International Journal of Game Theory (2025) 54:15
https://doi.org/10.1007/s00182-025-00940-8

ORIGINAL PAPER

A general model for multi-parameter weighted voting games

Sanjay Bhattacherjee[1] **· Satya R. Chakravarty**[2] **· Palash Sarkar**[3]

Accepted: 30 April 2025 / Published online: 21 May 2025

Abstract
We introduce a new and general model for voting games with multiple weight vectors. Previously studied models are obtained as special cases of the new model. In comparison to earlier models, games have more compact representation in the new model. In particular, we show that a previously well known example of a game having dimension exponential in the number of players can be represented in the new model using only two weight vectors. Further, we identify a new sub-class of games, that we call hyperplane voting games, which are compactly expressible in the new model, but not necessarily so in the previous models. For games represented in the new model, we present dynamic programming algorithms for determining various quantities required for computing different voting power indices. Methods for computing the number of minimal winning coalitions under various restrictions were not previously known even for the earlier models.

Keywords Weighted majority voting game · Multi-parameter games · Boolean formula · Voting power · Dynamic programming

JEL Classification C71

✉ Palash Sarkar
palash@isical.ac.in

Sanjay Bhattacherjee
s.bhattacherjee@kent.ac.uk

Satya R. Chakravarty
satyarchakravarty@gmail.com

[1] School of Computing, University of Kent, Kent CT2 7NF, UK

[2] Independent Researcher, 105/1 Bidhan Nagar Road, Suncity, Block D, Flat 202, Kolkata 700067, India

[3] Indian Statistical Institute, 203, B.T. Road, Kolkata 700108, India

1 Introduction

A simple voting game consists of a set of players, where each player votes yes or no on a resolution, and depending upon the votes of all the players, the rules of the game determine whether the resolution is accepted or rejected. A very well-known game is a weighted majority voting game in which each player has a non-negative weight, and a resolution is accepted if the sum of the weights of all the players who vote yes is at least as high as a pre-determined quota. A weighted voting game is characterised by the quota and a single weight vector which determines the weights of all the players.

Games with multiple weight vectors have been studied in the literature. A weighted AND game is the intersection of k different weighted majority voting games; a coalition in the extended game is said to be winning if it wins in every constituent game (Taylor and Zwicker 1993; Algaba et al. 2003; Aziz et al. 2007; Alonso-Meijide et al. 2009; Bolus 2011; Wilms 2020). In a similar manner, a weighted OR game is the OR of k different weighted majority voting games; a coalition in the extended game is said to be winning if it wins in at least one of the constituent games (Aziz et al. 2007; Wilms 2020). A more general kind of game considers a combination of the component games using a monotone Boolean formula. To determine whether a coalition in the extended game is winning, first it is required to determine the Boolean value (1 for winning and 0 for losing) of the coalition in the component games. Next these Boolean values are combined using the monotone Boolean formula to determine whether the coalition is winning in the extended game. Such games have been studied by Faliszewski et al. (2009), Kurz and Napel (2016) and Wilms (2020).

All previous models of games with multiple weight vectors have the following two-stage structure. There are k constituent weighted majority voting games $G_1, \dots, G_k$ on the same set of players. To determine whether a coalition S of players is winning in the overall game G, in the first stage it is required to determine the Boolean value X_j of whether S is winning in the game G_j. In the second stage, the values $X_1, \dots, X_k$ are combined using a Boolean formula to determine whether S is winning in the game G. For weighted AND and weighted OR games, the Boolean formulas are simply $X_1 \wedge \cdots \wedge X_k$ and $X_1 \vee \cdots \vee X_k$ respectively.

We introduce a new and general model of games with multiple weight vectors, which we call multi-parameter weighted voting games. In the new model, it is not required for the weight vectors to be associated with some constituent weighted majority voting games. Instead, the winning coalitions are defined by a Boolean valued decision function f which is defined on the set of all possible k-tuples of weights that may arise from the set of all possible coalitions. A coalition S is said to be winning if and only if $f(s_1, \dots, s_k) = 1$, where $s_1, \dots, s_k$ are the weights of S (i.e. the sum of the weights of the players in S) in the k weight vectors. By appropriately defining the decision function f, our model can be specialised to obtain any of the previously studied two-stage models of multi-parameter games; see Sect. 3.1 for details.

Our definition allows the modelling of a new and interesting sub-class of simple games which we call *hyperplane voting games*.[1] We motivate hyperplane voting games by the following hypothetical example. A coalition of shareholders of a company has two kinds of weights. The first is the total number of shares that the coalition holds and the second is the total number of entities in the coalition. Let the proportion of shares held by a coalition S be denoted by $\rho_S^{(1)}$ (where $\rho_S^{(1)}$ is the ratio of the total number of shares held by S to the total number of shares of the company) and the proportion of the number of shareholders be denoted by $\rho_S^{(2)}$ (where $\rho_S^{(2)}$ is the ratio of the size of S to the total number of shareholders of the company). For taking certain decisions (such as those on corporate social responsibilities), suppose a company wishes to consider both the proportion of shares and the proportion of shareholders supporting a decision. This can be modelled by defining two non-negative numbers c_1 and c_2, with $c_1 + c_2 = 1$, and defining a coalition S to be winning if $c_1\rho_S^{(1)} + c_2\rho_S^{(2)} \geq q$, where $q \in (0, 1)$ is a pre-defined quota. The winning condition can be visualised as a straight line separating the two-dimensional Euclidean space into two parts, one corresponding to winning and the other corresponding to losing. More generally, in a hyperplane voting game with k weight vectors the decision function can be visualised as a hyperplane separating the k-dimensional Euclidean space into winning and losing subspaces; see Sect. 3.2 for details.

An important result on simple games shows that any simple game can be expressed as a weighted AND game (Taylor and Zwicker 1993, 1995, 1999). Since the games that can be represented using the new model are simple games, it follows that they can also be represented as weighted AND games. The issue, however, is of the compactness of the representation. Suppose a game G can be represented in the new model using k weight vectors and as a weighted AND game consisting of k' individual weighted majority games. Further, suppose that k is small, i.e. G has a compact representation in the new model. The moot question is whether k' is also small, i.e. whether G also has a compact representation as a weighted AND game. From known results it follows that in general, it is possible to represent G as a weighted AND game where the number of individual weighted majority games is exponential in the number of players. Further, in general, G may not have a compact representation, and even if it does, it is computationally infeasible to find such a representation. See Sect. 3.3 for details.

Taylor and Zwicker (1993) defined the notion of dimension of a game G which is the minimum positive integer d such that G can be represented as an intersection (i.e. AND) of d weighted majority games. In a similar manner the

[1] A special class of non-transferable utility (NTU) games defines the feasible pay-off vectors of a coalition using a hyperplane; such games are called hyperplane games (Maschler and Owen 1989; Yu 2022). In an NTU hyperplane game, for each coalition there is an associated hyperplane which defines its feasible pay-off vectors, whereas, in a hyperplane voting game, there is a single hyperplane which separates the winning from the losing coalitions. Consequently, NTU hyperplane games studied in the literature and the hyperplane voting games that we introduce are completely different concepts.

notions of co-dimension (Freixas and Marciniak 2010) (for representation as OR of weighted majority games) and Boolean dimension (Faliszewski et al. 2009) (for representation as a Boolean combination of weighted majority games) have been defined. Based on the new model, we introduce the notion of weight dimension of G which is the minimum integer k such that G can be represented as a multi-parameter weighted voting game with k weight vectors. To bring out the advantage of the weight dimension, we consider two examples. For the EU voting game, the Boolean dimension is 3 (see Sect. 3.3). The dimension of this game is not exactly known, and only lower and upper bounds are known (Kurz and Napel 2016; Kober and Weltge 2021; Chen et al. 2019). We show that the weight dimension of the EU voting game is 2. A well known example by Taylor and Zwicker (1999) provides a sequence of games where the dimension of any game in the sequence is exponential in the number of players. We show that for all games in the sequence the weight dimension is 2. See Sect. 3.3 for details.

The fundamental notion related to a voting game is voting power. The voting power of a player quantifies a player's ability to affect the outcome. The literature contains a number of voting power measures, the most important among them being the Penrose–Banzhaf (proposed independently by Penrose 1946; Banzhaf 1965) and the Shapley and Shubik (1954) measures.[2] To study a particular game, it is important to be able to compute the voting powers of all the players. This requires computing several basic values related to a voting game. Algorithms for computing various power measures for weighted voting games are known (see Matsui and Matsui 2000; Chakravarty et al. 2015). These algorithms become important tools for analysing practical games. As an example, we mention the work of Bhattacherjee and Sarkar (2019) which used such algorithms for studying the EU and IMF voting games with respect to the inequality in their voting powers.[3,4]

For any game represented using the new model, we provide dynamic programming algorithms for computing the following: number of winning coalitions, number of winning coalitions containing a particular player, determining whether a given player is a blocker or not, number of coalitions in which a player is a swing, number of coalitions of a particular cardinality in which a player is a swing, number of minimal winning coalitions, number of minimal winning coalitions containing a designated player, number of minimal winning coalitions of a particular cardinality containing a designated player. Further, we show that it is possible to use parallel processes to compute the above statistics for all the players in essentially the same time as that for computing the statistics for one player.

