. For participatory budgeting and multi-winner elections, which are modelled with* $\boldsymbol{w}_{we}^{asym}$ *and* $\boldsymbol{w}_{bin}^{asym}$ *weights, respectively, WRank-*CRED *is in* p.

Proof The upper bound for *WRank*-CRED for the (weighted, asymmetric) ranked agenda rule can be derived by executing the rule, and then verifying if a given agenda item is in the final outcome. This can be done by ordering the agenda items by descending weighted support (using a fixed tie-breaking) and querying an NP-oracle in (at most) each of the m iterations (one for each item). For *WRank*-CRED the answer is yes, if the distinct agenda item is also in the outcome. The bounds are inherited from the binary, symmetric version, whose decision problem *Rank*-CRED is Δ_2^{p}-complete (see [15]).

For the (weighted) *WRank* rule there is a complexity gap between judgment aggregation and CCO problems, where we can find efficiently whether a subset of items can be extended to a feasible outcome (e.g., participatory budgeting). For a linear tie-breaking, we solve its decision problem by executing the rule and checking whether some item is in the outcome (which can be done in polynomial time if we can check the constraint efficiently). Note that $\mathit{Rank}(\boldsymbol{B}, \Gamma_F, \boldsymbol{w}) \in \mathit{Med}(\boldsymbol{B}, \Gamma_F, \boldsymbol{w})$ holds for multi-winner elections. □

Finally, Sonar et al. [39] showed Θ_2^{p}-completeness for *CC* with approval ballots in multi-winner elections. For asymmetric agendas, the lower bound inherits to judgment aggregation,

while the upper bound can be shown analogously to the upper bound of *Med*-CRED in the proof of Proposition 5.

Proposition 8 *CC-CRED is Θ_2^{p}-complete with either $\boldsymbol{w}_{bin}^{asym}$ or $\boldsymbol{w}_{we}^{asym}$ weights.*

In classical judgment aggregation, where the agenda is symmetric and every rational judgment is feasible, the *CC* rule becomes degenerate, because the only time an agent is not happy with an outcome is when it is the complement of their ballot. Although the usability of the Chamberlin-Courant rule is limited for symmetric agendas, we still provide tight bounds for CC-CRED on symmetric weights.

In the following proofs, we denote by $\overline{X}$ the complement of a vector X, i.e., a vector where all 0s and 1s have been swapped and by $CC_{score}(\boldsymbol{B})$ the number of voters who are simultaneously satisfied by a given outcome $X \in CC(\boldsymbol{B})$. Also, for any vector X we denote $Occur(X, \boldsymbol{B})$ the number of occurrences of X in $\boldsymbol{B}$, *i.e.*, the number of voters i such that $B_i = X$. We start with the following observation.

Observation 9 *For an arbitrary agenda A with symmetric weights $\boldsymbol{w}_{we}^{sym}$, and assuming $\Gamma_R = \Gamma_F$, then*

1. $CC_{score}(\boldsymbol{B}) = n - \min_{X \in \mathcal{B}_R} Occur(\overline{X}, \boldsymbol{B})$.
2. *If $CC_{score}(\boldsymbol{B}) = n$ then $X \in CC(\boldsymbol{B})$ if and only if $X \in \mathcal{B}_R$ and $Occur(\overline{X}, \boldsymbol{B}) = 0$, that is, for each $i \in \mathcal{N}$, $B_i \neq \overline{X}$.*
3. *If $CC_{score}(\boldsymbol{B}) < n$, then for all $X \in CC(\boldsymbol{B})$ there is some $i \in \mathcal{N}$ with $B_i = \overline{X}$.*

Proof For point 1, note that we have $CC(\boldsymbol{B}) = \operatorname{argmin}_{X \in \mathcal{B}_R} Occur(\overline{X}, \boldsymbol{B})$ by definition. Therefore, $CC_{score}(\boldsymbol{B}) = n - \min_{X \in \mathcal{B}_R} Occur(\overline{X}, \boldsymbol{B})$.

For point 2, assume $CC_{score}(\boldsymbol{B}) = n$. Then, point 1 is $\min_{X \in \mathcal{B}_R} Occur(\overline{X}, \boldsymbol{B}) = 0$, i.e., there exists a feasible vector X such that for all i, $B_i \neq \overline{X}$.

For point 3, assume $CC_{score}(\boldsymbol{B}) < n$. Then, by point 1, for each $X \in \mathcal{B}_R$, we have $Occur(\overline{X}, \boldsymbol{B}) > 0$, which means that there is an $i \in \mathcal{N}$ such that $B_i = \overline{X}$. □

We can now prove that credulous outcome determination becomes coDP-complete, where coDP $= \{L \cup L' \mid L \in \mathrm{NP}, L' \in \mathrm{coNP}\}$ (see [34]). Less formally, DP (resp. coDP) is the class of decision problems that can be written as the intersection (resp. the union) of a problem in NP and a problem in coNP. A canonical DP-complete problem is SAT- UNSAT by Papadimitriou and Yannakakis [34]: an instance (ϕ, ψ) of the problem consists of two boolean formulas ϕ and ψ, and the question is whether ϕ is satisfiable while ψ is unsatisfiable.

Proposition 10 *If $\Gamma_R = \Gamma_F$ and the agenda weights are restricted by $\boldsymbol{w}_{bin}^{sym}$ or $\boldsymbol{w}_{we}^{sym}$, CC-CRED is* coDP*-complete.*

Proof We begin with the upper bound. Observation 9 gives us an algorithm for computing $CC(\boldsymbol{B})$: if there is $X \in \mathcal{B}_R$ such that for all i, $B_i \neq \overline{X}$, then output all such vectors X; else output $\operatorname{argmin}_{i \in \mathcal{N}} Occur(\overline{B_i}, \boldsymbol{B})$. Therefore, there exists a feasible outcome $X \in CC(\boldsymbol{B})$ such that $X(a^*) = 1$ if (at least) one of these two conditions is met:

1. There is $X \in \mathcal{B}_R$ such that for all i, $B_i \neq \overline{X}$, and $X(a^*) = 1$.
2. There is no $X \in \mathcal{B}_R$ such that for all i, $B_i \neq \overline{X}$, and there is an i with $B_i(a^*) = 1$ such that $Occur(\overline{B_i}, \boldsymbol{B}) \leq Occur(\overline{B_j}, \boldsymbol{B})$ for all j.

The set of all instances meeting condition 1 (resp. 2) is a problem in NP (resp. coNP), therefore CC-CRED restricted to agenda weights in $\boldsymbol{w}_{bin}^{sym}$ or $\boldsymbol{w}_{we}^{sym}$ is in coDP.

For the lower bound, we reduce the DP-complete problem SAT- UNSAT to the complement of CC-CRED, mapping YES-instances of SAT- UNSAT to NO-instances of CC-CRED and vice-versa. Given any SAT- UNSAT instance (ϕ, ψ), we construct an instance for the complement of CC-CRED as follows.

Let X (respectively, Y) contain all variables appearing in ϕ (respectively, ψ) and their negations. We assume without loss of generality that X and Y are disjoint, i.e., $X \cap Y = \emptyset$ and set the agenda to be $A = X \cup Y \cup \{a^*, \overline{a^*}, b, \overline{b}\}$, where a^* and b are newly introduced variables; thus, $m = |X| + |Y| + 4$. The rationality and feasibility constraint is defined as $\Gamma_R = \Gamma_F = \alpha \vee \beta \vee \gamma \vee \delta$, where $\alpha = \overline{a^*} \wedge \phi \wedge b$, $\beta = a^* \wedge \psi \wedge \overline{b}$, $\gamma = \bigwedge_{a \in A^+} a$ and $\delta = \bigwedge_{\overline{a} \in A^-} \overline{a}$. [11] Note that a vote on all positive literals satisfies only γ, while its complement only satisfies δ. The profile $\boldsymbol{B}$ consist of three voters, for simplicity we give their votes on A^+: $\boldsymbol{B}^+ = (\{1\}^{m/2}, \{1\}^{m/2}, \{0\}^{m/2})$, whereby $\{1\}^{m/2}$ we denote for simplicity an agent approving all the items in A^+ and by $\{0\}^{m/2}$ an agent approving all the items in A^-. We claim that a^* is not in any outcome of $CC(\boldsymbol{B})$ if and only if (ϕ, ψ) is a YES-instance.