For multiple weighted games, algorithms for computing the Banzhaf and the Shapley–Shubik voting powers have been proposed only for representation of games in the following special models: weighted AND games (Algaba et al. 2003;

[2] Alternatives and variations of these indices were suggested, among others, by Rae (1969), Coleman (1971), Deegan and Packel (1978), Johnston (1978), and Holler (1982).

[3] For an earlier analysis on the inequality of voting system in the EU, see Leech (2002).

[4] For earlier but not closely related approaches to the measurement of inequality in voting games, see Einy and Peleg (1991), Laruelle and Valenciano (2004) and Weber (2016).

Bolus 2011; Wilms 2020), weighted OR games (Bolus 2011; Wilms 2020) and combination of weighted games represented by a monotone Boolean formula in conjunctive normal form (Bolus 2011; Wilms 2020). The most recent of these works is that of Wilms (2020). The complexities of the algorithms presented by Wilms are determined by k which is the number of constituent weighted majority games. In contrast, the complexities of our algorithms are determined by k' which is the number of distinct weight vectors used in the representation of the game in the new model. In general $k' \leq k$ and there are examples where k' is much smaller than k (see Sect. 3.3). So in general, our algorithms are faster than those of Wilms. See Sect. 5.1 for details.

In Sect. 2 we provide the necessary background. The new model of multi-parameter voting games is introduced in Sect. 3. Two basic recurrences and their computation using dynamic programming are described in Sect. 4. These recurrences are used in Sect. 5 to obtain formulas for the number of winning coalitions and the number of coalitions in which a player is a swing and in Sect. 6 to obtain the number of minimal winning coalitions.

2 Preliminaries

In the following, the cardinality of a finite set S will be denoted by $\#S$ and the absolute value of a real number x will be denoted by $|x|$.

2.1 Voting games

We provide some standard definitions arising in the context of voting games. For details the reader may consult Felsenthal and Machover (1998), Laruelle and Valenciano (2008) and Chakravarty et al. (2015).

Let $N = \{1, \dots, n\}$ be a set of n players. A subset of N is called a *coalition*. The set of all voting coalitions is denoted by 2^N. A *simple voting game* G is given by its characteristic function $\widehat{G} : 2^N \rightarrow \{0, 1\}$, where a winning coalition is assigned the value 1 and a losing coalition is assigned the value 0. Below we recall some basic notions about simple voting games.

1. The set of all winning coalitions of G will be denoted by W_G.
2. A player $i \in N$ is called a *blocker* if i is present in every winning coalition.
3. Let $S \subseteq N$ and $i \in S$.
   - The player i is said to be a *positive swing* in S if $i \in S$, $\widehat{G}(S) = 1$, and $\widehat{G}(S \setminus \{i\}) = 0$. That is, if player i leaves the winning coalition S then the resulting coalition is a losing coalition.
   - The player i is said to be a *negative swing* in S if $i \in S$, $\widehat{G}(S) = 0$, and $\widehat{G}(S \setminus \{i\}) = 1$. That is, if player i leaves the losing coalition S then the resulting coalition is a winning coalition.

- The number of coalitions in which a player i is a positive swing will be denoted by m_i^+, and the number of coalitions in which i is a negative swing will be denoted by m_i^-. Further, $\mathbf{m}^+$ (resp. $\mathbf{m}^-$) denotes the vector $\mathbf{m}^+ = (m_1^+, \dots, m_n^+)$ (resp. $\mathbf{m}^- = (m_1^-, \dots, m_n^-)$).
- We let $m_i = m_i^+ + m_i^-$ denote the total number of coalitions in which player i is a swing (either positive or negative) and $\mathbf{m} = (m_1, \dots, m_n)$.
- The number of coalitions of cardinality $c \in \{0, \dots, n\}$ in which a player i is a positive (resp. negative) swing will be denoted by $m_{i,c}^+$ (resp. $m_{i,c}^-$), and $m_{i,c} = m_{i,c}^+ + m_{i,c}^-$. Further, $\mathbf{M}^+$, $\mathbf{M}^-$ and $\mathbf{M}$ denote the $n \times (n+1)$ matrices whose $(i,\ c)$-th entries, for $1 \le i \le n$ and $0 \le c \le n$, are $m_{i,c}^+$, $m_{i,c}^-$ and $m_{i,c}$ respectively.

4. A coalition $S \subseteq N$ is called a *minimal winning coalition* if $\widehat{G}(S) = 1$ and there is no $T \subset S$ for which $\widehat{G}(T) = 1$. That is, no proper subset of the winning coalition S can be winning. The set of all minimal winning coalitions will be denoted by MW_G.

   - For a player $i \in N$, by γ_i we will denote the number of minimal winning coalitions containing the player i. Further, $\boldsymbol{\gamma}$ denotes the vector $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_n)$.
   - For a player $i \in N$ and $c \in \{0, \dots, n\}$, by $\gamma_{i,c}$ we will denote the number of minimal winning coalitions of cardinality c containing the player i. Further, $\boldsymbol{\Gamma}$ denotes the $n \times (n+1)$ matrix whose (i, c)-th entry is $\gamma_{i,c}, 1 \le i \le n, 0 \le c \le n$.

5. A voting game G is said to be *monotone* if for coalitions S, T with $S \subseteq T \subseteq N$, $\widehat{G}(S) = 1$ implies $\widehat{G}(T) = 1$. Note that in a monotone game, a player i cannot be a negative swing in any coalition. So for monotone games, we simply say "swing" instead of "positive swing." In this case $m_i = m_i^+$ and $m_{i,c} = m_{i,c}^+$.
6. A player $i \in N$ is called a *null* (or often also *dummy*) if i is not a swing in any coalition, i.e., if $m_i = 0$.

2.2 Voting power

The notion of power is an important concept in a voting system. A *power measure* captures the capability of a player to influence the outcome of a vote. Given a game G on a set of players N and a player i in N, a power measure $\mathcal{P}$ associates a non-negative real number $v_i = \mathcal{P}_G(i)$ to the player i. The value v_i captures the power that i has in the game G. A voting power measure is said to be *anonymous* if a permutation of the voters does not change the power of a voter. All the voting power measures that we consider satisfy anonymity.

There are a number of voting power measures in the literature. The following are important quantities and the ability to compute them permits the computation of several of the important voting power measures.

1. Number of winning coalitions.
2. Number of winning coalitions containing a particular player.

3. Number of coalitions in which a player is a swing.
4. Number of coalitions of a particular cardinality in which a player is a swing.
5. Number of minimal winning coalitions.
6. Number of minimal winning coalitions containing a designated player.
7. Number of minimal winning coalitions of a particular cardinality containing a designated player.

The Penrose–Banzhaf (Penrose 1946; Banzhaf 1965) measure can be computed from the number of coalitions in which a player is a swing; the Shapley–Shubik (Shapley and Shubik 1954) index can be computed from the number of coalitions of a particular cardinality in which a player is a swing; the two Coleman measures (Coleman 1971) can be computed from the number of coalitions in which a player is a swing and the total number of winning coalitions in the game; the Deegan–Packel (Deegan and Packel 1978) index can be computed from the total number of minimal winning coalitions in the game and the number of minimal winning coalitions of a particular cardinality containing a player. On the other hand, the Johnston (1978) index can be regarded as an amalgam of the Banzhaf and Deegan–Packel indices; the Rae (1969) index relies on the total number of ways in which a voter agrees with the outcome of the voting system; the Holler (1982) index, like the Deegan–Packel index, is also based on the minimal winning coalitions, though the argument for using minimal winning coalitions is distinct from that used in the Deegan–Packel index. See Felsenthal and Machover (1998), Matsui and Matsui (2000), and Chakravarty et al. (2015) for further details.

3 General multi-parameter weighted voting games

We consider voting games with multiple weight vectors with each weight vector capturing some aspect of the background problem. Further, we consider the winning rule to be quite general which permits wide flexibility in modelling. We start with the formal definition of such a game.

Definition 1 (*General k-parameter weighted voting game*) Consider a tuple $(N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, where

- $N = \{1, \dots, n\}$ is a set of players;
- for $1 \le j \le k$, $\mathbf{w}^{(j)} = (w_1^{(j)}, w_2^{(j)}, \dots, w_n^{(j)})$ is a vector of non-negative weights with $w_i^{(j)}$ being the weight of player i in the j-th weight vector;
- and $f : \Omega \to \{0, 1\}$, is a decision function, where

$$\Omega = \{(s_1, \dots, s_k) \colon \text{there is a coalition } S \subseteq N \text{ satisfying } w_S^{(1)} = s_1, \dots, w_S^{(k)} = s_k\},$$

  with $w_S^{(j)} = \sum_{i \in S} w_i^{(j)}$, $j = 1, \dots, k$, being the sum of the weights of all the players in the coalition S as given by the j-th weight vector.

The tuple $(N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$ defines a voting game G given by its characteristic function $\widehat{G} : 2^N \to \{0, 1\}$ in the following manner.

$$\widehat{G}(S) = f(w_S^{(1)}, w_S^{(2)}, \dots, w_S^{(k)}).$$

We will write

$$G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$$

to denote the voting game arising from the tuple $(N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$.

We note the following points regarding Definition 1.

1. The set Ω can be visualised as follows. The k-tuple $(s_1, \dots, s_k)$ is in Ω if and only if there is some coalition S whose weight according to the j-th weight vector is s_j, $j = 1, \dots, k$.
2. The decision function f determines the winning coalitions. A coalition S is winning if and only if f takes the value 1 on input $(w_S^{(1)}, \dots, w_S^{(k)})$.
3. The weights are restricted to be non-negative. See Remark 3 for a discussion on how this restriction can be lifted.

We say that the decision function f is monotone, if it is monotone on each component, i.e. if for any $i \in \{1, \dots, n\}$, and

$$(s_1, \dots, s_{i-1}, s_i, s_{i+1}, \dots, s_k), (s_1, \dots, s_{i-1}, s_i', s_{i+1}, \dots, s_k) \in \Omega$$

with $s_i' > s_i$, then

$$f(s_1, \dots, s_{i-1}, s_i', s_{i+1}, \dots, s_k) \geq f(s_1, \dots, s_{i-1}, s_i, s_{i+1}, \dots, s_k).$$

We have the following simple result on monotone decision functions.

Proposition 1

1. The decision function f is monotone if and only if for $(s_1, \dots, s_k), (s_1', \dots, s_k') \in \Omega$ with $s_1' \geq s_1, \dots, s_k' \geq s_k$, $f(s_1', \dots, s_k') \geq f(s_1, \dots, s_k)$.
2. If the decision function f is monotone, then for $(s_1, \dots, s_k), (s_1', \dots, s_k') \in \Omega$ with $s_1' \geq s_1, \dots, s_k' \geq s_k$, we have

$$f(s_1, \dots, s_k) f(s_1', \dots, s_k') = f(s_1, \dots, s_k). \tag{1}$$

Proof We start with the proof of the first point. Clearly, if the given condition holds, then f is monotone on each component. Conversely, suppose f is monotone on each component. Given $(s_1, \dots, s_k), (s_1', \dots, s_k') \in \Omega$ with $s_1 \geq s_1', \dots, s_k \geq s_k'$ and $(s_1, \dots, s_k) \neq (s_1', \dots, s_k')$, suppose that $s_1' > s_1$. Then

$$f(s_1', s_2', \dots, s_k') \geq f(s_1' - 1, s_2', \dots, s_k') \geq \cdots \geq f(s_1, s_2', \dots, s_k').$$

Continuing over each index, we finally obtain $f(s_1', \dots, s_k') \geq f(s_1, \dots, s_k)$.

The argument for the second point is the following. If $f(s_1, \dots, s_k) = 0$, then both sides of (1) are equal to 0. On the other hand, if $f(s_1, \dots, s_k) = 1$, then from the first point, it follows that $f(s'_1, \dots, s'_k)$ is also equal to 1, and in this case, both sides of (1) are equal to 1. □

It is easy to see that a k-parameter weighted voting game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$ is monotone if and only if f is monotone. We note that the formulas in Sect. 5 hold irrespective of whether f is monotone or not, while the formulas in Sect. 6 require f to be monotone.

We introduce the following notation. For $j = 1, \dots, k$, let

$$\sigma^{(j)} = w_1^{(j)} + \cdots + w_n^{(j)}. \tag{2}$$

In other words, $\sigma^{(j)}$ is the total weight of the j-th weight vector. Note that for $(s_1, \dots, s_k) \in \Omega$, we have $0 \le s_j \le \sigma^{(j)}$ for each $j = 1, \dots, k$.

We illustrate the expressive power of the new model in two ways. Firstly, we show that all previously considered models of weighted voting can be obtained as particular cases of the new model. Secondly, we introduce an interesting new model of games which can be compactly modelled using the new framework, but cannot necessarily be done so using known models of weighted voting. Further, we show that certain games can be (much) more compactly expressed in the new model than in earlier models.

3.1 Particular cases of general multi-parameter weighted voting games

By suitably defining the decision function f, it is possible to obtain particular models of multi-parameter weighted voting games. Below we show how previously known models of weighted voting games can be derived from general multi-parameter weighted voting games by suitably defining f.

Weighted majority voting games. Suppose $k = 1$ and $f(w_S^{(1)})$ is defined to take the value 1 if and only if $w_S^{(1)}/\sigma^{(1)} \ge q$, where $q \in (0, 1)$ is a fixed real number. Then G is a weighted majority voting game.

Weighted AND games. Suppose G is the weighted AND of k games $G_1, \dots, G_k$, where for $j = 1, \dots, k$, the set of players of G_j is N, the weight vector of G_j is $\mathbf{w}^{(j)} = (w_1^{(j)}, \dots, w_n^{(j)})$, and the quota for G_j is $q_j \in (0, 1)$. Weighted AND games have been variously called vector-weighted system (Taylor and Zwicker 1993), weighted k-majority voting game (Algaba et al. 2003), m-multiple weighted voting game (Aziz et al. 2007), weighted multiple majority game (Alonso-Meijide et al. 2009), vector-weighted majority game (Bolus 2011) and weighted k-tier AND-game (Wilms 2020).

We define a multi-parameter representation of G as $(N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, where for any coalition $S \subseteq N$, $f(w_S^{(1)}, \dots, w_S^{(k)})$ is defined to take the value 1 if and only if

$$\left(w_S^{(1)}/\sigma^{(1)} \geq q_1\right) \wedge \cdots \wedge \left(w_S^{(k)}/\sigma^{(k)} \geq q_k\right).$$

Weighted OR games. Suppose G is the weighted OR of k games $G_1, \ldots, G_k$, where for $j = 1, \ldots, k$, the set of players of G_j is N, the weight vector of G_j is $\mathbf{w}^{(j)} = (w_1^{(j)}, \ldots, w_n^{(j)})$, and the quota for G_j is $q_j \in (0, 1)$. Weighted OR games have been called weighted k-tier OR-game (Wilms 2020).

We define a multi-parameter representation of G as $(N, \mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}, f)$, where for any coalition $S \subseteq N$, $f(w_S^{(1)}, \ldots, w_S^{(k)})$ is defined to take the value 1 if and only if

$$\left(w_S^{(1)}/\sigma^{(1)} \geq q_1\right) \vee \cdots \vee \left(w_S^{(k)}/\sigma^{(k)} \geq q_k\right).$$

Monotone Boolean combination of weighted games. Suppose G is represented as a monotone Boolean combination of k games $G_1, \ldots, G_k$, where for $j = 1, \ldots, k$, the set of players of G_j is N, the weight vector of G_j is $\mathbf{w}^{(j)} = (w_1^{(j)}, \ldots, w_n^{(j)})$, and the quota for G_j is $q_j \in (0, 1)$. Let $X_1, \ldots, X_k$ be Boolean valued variables such that for a coalition S, the Boolean variable X_j, $j = 1, \ldots, k$, takes the value 1 if and only if S is winning in the game G_j. Let $\phi(X_1, \ldots, X_k)$ be a monotone Boolean function[5] of the variables $X_1, \ldots, X_k$. Then a coalition S is winning in G if and only if $\phi(X_1, \ldots, X_k) = 1$. The above model of games has been studied by Faliszewski et al. (2009), Kurz and Napel (2016) and Wilms (2020).

We define a multi-parameter representation of G as $(N, \mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}, f)$, where for any coalition $S \subseteq N$, $f(w_S^{(1)}, \ldots, w_S^{(k)})$ is defined as follows.

$$f(w_S^{(1)}, \ldots, w_S^{(k)}) = \phi(X_1, \ldots, X_k). \tag{3}$$

3.2 Hyperplane voting games

Suppose the set N of players participate in k committees. The voting weights of the players in the various committees are not necessarily the same. So we have k weight vectors $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}$, with weight vector $\mathbf{w}^{(j)}$ specifying the weights of the n players in the j-th committee. Let as before $\sigma^{(j)}$ be the sum of the j-th weight vector. A coalition S has weight $w_S^{(j)}$ in the j-th committee. The proportional weight of S in the j-th committee is $w_S^{(j)}/\sigma^{(j)}$. All the committees do not necessarily have the same importance. We may assign weight c_j to the j-th committee, where $c_1, \ldots, c_k$ are real numbers in [0, 1] such that $c_1 + \cdots + c_k = 1$. Let $q \in (0, 1)$ be a quota. The coalition S is defined to be winning if

$$c_1 \cdot \frac{w_S^{(1)}}{\sigma^{(1)}} + \cdots + c_k \cdot \frac{w_S^{(k)}}{\sigma^{(k)}} \geq q. \tag{4}$$

[5] A Boolean function of variables $X_1, \ldots, X_k$ is said to be monotone if it can be expressed by a Boolean algebra expression involving the variables $X_1, \ldots, X_k$ and the logical connectives $\wedge$ (AND) and $\vee$ (OR).

This defines a general k-parameter weighted voting game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, where $f(w_S^{(1)}, \dots, w_S^{(k)})$ is 1 if and only if (4) holds. Taking $k = 2$ and the two weights of a coalition to be the number of shares held by the coalition and the size of the coalition, we obtain the example of 2-parameter hyperplane game described in the introduction.

The formal definition of hyperplane voting games is the following.