Without considering further satisfying assignments that do not appear in the profile $\boldsymbol{B}$, issue a^* will be a temporary winner (i.e., it appears in the judgment whose complement occurs in the profile the least number of times). There is only one way that prevents a^* from being in any outcome: i.e., if there exists a judgment $X \in \mathcal{B}_F$ with $X(a^*) = 0$ and $X \neq \{0\}^m$, while there exists no judgment $X \in \mathcal{B}_F$ with $X(a^*) = 1$ and $X \neq \{1\}^m$. By construction, this can only hold if and only if α is satisfiable and β is unsatisfiable. It is easy to see that α is satisfiable if and only if ϕ is satisfiable, while β is unsatisfiable if and only if ψ is unsatisfiable. □

5 Conclusion

We have four main take-home messages: *(i)* when looking for a declarative language to express various CCO problems, judgment aggregation is a good candidate; *(ii)* however, we need a slight generalisation where issues are weighted, and weights may be asymmetric—this generalisation allows for specific CCO problems to be seen through the lens of judgment aggregation; *(iii)* several rules studied for specific CCO problems, namely participatory budgeting, multi-winner elections, collective scheduling, and collective network design, are instances of the general settings—this shows strong connections between two specific 'sister' rules, that are instances of the same general rules and share common normative properties; *(iv)* in about half of the cases considered, the generalisation does not come with a complexity increase.

In the Appendix we present an experimental case study intended to show the applicability of our proposed general judgment aggregation framework. We report on experiments using an ILP solver to compute the result of collective networking problems, comparing the running time of three of the proposed rules in different graph configurations and vote generation models.

[11] Observe that we can adapt the construction of the constraints to be linear inequalities rather than propositional formulas. Assume ϕ and ψ are in conjunctive normal form (CNF), for which SAT- UNSAT is still DP-hard. Then, formulas α, β, γ and δ are also in CNF. For any two formulas $\varphi_1 = \bigwedge_{i=1}^{n} c_1^i$ and $\varphi_2 = \bigwedge_{j=1}^{m} c_2^j$ in CNF, we can transform $\varphi_1 \vee \varphi_2$ into CNF in polynomial time. By repeatedly using laws of distributivity, the resulting CNF formula contains a clause for every pair of clauses (one from each of the two formulas), i.e., $\varphi_1 \vee \varphi_2 \equiv \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{m} (c_1^i \vee c_2^j)$. Hence, we can transform Γ_F into CNF using space that is polynomial in the size of Γ_F. Finally, we can express any formula in CNF as ILP constraints by adding an inequality for each of its clauses (the sum of its variables' values must be at least one).

Appendix: General solvers

Our general framework of weighted asymmetric judgment aggregation can not only be used for theoretical comparisons of rules for specific CCO settings but also to obtain a modular implementation of the rules: by simply plugging in the constraints, one can focus on a particular application. Although there is no specific solver for *weighted asymmetric* judgment aggregation, integer linear programming (ILP) is an ideal choice for such a solver. Each rule given in this paper (see Section 2.2) can be translated into an ILP formulation, as shown in the following subsection.

To the best of our knowledge, this is the first ILP formulation for judgment aggregation rules—prior to this, the only other implementations have used answer set programming [26] and SAT solvers [9]. One of the benefits of using the ILP formalism is the ability to rely on its vast literature and efficient solvers. Furthermore, we conjecture that many more judgment aggregation rules (and their weighted extensions) can be expressed as an ILP, which may not be straightforward when using other solvers that do not use cardinal weights (such as SAT solvers, where weights might have to be simulated). A benefit of ILP solvers is that many are efficient, and constraints are expressed compactly as sets of inequalities. Moreover, the use of JA as a general model for CCO problems can be motivated by the natural translation of the rules into ILP. This is unlike some of the other general solvers where the translations of the rules are far more involved.

Note that the constraints for the CCO problems described in Section 3 are already presented as sets of linear inequations, allowing us to study those in ILP directly.

Integer Linear Program Formulations for Weighted Asymmetric Judgment Aggregation Rules

We give an ILP formulation for each of our studied rules. In the following, each agenda item $a_j \in A$ is given as binary variable $a_j \in \{0, 1\}$ and we assume that we are given a (possibly empty) feasibility constraint Γ_F as a set of linear inequalities.