Definition 2 A hyperplane voting game is a general multi-parameter weighted voting game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, where the decision function f is defined as follows:

$$f(w_S^{(1)}, \dots, w_S^{(k)}) = 1 \text{ if and only if } d_1 w_S^{(1)} + \cdots + d_k w_S^{(k)} \geq q, \tag{5}$$

for fixed real numbers $d_1, \dots, d_k$ and q.

Note that we obtain (4) by taking $d_j = c_j / \sigma^{(j)}$, for $j = 1, \dots, k$.

In Definition 2, the function f can be seen as a decision rule to partition the domain Ω of the decision function f into two parts by a hyperplane; a coalition $S \subseteq N$ is winning if and only if its corresponding weight vector $(w_S^{(1)}, \dots, w_S^{(k)}) \in \Omega$ is on or above the hyperplane which defines the support of the function f. In view of this, we call such games to be hyperplane voting games. Note that in the formal definition given in Definition 2, we do not insist that the sum of the d_i's must be 1. The coefficients $d_1, \dots, d_k$ are defined to be real numbers. If we restrict the $d_1, \dots, d_k$ to be non-negative, then we obtain a sub-class of hyperplane voting games which are monotone. We will primarily be interested in monotone hyperplane voting games, though below we briefly mention an example of a non-monotonic hyperplane voting game.

The crucial difference between hyperplane voting games and monotone Boolean combination of weighted games arises from the manner in which the weights are used in defining the function f. In a monotone Boolean combination of weighted games, the k weights are first used to determine winning/losing conditions individually in each of the k games and then these Boolean values are combined using a Boolean formula to obtain the value of f. In more details, to determine whether a coalition S is winning, first it is determined whether the coalition S is winning in each of the games $G_1, \dots, G_k$ defined by the k weight vectors. Suppose X_j, $j = 1, \dots, k$, denotes the Boolean value that indicates whether S is winning in the game G_j. Then the Boolean values $X_1, \dots, X_k$ are combined using a monotone Boolean formula ϕ to determine whether S is winning in the Boolean combination of the games $G_1, \dots, G_k$ given by ϕ. On the other hand, in hyperplane voting games, the decision function f can be defined directly using the k possible weights of a coalition. For a coalition S, from (5), we see that $f(w_S^{(1)}, \dots, w_S^{(k)})$ takes the value 1 if and only if $d_1 w_S^{(1)} + \cdots + d_k w_S^{(k)} \geq q$. This direct approach permits avoiding the more restrictive two-stage determination of winning condition in a monotone Boolean combination of weighted games.

Non-monotonic games: The Boolean formula ϕ in (3) is a monotone Boolean formula. Faliszewski et al. (2009) have considered the more general case where

ϕ is an arbitrary Boolean formula. The corresponding games are not necessarily monotone. Keeping this in mind, we discuss an interesting example of how hyperplane voting games can be used to model possibly non-monotonic decision making procedures arising in machine learning contexts.

Suppose in (5), the coefficients $d_1, \ldots, d_k$ are allowed to be negative. It follows that f is not necessarily monotone and so the hyperplane voting game arising from such an f is also not necessarily monotone. We provide a natural interpretation of such a hyperplane voting game in the context of the binary classification problem in machine learning. Suppose an object has k features. The requirement is to categorise it into one of two categories (say acceptable or unacceptable) based on the opinions of n persons. Each person classifies the object into one of the two categories. The opinions of the n persons have different weights for the k features which are given by k weight vectors. Suppose S is the set of persons who have classified the object as acceptable. The weights $w_S^{(1)}, \ldots, w_S^{(k)}$ of the coalition S for the k features are then used to determine whether the object is finally acceptable or not. So the decision is based on the value of a function f applied to the weights $w_S^{(1)}, \ldots, w_S^{(k)}$. In machine learning context, a commonly used decision function is a hyperplane i.e., $f(w_S^{(1)}, \ldots, w_S^{(k)}) = 1$ if and only if $d_1 w_S^{(1)} + \cdots + d_k w_S^{(k)} \geq d_0$ for some real numbers $d_0, d_1, \ldots, d_k$. The coefficient d_j is the weight associated with the j-th feature. A positive d_j represents a positive feature of the object, i.e. a feature which makes the object acceptable, while a negative d_j represents a negative feature of the object, i.e. a feature which makes the object unacceptable. If some negative feature is present, then the resulting k-parameter hyperplane voting game is not necessarily monotone.[6]

3.3 Compactness of representation

The following result shows that the new model that we introduce is a new way of representing simple games.

Proposition 2 The class of games which can be represented as multi-parameter weighted voting games is equal to the class of simple games.

Proof Suppose $G = (N, \mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}, f)$ is a multi-parameter weighted voting game. Then clearly G is a simple game. Conversely, suppose G is a simple game. Taylor and Zwicker (1993) showed that G can be represented as a weighted AND game. Since a weighted AND game is a special case of a multi-parameter weighted voting game (see Sect. 3.1), it follows that G can be represented as a multi-parameter weighted voting game. □

[6] It is also possible to allow the weights of the persons to be negative, where a negative weight indicates a person's disapproval of the object for a particular feature. Negative weights may also result in a non-monotonic voting game defined by a hyperplane. It does not, however, fit our definition of hyperplane voting games since our definition of general multi-parameter weighted voting games restricts the weight vectors to have non-negative components. See Remark 3 for a discussion on negative weights.

Remark 1 Since the model used to represent a game does not change the game itself, the equivalence stated in Proposition 2 also holds when specialised to monotone simple games.

Following Taylor and Zwicker (1993), a simple game G is said to have dimension d if d is the minimum positive integer such that G can be represented as a weighted AND game consisting of d weighted voting games. The dimension of a game determines its compactness as a weighted AND game. Taylor and Zwicker (1993) showed that there exists a sequence $G_0, G_1, G_2, \ldots$ of distinct simple games such that G_i has dimension at least 2^{n_i-1}, where n_i is the number of players in G_i. (We provide more details below.) This result shows that there exists games with exponentially large dimension and so given a simple game, in general it is not possible to obtain a compact representation of the game as a weighted AND game. Taylor and Zwicker (1993) also proved an upper bound on the dimension of a simple game G. They showed that G can be represented using a weighted AND game consisting of $O(\#(2^N \setminus W_G))$ weighted majority games and so the dimension is at most $O(\#(2^N \setminus W_G))$. This upper bound on the dimension is exponential in the number of players n. It is possible that the actual value of the dimension is smaller. However, the problem of determining the dimension of a simple monotone game is known to be NP-hard (see Deĭneko and Woeginger 2006). In particular, the following problem was shown to be NP-hard in Deĭneko and Woeginger (2006). Given a weighted AND game G with d_1 weight vectors and a positive integer $d_2 < d_1$, determine whether there exists a weighted AND game with d_2 weight vectors which represents G. Due to the NP-hardness of this problem, reducing the exponential upper bound on the dimension of a game is computationally intractable.

From Proposition 2, it follows that in theory any multi-parameter weighted voting monotone game can be expressed as a weighted AND game. Suppose that a multi-parameter game G with k weight vectors can be expressed as a weighted AND game consisting of k' weighted majority games. The compactness of the weighted AND game is determined by the value of k'. From the above discussion, it follows that using known results, in general it is possible to represent G as a weighted AND game consisting of $k' = O(\#(2^N \setminus W_G))$ constituent weighted majority games. Further, obtaining a representation of G as a weighted AND game with fewer number of constituent weighted majority games (or even deciding whether such a representation exists) is computationally infeasible. Suppose k is small. For one thing, it is not necessarily guaranteed that the dimension of G is small. Secondly, one may obtain a representation of G as a weighted AND game using exponentially many weighted majority games, and then determining whether there is another representation of G as a weighted AND game with a smaller number of weighted majority games is computationally infeasible.

The co-dimension of a simple game G is defined to be the least positive integer d such that G can be represented as a weighted OR game consisting of d weighted voting games (see Freixas and Marciniak 2010). The Boolean dimension (Faliszewski et al. 2009) of a simple game G is the least positive integer d such that G can be represented as a monotone Boolean combination of d weighted voting games. Among the three concepts of dimension, co-dimension and Boolean dimension, it is the notion of dimension which has been studied more extensively.

All the three concepts of dimension, co-dimension and Boolean dimension counts the number of constituent weighted voting games and not the number of weight vectors that are used to define the constituent games. It is possible that two constituent weighted voting games may use the same weight vector, but different quota values. In such a situation, the number of weight vectors will be less than the number of constituent games. Such reduction in the number of weight vectors is not captured by the notions of dimension, co-dimension and Boolean dimension.

Remark 2 The compactness issue here is not merely that of counting the number constituent weighted majority games in weighted AND, weighed OR and Boolean combination of games versus counting the number of distinct weight vectors in multi-parameter weighted voting games. As we will see below, the time and space complexities of algorithms to compute various statistics of the game G represented as a multi-parameter weighted voting game are linear in $\sigma^{(1)} \cdots \sigma^{(k)}$, where k is the number of weight vectors. On the other hand, previous algorithms (see Algaba et al. 2003; Bolus 2011; Wilms 2020) for computing swings of G for weighted AND and weighted OR games have time and space complexities to be linear in $\sigma^{(1)} \cdots \sigma^{(k')}$, where k' is the number of constituent games. Since in general k' is greater than k, the time and space complexities obtained from the representation of G as a multi-parameter weighted voting game will in general be smaller.

Based on the new model, we introduce a new notion of dimension of a game. For a simple game G, we define the *weight dimension* of G to be the minimum positive integer k such that G can be represented as a multi-parameter weighted voting game with k weight vectors. From Sect. 3.1, it follows that the weight dimension of G is at most the Boolean dimension of G. We illustrate the non-triviality of the notion of weight dimension using the following examples.

EU voting. The Lisbon voting rules of the EU Council can be modelled using a game G which is a monotone Boolean combination of three weighted voting games (see Algaba et al. 2007; Kurz and Napel 2016). Let the games be $G^{(1)}, G^{(2)}$ and $G^{(3)}$ all defined on the set $N = \{1, \dots, n\}$ of players, where $n = 28$. (Note that pre-Brexit there were 28 countries in the EU.) The weight vector for $G^{(1)}$ is $(1, \dots, 1)$ and the quota is $q^{(1)} = 0.55$; the weight vector for $G^{(2)}$ is given by the population sizes of the n countries and the quota is $q^{(2)} = 0.65$; finally, the weight vector for $G^{(3)}$ is also $(1, \dots, 1)$ and the quota is $q^{(3)} = 0.86$. (Game $G^{(3)}$ captures the blocking minority condition which states that at least 4 nations are required to block a resolution.) For a coalition $S \subseteq N$, let X_i, $i = 1, 2, 3$ be the Boolean value which determines whether S is winning in the game $G^{(i)}$. Then the Boolean formula ϕ determining whether S is winning in G is $\phi(X_1, X_2, X_3) = (X_1 \wedge X_2) \vee X_3$.

The Boolean dimension of G is 3. The dimension of G is not exactly known. A lower bound of 7 on the dimension of G was proved by Kurz and Napel (2016) and later the lower bound was improved to 8 by Kober and Weltge (2021).

An upper bound of 24 on the dimension was proved by Chen et al. (2019). So representing G as a weighted AND game is less compact than as a monotone Boolean combination of weighted voting games.

The formulation of G as a monotone Boolean combination of weighted voting games uses three games and three weight vectors. Even though the weight vectors for the first and third games are the same, this is not reflected in the representation of the game.

Next we consider representing G using the model of multi-parameter weighted voting game. The set of players is N. There are two weight vectors $\mathbf{w}^{(1)} = (1, \dots, 1)$ and $\mathbf{w}^{(2)}$ which records the population sizes of the nations. The decision function f takes two inputs and is defined as follows. For a coalition $S \subseteq N$, $f(w_S^{(1)}, w_S^{(2)})$ takes the value 1 if and only if

$$\left(w_S^{(1)}/28 \geq 0.55 \wedge w_S^{(2)}/\sigma^{(2)} \geq 0.65\right) \vee \left(w_S^{(1)}/28 \geq 0.86\right). \tag{6}$$

So the weight dimension of G is two which is less than its Boolean dimension.

An example from Taylor and Zwicker (1999). Theorem 1.7.5 of Taylor and Zwicker (1999) (see also Section 3 of Olsen et al. 2016) provides an example of a simple game whose dimension is exponential in the number of players. For an odd integer m, the game G_m is defined as follows. Let $n = 2m$ be the number of players, and partition the set $N = \{1, \dots, 2m\}$ of players as $N = N_m \cup \overline{N}_m$, where $N_m = \{1, \dots, m\}$ and $\overline{N}_m = \{m+1, \dots, 2m\}$. A coalition $S \subseteq N$ is defined to be winning in G if and only if either $\#S \geq m+1$, or $\#S = m$ and $\#(S \cap \overline{N}_m)$ is even. The dimension of G_m is 2^{m-1}, i.e. to represent G as a weighted AND game, a total of 2^{m-1} constituent weighted voting games are required. See Theorem 1.7.5 of Taylor and Zwicker (1999) for a proof that the dimension is at least 2^{m-1} and Theorem 1 of Olsen et al. (2016) for a proof that the dimension is at most 2^{m-1}. (Note that due to the condition $\#(S \cap \overline{N}_m)$ is even, G_m is not a monotone game.)

We now show that G_m can be modelled as a multi-parameter weighted voting game using only two weight vectors. The first weight vector is $\mathbf{w}^{(1)} = (1, \dots, 1)$, i.e. it assigns the same weight to all the players. The second weight vector $\mathbf{w}^{(2)}$ assigns weight 0 to player i, for $1 \leq i \leq m$, and assigns weight 1 to player i, for $m+1 \leq i \leq 2m$. The decision function f takes two arguments. A coalition $S \subseteq N$ is winning if and only if $f(w_S^{(1)}, w_S^{(2)}) = 1$ if and only if

$$(w_S^{(1)} \geq m+1) \vee ((w_S^{(1)} = m) \wedge (w_S^{(2)} \equiv 0 \bmod 2)). \tag{7}$$

(Note that the condition $w_S^{(2)} \equiv 0 \bmod 2$ is equivalent to the condition $\#(S \cap \overline{N}_m)$ is even.) So the weight dimension of G_m is 2. This shows that for games with exponentially large dimensions, it is possible the weight dimension is only a small integer.

An example with a single weight vector. We consider an example of a simple game in the spirit of the above example from Taylor and Zwicker (1999), but this time with a single weight vector. Let N be a set of n players and the weights of the players are given by a vector $\mathbf{w} = (w_1, \dots, w_n)$. Let $\sigma = w_1 + \cdots + w_n$. The decision function f defining the winning condition is the following. Let $q, \varepsilon \in (0,1)$ be two real numbers. A coalition $S \subseteq N$ is defined to be winning if and only if $f(w_S) = 1$ if and only if $|w_S/\sigma - q| \leq \varepsilon$. In other words, S is a winning coalition, if and only if its proportional weight is ε-close to q.

Clearly the weight dimension of the above game is 1. On the other hand, obtaining the values of the dimension, co-dimension, and the Boolean dimension seem to be quite difficult and in any case will be greater than 1.

4 Basic recurrences

The crux of the methods to compute the quantities mentioned in Sect. 2.2 are two basic recurrences. In this section, we present these recurrences and in later sections, we show how various quantities can be obtained from these two recurrences.

Consider a k-parameter game $G = (N, \mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}, f)$. From the weight vectors $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}$ we define a $k+1$ dimensional table T, where $T(i, s_1, \ldots, s_k)$, with $1 \leq i \leq n$, being the number of subsets S of $\{1, \ldots, i\}$ with $w_S^{(1)} = s_1, \ldots, w_S^{(k)} = s_k$.

Proposition 3 For $i \geq 2$,

$$T(i, s_1, \ldots, s_k) = T(i-1, s_1, \ldots, s_k) + T(i-1, s_1 - w_i^{(1)}, \ldots, s_k - w_i^{(k)}) \quad (8)$$

with boundary conditions as follows.

1. If $(w_1^{(1)}, \ldots, w_1^{(k)}) \neq (0, \ldots, 0)$, then
   1. $T(1, 0, \ldots, 0) = 1, T(1, w_1^{(1)}, \ldots, w_1^{(k)}) = 1$, and
   2. $T(1, s_1, \ldots, s_k) = 0$ for any $(s_1, \ldots, s_k) \neq (0, \ldots, 0), (w_1^{(1)}, \ldots, w_1^{(k)})$.

3. If $(w_1^{(1)}, \ldots, w_1^{(k)}) = (0, \ldots, 0)$, then
   1. $T(1, 0, \ldots, 0) = 2$, and
   2. $T(1, s_1, \ldots, s_k) = 0$ for any $(s_1, \ldots, s_k) \neq (0, \ldots, 0)$.

Computing the table T using dynamic programming requires time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$. For $i = 2, \ldots, n$, obtaining $T(i, \ldots)$ from $T(i-1, \ldots)$ requires space $O(\sigma^{(1)} \cdots \sigma^{(k)})$.

Proof First we consider the boundary conditions. If $(w_1^{(1)}, \ldots, w_1^{(k)}) \neq (0, \ldots, 0)$, then the only coalition S with $w_S^{(1)} = \cdots = w_S^{(k)} = 0$ is $S = \emptyset$, and the only coalition S with $w_S^{(1)} = w_1^{(1)}, \ldots, w_S^{(k)} = w_1^{(k)}$ is $S = \{1\}$. On the other hand, if $(w_1^{(1)}, \ldots, w_1^{(k)}) = (0, \ldots, 0)$, then the coalitions S such that $w_S^{(1)} = \cdots = w_S^{(k)} = 0$ are $S = \emptyset, \{1\}$. These are covered by the boundary conditions.

The proof of (8) follows from the following fact. Let S be a subset of $\{1,\dots,i\}$. Then $w_S^{(1)} = s_1,\dots,w_S^{(k)} = s_k$ if and only if one of the following two conditions hold:

- either S is a subset of $\{1,\dots,i-1\}$, with $w_S^{(1)} = s_1,\dots,w_S^{(k)} = s_k$, or
- $S \setminus \{i\}$ is a subset of $\{1,\dots,i-1\}$ with $w_S^{(1)} = s_1 - w_i^{(1)},\dots,w_S^{(k)} = s_k - w_i^{(k)}$.