The (weighted) median rule

Following Definition 1, we can formulate *WMed* as the following ILP.

$$\begin{aligned} \text{Maximise} \quad & \sum_{a_j \in A} \sum_{i=1}^{n} w_{a_j} \times B_i(a_j) \times a_j \\ \text{Subject to} \quad & \Gamma_F \\ & \forall a_j \in A : a_j \in \{0, 1\} \\ & \forall a_j, \overline{a_j} \in A : a_j = 1 - \overline{a_j} \end{aligned} \tag{A1}$$

The (weighted) egalitarian rule

To express the egalitarian rule in ILP we use an additional variable Z, which represents the lowest score of any agent (and thus should be maximised). This maximisation is done subject to the feasibility constraints (Γ_F) and the intersection of the outcome assignment over the agenda A with respect to each agent's ballot, which must be greater than or equal to Z. A similar ILP formulation in the context of participatory budgeting was given by Sreedurga et

al. [40].

$$\begin{array}{rc} \text{Maximise} & Z \\ \text{Subject to} & \Gamma_F \\ & \text{for } i \in \mathcal{N} : \sum_{a_j \in A} w_{a_j} \times B_i(a_j) \times a_j \geq Z \\ & \forall a_j \in A : a_j \in \{0, 1\} \\ & Z \in [0, \sum_{a \in A^+} w_a] \\ & \forall a_j, \overline{a_j} \in A : a_j = 1 - \overline{a_j} \end{array} \tag{A2}$$

The (weighted) ranked agenda rule

Note that *Rank* and *WRank* can be computed efficiently for easy-to-solve constraints (e.g., a participatory budgeting constraint). Nevertheless, we provide an ILP formulation to compute the outcome in general. In a pre-processing step, we compute the order in which decisions over the agenda are made, i.e., descending by the agents (weighted) support, where ties are broken alphabetically. For $a \in A$, let π_a be the number of items that are ranked after a in that order. Then, for the ILP formulation of *WRank* it is sufficient to ensure that it is always preferred to include issues that are processed earlier over all issues that are processed later.

$$\begin{array}{rc} \text{Maximise} & \sum_{a_j \in A} \sum_{i=1}^{n} 2^{\pi_{a_j}} \times a_j \\ \text{Subject to} & \Gamma_F \\ & \forall a_j \in A : a_j \in \{0, 1\} \\ & \forall a_j, \overline{a_j} \in A : a_j = 1 - \overline{a_j} \end{array} \tag{A3}$$

The Chamberlin-Courant rule

For an ILP formulation of the Chamberlin-Courant rule, we take inspiration from Talmon and Faliszewski [41]—see also [38] for a translation of the multi-winner election variant into ILP. We need an extra variable c_i for each agent $i \in \mathcal{N}$ that will model their satisfaction. In particular, c_i will be set to 1 if there is a project in the outcome which the agent approves.

$$\begin{array}{rc} \text{Maximise} & \sum_{i=1}^{n} c_i \\ \text{Subject to} & \Gamma_F \\ \forall i \in \mathcal{N} : & \sum_{a_j \in A} B_i(a_j) \times a_j \times w_{a_j} \geq c_i \\ \forall i \in \mathcal{N} : & c_i \in \{0, 1\} \\ \forall a_j \in A : & a_j \in \{0, 1\} \\ \forall a_j, \overline{a_j} \in A : a_j = 1 - \overline{a_j} & \end{array} \tag{A4}$$

A Case Study for General Solvers: Collective Networking

In this section, we provide a case study for the implementation of the collective networking problem (described in Section 3.3) without a budget constraint, comparing the processing time of three (binary) rules: the median rule (*Med*), the egalitarian rule (*Egal*), and the Chamberlin-Courant rule (*CC*). We do not study the (weighted) ranked agenda rules, as they are solvable in polynomial time (if it can be checked in polynomial time whether a partial assignment can be extended to a full assignment satisfying the given constraint). The implementation used the open-source *GNU Octave* software [12], and its standard ILP solver `glpk`, using two-phase primal simplex method. Our implementation is modular: i.e., the same set-up can

be altered to account for any CCO problem by changing the problem-specific components (the agenda, the constraints, and the CCO rule).

Recall that in the collective networking problem, we have a graph $G = (V, E)$ and the agents vote on the edges E in the corresponding agenda, while the rule has to find a collective spanning tree. Regarding the constraints, we allow for any possible ballot and we only impose that the outcome must be a spanning tree, thus Γ_F is composed of the ILP constraints expressed in (1), (2) and (3).

We generate the underlying network in the form of 49 connected graphs $G = (V, E)$ with number of nodes varying between 6 and 8, i.e., $V \in [6, 8]$. For each value of V we generate connected graphs with $E \in \left[|V| - 1, \frac{|V|(|V|-1)}{2}\right]$: i.e., the graphs vary from being trees (for $|E| = |V| - 1$) to being complete (for $|E| = \frac{|V|(|V|-1)}{2}$). Each graph is randomly generated as follows, for a given $|V|$ and $|E| = e$.

We initially let S and S' be two sets such that one element v_0 of V is in S (i.e., $v_0 \in S$) and $S' = V \backslash \{v_0\}$. The set S and S' can be seen as the set of connected and unconnected nodes, respectively. The algorithm iteratively chooses, at random, an item $(v_i, v_j) \in S \times S'$, moves v_j from S' to S, and adds (v_i, v_j) to the set of edges E. When S' is empty, we randomly add $e - (|V| - 1)$ extra edges from $(V \times V) \setminus E$.

We let $|\mathcal{N}| = 100$. On each generated graph G we create 10 *base profiles*. Each base profile (bp), is an $n \times E$ matrix, where for each $i \in \mathcal{N}$ we have that $bp_i \in (0, 1]^E$: that is, for each item of the agenda a real number between 0 and 1 is assigned to represent the acceptance rate of an issue by an agent. Each base profile is then transformed into 9 new profiles as follows, under the variant of the p-impartial culture model presented by [6]. According to this model, when generating approval voting ballots, one can assume that every agent independently approves each item of the agenda with probability p. Therefore, for each base profile (generated for each graph) we create 9 profiles, one for every $p \in \{0.1, \ldots, 0.9\}$. If the value of an entry b in the base profile is such that $b \leq p$, then in the created profile the agent's preference on this issue will be an approval, and an abstention otherwise.

Therefore, for each of our generated graphs (a total of 49 graphs), we created a total of 90 profiles originating from 10 base profiles and the 9 probabilities from our p-impartial culture assumption. Thus, we ran the three CCO rules on 4410 instances. We then run the ILP solver on each profile and its related graph to find an outcome for the three rules: *Med*, *Egal* and *CC*.

For each generated problem instance we measured the processing times of computing the outcome of three rules, given varying levels of the p-impartial culture and number of nodes in the underlying initial network.[12]

We begin by comparing the run-times of the three CCO rules with respect to four levels of acceptance, $p \in \{0.2, 0.4, 0.6, 0.8\}$ and varying the number of nodes in the graph (we only highlight four of our nine values of p). Figure 2 shows, for each value of p, the mean processing times over all profiles and all generated networks varying the number of nodes of the network $V \in \{6, 7, 8\}$. In Fig. 2 we use a $\log_2$-scale on the y-axis due to the exponential increase in processing times.

If we focus on *CC*, we observe that the run-time is inversely proportional to the acceptance level p, confirming the intuition that finding a collective spanning tree with *CC* is more difficult with sparse ballots.

The *Med* rule is slower to compute than the other rules for almost all p values (with some exceptions for small values of p against *CC*). Observe that without additional constraints

[12] The experiments ran on an Intel i7 processor at 4.2 GHz with 4 physical and 8 logical cores and 32 GB of memory. At any time, six instances were computed in parallel.