For $1 \le j \le k$, the value of s_j lies in the set $\{0,\dots,\sigma^{(j)}\}$. So for a fixed value of i, the table T has $\sigma^{(1)} \cdots \sigma^{(k)}$ entries. A dynamic programming algorithm will fill up the entries of the complete table T in the following manner. First the entries of $T(1,\dots)$ are filled up using the boundary condition. Now for each $i \in \{2,\dots,n\}$, use (8) to obtain the value of $T(i,s_1,\dots,s_k)$, with $0 \le s_j \le \sigma^{(j)}, 1 \le j \le k$, from the already obtained values of the table for $i-1$. Thus, the time taken for each i is $O(\sigma^{(1)} \cdots \sigma^{(k)})$, and so the time taken to compute the entire table is $O(n\sigma^{(1)} \cdots \sigma^{(k)})$.

To see the expression for the space complexity, note that for any $i \in \{1,\dots,n\}$, once $T(i,s_1,\dots,s_k)$ has been computed, $T(i-1,s_1,\dots,s_k)$ is no longer required and the space to store $T(i-1,s_1,\dots,s_k)$ can be reused. So for any $i \in \{1,\dots,n\}$, space is required to store $T(i-1,s_1,\dots,s_k)$ and $T(i,s_1,\dots,s_k)$. Since $s_1 \le \sigma^{(1)},\dots,s_k \le \sigma^{(k)}$, the statement of the space complexity follows. ☐

The literal interpretation of the recurrence given by (8) is the following. Let S be a coalition which is a subset of the players $\{1,\dots,i\}$ and having weights $s_1,\dots,s_k$ in the k parameters. Then S is counted in $T(i,s_1,\dots,s_k)$. Now either i is in S or i is not in S. Suppose i is in S. Then the weights of $S\setminus\{i\}$ are $s_1 - w_i^{(1)},\dots,s_k - w_i^{(k)}$ and so $S\setminus\{i\}$ is counted in $T(i-1,s_1 - w_i^{(1)},\dots,s_k - w_i^{(k)})$. Now suppose i is not in S. Then S itself is a subset of $\{1,\dots,i-1\}$ and so S is counted in $T(i-1,s_1,\dots,s_k)$.

The value of $T(i,s_1,\dots,s_k)$ is the number of coalitions $S \subseteq \{1,\dots,i\}$ with $w_S^{(1)} = s_1,\dots,w_S^{(k)} = s_k$. Suppose that we also wish to obtain the number of coalitions with specified values of the weights and a specified value of its cardinality. To this end, we define a $k+2$ dimensional table C, where $C(i,c,s_1,\dots,s_k)$, with $1 \le i \le n$, $0 \le c \le i$, is the number of subsets S of $\{1,\dots,i\}$ having cardinality c with $w_S^{(1)} = s_1,\dots,w_S^{(k)} = s_k$.

Proposition 4 For $i \ge 2$ and $1 \le c \le i$,

$$C(i,c,s_1,\dots,s_k) = C(i-1,c,s_1,\dots,s_k) + C(i-1,c-1,s_1 - w_i^{(1)},\dots,s_k - w_i^{(k)}), \tag{9}$$

where for $i \ge 2$, we define $C(i-1,i,s_1,\dots,s_k)$ to be equal to 0. The boundary conditions are as follows.

1. $C(i,0,0,\dots,0) = 1$ for $i \ge 1$,
2. $C(i,0,s_1,\dots,s_k) = 0$ for $i \ge 1$ and $(s_1,\dots,s_k) \ne (0,\dots,0)$,
3. $C(1,1,w_1^{(1)},\dots,w_1^{(k)}) = 1$,

4. $C(1, c, s_1, \ldots, s_k) = 0$, for $(s_1, \ldots, s_k) \neq (0, \ldots, 0), (w_1^{(1)}, \ldots, w_1^{(k)})$.

Computing the table C using dynamic programming requires time $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$. For $i = 2, \ldots, n$, obtaining $C(i, \ldots)$ from $C(i-1, \ldots)$ requires space $O(n\sigma^{(1)} \cdots \sigma^{(k)})$.

Proof The proof of the boundary conditions is similar to that in Proposition 3.

The proof of (9) follows from the following fact. Let S be a subset of $\{1, \ldots, i\}$. Then $\#S = c$ and $w_S^{(1)} = s_1, \ldots, w_S^{(k)} = s_k$ if and only if one of the following two conditions hold:

- either S is a subset of $\{1, \ldots, i-1\}$ with $\#S = c$ and $w_S^{(1)} = s_1, \ldots, w_S^{(k)} = s_k$, or
- $S \setminus \{i\}$ is a subset of $\{1, \ldots, i-1\}$ with $\#S = c - 1$ and $w_S^{(1)} = s_1 - w_i^{(1)}, \ldots, w_S^{(k)} = s_k - w_i^{(k)}$.

The proof of the runtime is similar to that of Proposition 3. For a pair (i, c) there are $O(\sigma^{(1)} \cdots \sigma^{(k)})$ entries in the table C. The dynamic programming algorithm proceeds as follows. First fill up the entries of $C(1, \ldots)$ using the boundary conditions. Next for each (i, c), with $i \geq 2$ and $0 \leq c \leq i$, use (9) to determine the values of $C(i, c, s_1, \ldots, s_k)$ from those found in the previous steps. This requires considering the value $C(i-1, i, \ldots)$ which we have defined to be 0. Thus, for each pair (i, c), the time taken to fill up the table C is $O(\sigma^{(1)} \cdots \sigma^{(k)})$ and so the total time taken is $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$.

The argument for the space complexity is similar to the argument for space complexity in Proposition 3. □

We provide a literal interpretation of the recurrence given by (9). Let S be a coalition which is a subset of the players $\{1, \ldots, i\}$ and having cardinality c and weights $s_1, \ldots, s_k$ in the k parameters. Then S is counted in $C(i, c, s_1, \ldots, s_k)$. Now either i is in S or i is not in S. Suppose i is in S. Then the cardinality of $S \setminus \{i\}$ is $c - 1$ and the weights of $S \setminus \{i\}$ are $s_1 - w_i^{(1)}, \ldots, s_k - w_i^{(k)}$ and so $S \setminus \{i\}$ is counted in $C(i-1, c-1, s_1 - w_i^{(1)}, \ldots, s_k - w_i^{(k)})$. Now suppose i is not in S. Then S itself is a subset of $\{1, \ldots, i-1\}$ of cardinality c and so S is counted in $C(i-1, c, s_1, \ldots, s_k)$.

Remark 3 In the definition of multi-parameter weighted voting games, we have restricted weights to take non-negative values. We briefly discuss how this restriction can be lifted. Suppose we allow the weights to take any real value. For $1 \leq j \leq k$, let $\mu_{\min}^{(j)}$ and $\mu_{\max}^{(j)}$ be such that for any $S \subseteq N$, $\mu_{\min}^{(j)} \leq w_S \leq \mu_{\max}^{(j)}$. An easy value for $\mu_{\max}^{(j)}$ is $\mu_{\max}^{(j)} = |w_1^{(j)}| + \cdots + |w_n^{(j)}|$ and $\mu_{\min}^{(j)} = -\mu_{\max}^{(j)}$. Let $\sigma^{(j)} = \mu_{\max}^{(j)} - \mu_{\min}^{(j)}$.

We proceed under the assumption that appropriate values of $\mu_{\min}^{(j)}$ and $\mu_{\max}^{(j)}$ are known. In the construction of the tables T and C, the dimension corresponding to j will be indexed by values in the set $\{\mu_{\min}^{(j)}, \ldots, \mu_{\max}^{(j)}\}$. The basic recurrences for the tables T and C given by (8) and (9) still hold.

The upper bound for the time and space complexities given in Propositions 3 and 4 in terms of $\sigma^{(j)}$ remain unchanged.

Remark 4 From the definitions of $T(i, s_1, \ldots, s_k)$ and $C(i, c, s_1, \ldots, s_k)$, it follows that if $(s_1, \ldots, s_k) \notin \Omega$, then $T(i, s_1, \ldots, s_k) = C(i, c, s_1, \ldots, s_k) = 0$, where Ω is the domain of the decision function f. In our analysis in Sects. 5 and 6, we obtain sums over tuples $(s_1, \ldots, s_k)$, where the individual terms of the sum are products of $f(s_1, \ldots, s_k)$ with other quantities. While taking the sums, we will not restrict the tuples to be in Ω. This will mean applying f to tuples outside Ω. Formally, this is handled by extending the domain of f, where for $(s_1, \ldots, s_k) \notin \Omega$ the value of $f(s_1, \ldots, s_k)$ is arbitrarily set to be either 0 or 1. This will not cause any problem, since in all such cases we will show that the corresponding terms in the sum will turn out to be 0.

5 Winning coalitions, blockers and swings

In this section, we use the results of Sect. 4 to show how the following quantities can be computed: number of winning coalitions, number of winning coalitions containing a particular player, determining whether a player is a blocker, the number of coalitions in which a player is a swing, and the number of coalitions of a particular cardinality in which a player is a swing.