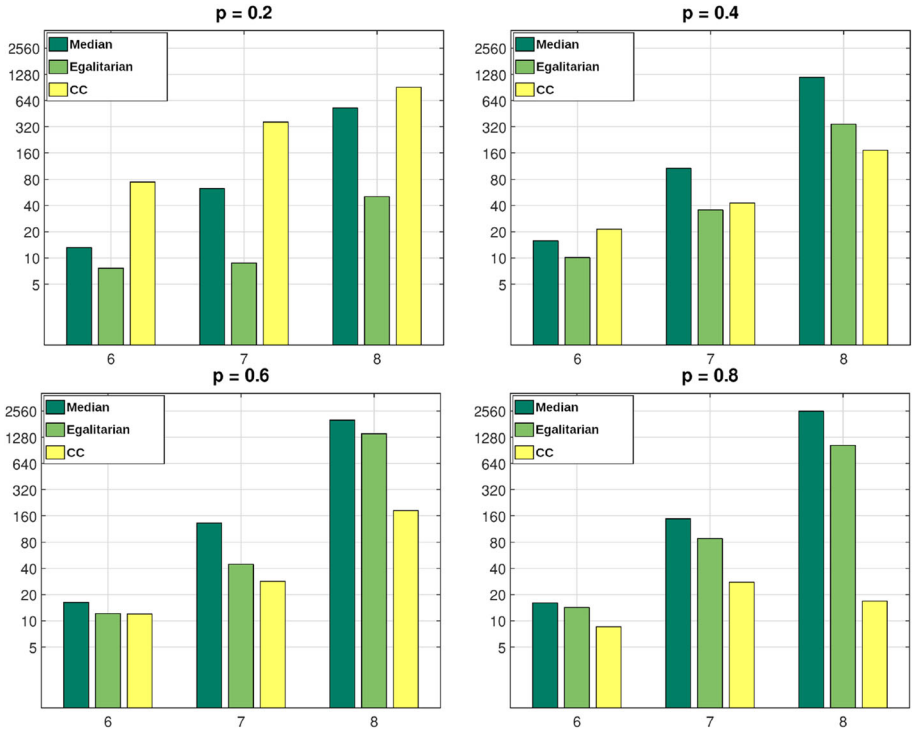

Fig. 2 Mean processing time for the *Med*, *Egal* and *CC* rules applied on the spanning tree problem. The x axis represents the number of nodes in the graph (from 6 to 8); the y axis represents the mean processing time (milliseconds) on a logarithmic scale ($\log_2$-scale). Each figure shows the mean results for a specific level of acceptance $p \in \{0.2, 0.4, 0.6, 0.8\}$

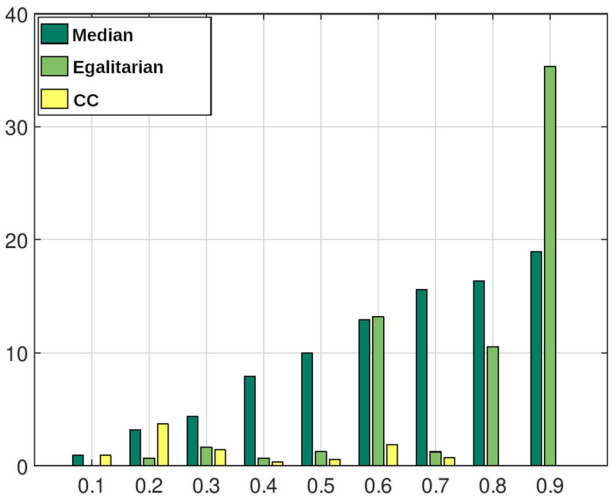

Fig. 3 Mean processing time (seconds) for the *Med*, *Egal* and *CC* rules applied on the spanning tree problem, for $p \in \{0.1, \ldots, 0.9\}$, on a complete graph with 8 nodes

on the budget, the *Med* rule is equivalent to finding the maximum spanning tree where the weights of the edges are determined by how many voters approved them. Therefore, the complexity of computing the outcome of *Med* is proportional to $N \cdot p$, i.e., the weights of the edges. A future step would be to check if the same increase in run-time can be observed for the related rule *WMed*.

Finally, the run-time of *Egal* increases steadily with the number of nodes when $p = 0.2, 0.4$, while for $p = 0.6, 0.8$ it increases at a much quicker rate. This can be explained by referring back to (A2), where we see that *Egal* maximises the value of variable Z, whose upper bound is the minimum number of items that any agents has approved. Therefore, when p is low, so is the upper bound on Z, reducing the search space; while when p is high there are more values which Z can take.

The same experiments for networks with 9 or more nodes resulted in some of the instances not completing before the chosen time-out of 1200 seconds. The observed behaviour varied with each rule: for *Med* the non-completing instances corresponded to the graphs that were close to being complete, whereas for *Med* and *Egal* the pattern was more complex, and it seemed to depend on both the structure of the graph as well as the profile of individual ballots.

Figure 3 presents the mean run-times of the three chosen rules for varying levels of acceptance p, starting from a fixed complete network with $|V| = 8$ and $|E| = 28$. Each bar represents the mean run-time for 10 different profiles, for a total of 90 instances. The figure highlights the exponential increase in running time of *Egal*, and confirms the observation that *CC* is easy on complete graphs.

Acknowledgements We thank the AMAI reviewers, as well as the reviewers of M-PREF 2022, for their valuable feedback. This work was supported by the French "Agence Nationale de la Recherche", both as part of the 'Investissements d'avenir' programme, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute), and by the ANR JCJC project SCONE (ANR 18-CE23-0009-01). It was also supported by the NRW project "Online Participation".

Data Availability The datasets for the experiments in the appendix generated and analysed during the current study are available from the corresponding author on reasonable request.

Code Availability The code for performing the experiments in the appendix is available from the corresponding author on reasonable request.

References

1. Abdelmaguid, T.F.: An efficient mixed integer linear programming model for the minimum spanning tree problem. Mathematics **6**(10), 183 (2018)
2. Aziz, H., Faliszewski, P., Grofman, B. et al.: Egalitarian committee scoring rules. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-2018) (2018)
3. Baumeister, D., Boes, L., Hillebrand, J.: Complexity of manipulative interference in participatory budgeting. In: Proceedings of the 7th international conference on algorithmic decision theory (ADT-2021) (2021)

4. Botan, S., de Haan, R., Slavkovik, M. et al.: Egalitarian judgment aggregation. In: Proceedings of the 20th international conference on autonomous agents and multiagent systems (AAMAS-2021) (2021)
5. Brams, S.J., Kilgour, D.M., Sanver, M.R.: A minimax procedure for electing committees. Public Choice **132**(3–4), 401–420 (2007)
6. Bredereck, R., Faliszewski, P., Kaczmarczyk, A. et al.: An experimental view on committees providing justified representation. In: Proceedings of the 28th international joint conference on artificial intelligence (IJCAI-2019) (2019)
7. Chamberlin, J.R., Courant, P.N.: Representative deliberations and representative decisions: proportional representation and the Borda rule. The American Political Science Review, pp. 718–733 (1983)
8. Chingoma, J., Endriss, U., de Haan, R.: Simulating multiwinner voting rules in judgment aggregation. In: Proceedings of the 21st international conference on autonomous agents and multiagent systems (AAMAS-2022) (2022)
9. Conati, A., Niskanen, A., Järvisalo, M.: Sat-based judgment aggregation. In: Proceedings of the 2023 international conference on autonomous agents and multiagent systems (AAMAS-2023) (2023)
10. Darmann, A., Klamler, C., Pferschy, U.: Computing spanning trees in a social choice context. In: Proceedings of the 2nd international workshop on computational social choice (COMSOC-2008) (2008)
11. Darmann, A., Klamler, C., Pferschy, U.: Maximizing the minimum voter satisfaction on spanning trees. Math. Soc. Sci. **58**(2), 238–250 (2009)
12. Eaton, J.W., Bateman, D., Hauberg, S. et al.: GNU Octave version 5.2.0 manual: A high-level interactive language for numerical computations (2020). https://www.gnu.org/software/octave/doc/v5.2.0/
13. Endriss, U.: Judgment aggregation. In: Brandt, F., Conitzer, V., Endriss, U., et al. (eds.) Handbook of Computational Social Choice. chap 17 (2016)
14. Endriss, U.: Judgment aggregation with rationality and feasibility constraints. In: Proceedings of the 17th international conference on autonomous agents and MultiAgent systems (AAMAS-2018) (2018)
15. Endriss, U., de Haan, R.: Complexity of the winner determination problem in judgment aggregation: Kemeny, Slater, Tideman, Young. In: Proceedings of 14th international conference on autonomous agents and multiagent systems (AAMAS-2015) (2015)
16. Endriss, U., de Haan, R., Lang, J., et al.: The complexity landscape of outcome determination in judgment aggregation. Journal of Artificial Intelligence Research **69**, 687–731 (2020)
17. Escoffier, B., Gourvès, L., Monnot, J.: Fair solutions for some multiagent optimization problems. Auton. Agent. Multi-Agent Syst. **26**(2), 184–201 (2013)
18. Everaere, P., Konieczny, S., Marquis, P.: On egalitarian belief merging. In: Proceedings of the 14th international conference on the principles of knowledge representation and reasoning (KR-2014) (2014)
19. Everaere, P., Konieczny, S., Marquis, P.: Belief merging versus judgment aggregation. In: Proceedings of the 14th international conference on autonomous agents and MultiAgent systems (AAMAS-2015) (2015)
20. Fluschnik, T., Skowron, P., Triphaus, M., et al.: Fair knapsack. In: Proceedings of the 33rd AAAI conference on artificial intelligence (AAAI-2019) (2019)
21. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H, Freeman and Company (1979)
22. Goel, A., Krishnaswamy, A.K., Sakshuwong, S. et al.: Knapsack voting. Collective Intelligence 1 (2015)
23. Grandi, U., Endriss, U.: Lifting integrity constraints in binary aggregation. Artif. Intell. **199–200**, 45–66 (2013)
24. Grossi, D., Pigozzi, G.: Judgment Aggregation: A Primer. Synthesis Lectures on Artificial Intelligence and Machine Learning (2014)
25. de Haan, R.: Hunting for tractable languages for judgment aggregation. In: Proceedings of the 16th international conference on principles of knowledge representation and reasoning (KR-2018) (2018)
26. de Haan, R., Slavkovik, M.: Answer set programming for judgment aggregation. In: Proceedings of the 28th international joint conference on artificial intelligence (IJCAI-2019) (2019)
27. Klamler, C., Pferschy, U., Ruzika, S.: Committee selection under weight constraints. Math. Soc. Sci. **64**(1), 48–56 (2012)
28. Konieczny, S., Pino Pérez, R.: Logic based merging. J. Philos. Log. **40**(2), 239–270 (2011)
29. Lackner, M., Skowron, P.: Multi-Winner Voting with Approval Preferences. Springer Briefs in Intelligent Systems, Springer (2023)
30. Lang, J., Slavkovik, M.: Judgment aggregation rules and voting rules. In: Proceedings of the 3rd International Conference on Algorithmic Decision Theory (ADT-2013) (2013)
31. Lang, L., Pigozzi, P., Slavkovik, M., et al.: A partial taxonomy of judgment aggregation rules and their properties. Soc. Choice Welfare **48**(2), 327–356 (2017)
32. Nehring, K., Pivato, M.: The median rule in judgement aggregation. Econ. Theor. **73**(4), 1051–1100 (2022)