Proposition 5 The number of winning coalitions in the game $G = (N, \mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}, f)$ is given by the following formula.

$$\#W_G = \sum_{\substack{0 \le s_1 \le \sigma^{(1)}, \\ \cdots \\ 0 \le s_k \le \sigma^{(k)}}} f(s_1, \ldots, s_k) \cdot T(n, s_1, \ldots, s_n). \tag{10}$$

Proof First note that if $(s_1, \ldots, s_k) \notin \Omega$, then $T(n, s_1, \ldots, s_k) = 0$ and so the product $f(s_1, \ldots, s_k) \cdot T(n, s_1, \ldots, s_k)$ is also equal to 0 (see Remark 4).

$T(n, s_1, \ldots, s_k)$ is the number of coalitions $S \subseteq \{1, \ldots, n\}$ such that $w_S^{(1)} = s_1, \ldots, w_S^{(k)} = s_k$. Any such coalition S is a winning coalition if and only if $f(s_1, \ldots, s_k) = 1$. So the right hand side of (10) counts the number of winning coalitions in G. □

For $i \in N$, let $W_G(i)$ be the set of winning coalitions containing the player i. Next we consider the cardinality of $W_G(i)$. The players and the weight vectors can be reordered so that the player under consideration can be considered to be n. So it is sufficient to find the number of winning coalitions containing the player n. This is given by the following result.

Proposition 6 In the game $G = (N, \mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(k)}, f)$, the number of winning coalitions containing the n-th player is given by the following relation.

$$\#W_G(n) = \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_n^{(1)}, \\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_n^{(k)}}} f(s_1 + w_n^{(1)}, \dots, s_k + w_n^{(k)}) \cdot T(n-1, s_1, \dots, s_k). \tag{11}$$

Proof Let $\Omega_{|_i}$ be the restriction of Ω to the k-tuples arising from coalitions of the players $\{1, \dots, i\}$. If $(s_1 + w_n^{(1)}, \dots, s_k + w_n^{(k)}) \notin \Omega$, then it follows that $(s_1, \dots, s_k) \notin \Omega_{|_{n-1}}$ and so $T(n-1, s_1, \dots, s_k) = 0$ (see Remark 4).

The requirement is to count the number of coalitions $S \subseteq \{1, \dots, n-1\}$ such that $S \cup \{n\}$ is winning. Since S is a subset of $\{1, \dots, n-1\}$, we have $w_S^{(j)} \le \sigma^{(j)} - w_n^{(j)}$, $1 \le j \le k$. So it is sufficient to consider k tuples $(s_1, \dots, s_k)$ with $0 \le s_j \le \sigma^{(j)} - w_n^{(j)}$, $j = 1, \dots, k$. The number of subsets S of $\{1, \dots, n-1\}$ such that $w_S^{(1)} = s_1, \dots, w_S^{(k)} = s_k$ is equal to $T(n-1, s_1, \dots, s_k)$. These subsets are included in the count if $S \cup \{n\}$ is winning, i.e. if $f(s_1 + w_n^{(1)}, \dots, s_k + w_n^{(k)}) = 1$. □

We next consider the characterisation of whether a player is a blocker or not. Player i is a blocker if and only if any coalition not containing i is a losing coalition. A coalition not containing i has weights $s_1, \dots, s_k$ in the k parameters with $(s_1, \dots, s_k) \in \Omega$ such that $s_1 \le \sigma^{(1)} - w_i^{(1)}, \dots, s_k \le \sigma^{(k)} - w_i^{(k)}$. So i is a blocker if and only if $f(s_1, \dots, s_k) = 0$ for all $(s_1, \dots, s_k) \in \Omega$ such that $s_1 \le \sigma^{(1)} - w_i^{(1)}, \dots, s_k \le \sigma^{(k)} - w_i^{(k)}$. This is equivalently stated as

$$\sum_{\substack{(s_1, \dots, s_k) \in \Omega, \\ s_1 \le \sigma^{(1)} - w_i^{(1)}, \\ \dots \\ s_k \le \sigma^{(k)} - w_i^{(k)}}} f(s_1, \dots, s_k) = 0. \tag{12}$$

The characterisation (12) is not computationally useful since the condition $(s_1, \dots, s_k) \in \Omega$ needs to be checked. The table $T()$ provides this information. As before, by reordering players and weight vectors we may consider the player under consideration to be n. The player n is a blocker if and only if $\#W_G = \#W_G(n)$, which can be determined from Propositions 5 and 6. Alternatively, player n is a blocker if and only if the number of winning coalitions not containing player n is equal to 0. This is characterised by the following result.

Proposition 7 In the game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, the player n is a blocker if and only if

$$\sum_{\substack{s_1 \le \sigma^{(1)} - w_n^{(1)}, \\ \dots \\ s_k \le \sigma^{(k)} - w_n^{(k)}}} f(s_1, \dots, s_k) T(n-1, s_1, \dots, s_k) = 0. \tag{13}$$

For $1 \le j \le k$, we have the weight vector $\mathbf{w}^{(j)} = (w_1^{(j)}, \dots, w_n^{(j)})$. For $1 \le i \le n$, define

$$\mathbf{w}^{(j,i)} = (w_1^{(j)}, \dots, w_{i-1}^{(j)}, w_{i+1}^{(j)}, \dots, w_n^{(j)}).$$

In other words, $\mathbf{w}^{(j,i)}$ is obtained from $\mathbf{w}^{(j)}$ by dropping the component corresponding to i. Recall that the table T is prepared from the weight vectors $\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}$. For $1 \le i \le n$, define $T^{(i)}$ to be the table corresponding to (8) prepared from the weight vectors $\mathbf{w}^{(1,i)}, \dots, \mathbf{w}^{(k,i)}$. This corresponds to considering a game for the players $N \setminus \{i\}$ with the weight vectors $\mathbf{w}^{(1,i)}, \dots, \mathbf{w}^{(k,i)}$.

Proposition 8 In the game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, the values of m_i^+ and m_i^- for a player i are given by

$$\begin{aligned} m_i^+ &= \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)},\\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} \left(1 - f(s_1, \dots, s_k)\right) \cdot f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) \cdot T^{(i)}(n-1, s_1, \dots, s_k), \\ m_i^- &= \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)},\\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} f(s_1, \dots, s_k) \cdot \left(1 - f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)})\right) \cdot T^{(i)}(n-1, s_1, \dots, s_k). \end{aligned} \tag{14}$$

Further, if G is monotone, then

$$\begin{aligned} m_i^+ &= \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)},\\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} \left(f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) - f(s_1, \dots, s_k)\right) \cdot T^{(i)}(n-1, s_1, \dots, s_k), \\ m_i^- &= 0. \end{aligned} \tag{15}$$

Proof We provide the argument for m_i^+, the argument for m_i^- being similar.

We first argue that if either $(s_1, \dots, s_k) \notin \Omega$ or $(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) \notin \Omega$, then $T^{(i)}(n-1, s_1, \dots, s_k) = 0$ and so the corresponding terms in (14) are zero (see Remark 4). Let $\Omega^{(i)} = \{(s_1, \dots, s_k) \in \Omega : (s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) \in \Omega\}$. It follows that if $(s_1, \dots, s_k) \notin \Omega^{(i)}$, then $T^{(i)}(n-1, s_1, \dots, s_k) = 0$. Now note that if either $(s_1, \dots, s_k) \notin \Omega$ or $(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) \notin \Omega$, then $(s_1, \dots, s_k) \notin \Omega^{(i)}$ and so $T^{(i)}(n-1, s_1, \dots, s_k) = 0$.

In the modified game obtained by dropping player i, there are $n-1$ players and $T^{(i)}(n-1, s_1, \dots, s_k)$ counts the number of coalitions $S \subseteq \{1, \dots, n\} \setminus \{i\}$ in this modified game such that $w_S^{(1)} = s_1, \dots, w_S^{(k)} = s_k$. Player i is a swing in the coalition $S \cup \{i\}$ in G if S is losing in G and $S \cup \{i\}$ is winning in G. So i is a swing in $S \cup \{i\}$ in G if $f(s_1, \dots, s_k) = 0$ and $f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) = 1$. The last condition is equivalent to $\left(1 - f(s_1, \dots, s_k)\right) \cdot f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) = 1$. So the right hand side of (14) counts the number of coalitions in which player i is a swing in the game G.

The statement on monotone functions follows from (1). □

Let $C^{(i)}$ be the table corresponding to (9) prepared from the weight vectors $\mathbf{w}^{(1,i)}, \dots, \mathbf{w}^{(k,i)}$. The following result provides the number of coalitions of a particular cardinality in which player i is a swing. The proof is similar to the proof of Proposition 8 and so we skip it.