33. Nehring, K., Pivato, M., Puppe, C.: The Condorcet set: Majority voting over interconnected propositions. Journal of Economic Theory **151**, 268–303 (2014)
34. Papadimitriou, C.H., Yannakakis, M.: The complexity of facets (and some facets of complexity). In: Proceedings of the 14th annual ACM symposium on theory of computing (1982)
35. Pascual, F., Rzadca, K., Skowron, P.: Collective schedules: Scheduling meets computational social choice. In: Proceedings of the 17th international conference on autonomous agents and MultiAgent systems (AAMAS-2018) (2018)
36. Rey, S., Endriss, U., de Haan, R.: Designing participatory budgeting mechanisms grounded in judgment aggregation. In: Proceedings of the 17th international conference on principles of knowledge representation and reasoning (KR-2020) (2020)
37. Skowron, P., Faliszewski, P.: Chamberlin-Courant rule with approval ballots: Approximating the Max-Cover problem with bounded frequencies in FPT time. Journal of Artificial Intelligence Research **60**, 687–716 (2017)
38. Skowron, P., Faliszewski, P., Slinko, A.M.: Achieving fully proportional representation: Approximability results. Artif. Intell. **222**, 67–103 (2015)
39. Sonar, C., Dey, P., Misra, N.: On the complexity of winner verification and candidate winner for multi-winner voting rules. In: Proceedings of the 29th international joint conference on artificial intelligence (IJCAI-2020) (2020)
40. Sreedurga, G., Ratan Bhardwaj, M., Narahari, Y.: Maxmin participatory budgeting. In: Proceedings of the 31st international joint conference on artificial intelligence (IJCAI-22) (2022)
41. Talmon, N., Faliszewski, P.: A framework for approval-based budgeting methods. In: Proceedings of the 33th AAAI conference on artificial intelligence (2019)