Proposition 9 In the game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, the values of $m^+_{i,c}$ and $m^-_{i,c}$ for a player i are given by

$$\begin{aligned}
m^+_{i,c} &= \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)},\\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} \left(1 - f(s_1, \dots, s_k)\right) \cdot f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) \cdot C^{(i)}(n-1, c-1, s_1, \dots, s_k),\\
m^-_{i,c} &= \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)},\\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} f(s_1, \dots, s_k) \cdot \left(1 - f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)})\right) \cdot C^{(i)}(n-1, c-1, s_1, \dots, s_k).
\end{aligned} \tag{16}$$

Further, if G is monotone, then

$$\begin{aligned}
m^+_{i,c} &= \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)},\\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} \left(f(s_1 + w_i^{(1)}, \dots, s_k + w_i^{(k)}) - f(s_1, \dots, s_k)\right) \cdot C^{(i)}(n-1, c-1, s_1, \dots, s_k),\\
m^-_{i,c} &= 0.
\end{aligned} \tag{17}$$

5.1 Computing statistics for all players

The above results are for a particular player, i.e., for a fixed player, the results show how to compute the number of winning coalitions containing that player, whether the player is a blocker, the number of coalitions in which the player is a swing, and the number of coalitions of a particular cardinality in which the player is a swing. We consider the time and space complexities of these algorithms and consider how to extend them to compute the relevant statistics for all the players.

From Proposition 6, computing the number of winning coalitions $\#W_G(n)$ containing player n requires computing the table T, in particular the entries in $T(n-1, \dots)$. From Proposition 3, this can be done in $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ time and $O(\sigma^{(1)} \cdots \sigma^{(k)})$ space. To compute the number of winning coalitions for any player, we need to reorder the players so that the particular player is in the n-th position and apply Proposition 6. So to compute $\#W_G(i)$, for $i = 1, \dots, n$, one after the other, the time and space complexities are $O(n^2\sigma^{(1)}\cdots\sigma^{(k)})$ and $O(\sigma^{(1)}\cdots\sigma^{(k)})$ respectively.

Similarly, from Proposition 7, determining whether player n is a blocker requires the entries in $T(n-1,\ldots)$. So as above, to determine the set of blockers in the game one by one requires time $O(n^2\sigma^{(1)}\cdots\sigma^{(k)})$ and space $O(\sigma^{(1)}\cdots\sigma^{(k)})$.

From Proposition 8, for any designated player $i\in N$, computing m_i^+, m_i^- and $m_i = m_i^+ + m_i^-$ require computing the table $T^{(i)}$, in particular the entries in $T^{(i)}(n-1,\ldots)$. From Proposition 3, this can be done in $O(n\sigma^{(1)}\cdots\sigma^{(k)})$ time and $O(\sigma^{(1)}\cdots\sigma^{(k)})$ space. So to compute the vectors $\mathbf{m}^+$, $\mathbf{m}^-$ and $\mathbf{m}$, $T^{(i)}(n-1,\ldots)$ has to be computed for $i=1,\ldots,n$. Consequently, a sequential algorithm to compute these vectors require total time $O(n^2\sigma^{(1)}\cdots\sigma^{(k)})$ and space $O(\sigma^{(1)}\cdots\sigma^{(k)})$.

Similarly, from Proposition 9, for any designated player $i\in N$, and $0\le c\le n$, computing $m_{i,c}^+$, $m_{i,c}^-$ and $m_{i,c} = m_{i,c}^+ + m_{i,c}^-$ require computing the table $C^{(i)}$, in particular the entries in $C^{(i)}(n-1,\ldots)$. From Proposition 4, this can be done in $O(n^2\sigma^{(1)}\cdots\sigma^{(k)})$ time and $O(n\sigma^{(1)}\cdots\sigma^{(k)})$ space. So to compute the matrices $\mathbf{M}^+$, $\mathbf{M}^-$ and $\mathbf{M}$, $C^{(i)}(n-1,\ldots)$ has to be computed for $i=1,\ldots,n$. Again, a sequential algorithm to compute these matrices require total time $O(n^3\sigma^{(1)}\cdots\sigma^{(k)})$ and space $O(n\sigma^{(1)}\cdots\sigma^{(k)})$.

Parallelism. Using parallelism, it is possible to obtain a trade-off between time and space. The time complexity decreases by a factor of n, while the space complexity increases by a factor of n.

First consider the case of $\#W_G(i)$, for $i=1,\ldots,n$. This requires constructing n tables where in the i-th table, player i occurs as the last player. (Note that these tables consist of all the n players and should not be confused with table $T^{(i)}$, which is constructed by dropping the i-th player.) From the i-th table it is also possible to determine whether player i is a blocker. The opportunity for parallelism arises from the following simple observation. The construction of the n tables are independent. As a result, it is possible to set up n processes to compute the n tables. Each process requires sequential time $O(n\sigma^{(1)}\cdots\sigma^{(k)})$ and space $O(\sigma^{(1)}\cdots\sigma^{(k)})$. The i-th process computes $\#W_G(i)$ and determines whether i is a blocker. The parallel time required by the n processes is $O(n\sigma^{(1)}\cdots\sigma^{(k)})$. Since all the processes work in parallel, they require their own memory, and so the total space required by all the processes is $O(n\sigma^{(1)}\cdots\sigma^{(k)})$.

Next consider the case of computing the vectors $\mathbf{m}^+$, $\mathbf{m}^-$ and $\mathbf{m}$. Again, the opportunity for parallelism arises from the fact that the constructions of the tables $T^{(1)},\ldots,T^{(n)}$ are independent. So n processes can work in parallel to construct these n tables, with the i-th process computing table $T^{(i)}$ and then computing m_i^+, m_i^- and m_i. As a result, the parallel time taken by these n processes is $O(n\sigma^{(1)}\cdots\sigma^{(k)})$, and each process requires space $O(\sigma^{(1)}\cdots\sigma^{(k)})$, for a total space complexity of $O(n\sigma^{(1)}\cdots\sigma^{(k)})$.

Similarly, to compute the matrices $\mathbf{M}^+, \mathbf{M}^-$ and $\mathbf{M}$, one may set up n processes, where the i-th process computes the table $C^{(i)}$ and hence the i-th row of the matrices $\mathbf{M}^+, \mathbf{M}^-$ and $\mathbf{M}$. The parallel time required by the n processes in this case is $O(n^2\sigma^{(1)}\cdots\sigma^{(k)})$ and each process requires space $O(n\sigma^{(1)}\cdots\sigma^{(k)})$ for a total space complexity of $O(n^2\sigma^{(1)}\cdots\sigma^{(k)})$.

A requirement for the above parallelism is that it should be possible to execute n processes in parallel, i.e. the computer server on which the processes will run should have at least n cores. Present day computer servers typically have a few hundred cores, so the above parallel algorithms can be executed on such servers for games where the number of players is a hundred or so. The space complexities increase by a factor of n. The main determining factor in the space complexity is the term $\sigma^{(1)} \cdots \sigma^{(k)}$. As long as this quantity is within reasonable bounds, increase of space by a factor of n is not much of a problem.

Improved sequential algorithm in special cases. Coming back to the case of sequential algorithms, for the case of weighted majority games (i.e. where there is only one weight vector) Uno (2012) proposed elegant sequential algorithms to compute the vector $\mathbf{m}$ in time $O(n\sigma^{(1)})$ and space $O(\sigma^{(1)})$ and the matrix $\mathbf{M}$ in time $O(n^2\sigma^{(1)})$ and space $O(n\sigma^{(1)})$. This improves the time complexity of the sequential algorithm by a factor of n. This method was adapted by Wilms (2020) to the special cases of weighted AND and weighted OR games, where the sequential time and space complexities for computing $\mathbf{m}$ were shown to be $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ and $O(\sigma^{(1)} \cdots \sigma^{(k)})$ respectively, while the sequential time and space complexities for computing $\mathbf{M}$ were shown to be $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ and $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ respectively. In Appendix A, we describe how the general framework that we have introduced can be specialised to obtain the results by Wilms. In the appendix we also discuss how the approach can be applied to computing $\#W_G(i)$ for $i = 1, \ldots, n$ and to compute the set of blockers in the game. These quantities were not considered by Wilms (2020).

Remark 5 For weighted AND and weighted OR games, the k in the expressions for time and space complexities of the corresponding algorithms described by Wilms (2020) is the number of constituent weighted majority games. For the case of weighted AND games, since it is difficult to determine the dimension of a game, it is very likely that k is (significantly) greater than the dimension of the particular game. In contrast, the k in the expressions for time and space complexities in the algorithms that we describe is the number of weight vectors in the representation of the game as a multi-parameter weighted voting game. The number of weight vectors in a multi-parameter weighted voting representation of a game G can be much smaller than the number of constituent games in a representation of G as a weighted AND game (see Sect. 3.3 for an example). So in general our algorithms will be faster than that of Wilms for the same game.

Wilms (2020) (see Section 5) also considered the case where a monotone simple game is presented in the conjunctive normal form, where there are ρ conjuncts and each conjunct is a weighted OR game. For this case, Wilms showed that $\mathbf{m}$ can be computed in $O(n2^\rho\sigma^{(1)} \cdots \sigma^{(k)})$ time and $O(\sigma^{(1)} \cdots \sigma^{(k)})$ space. Similarly, it was shown in Wilms (2020) that computing $\mathbf{M}$ can be done in $O(n^2 2^\rho\sigma^{(1)} \cdots \sigma^{(k)})$ and $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ space. The corresponding time complexities for computing $\mathbf{m}$ and $\mathbf{M}$ by our sequential algorithms are $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ and

$O(n^3\sigma^{(1)}\cdots\sigma^{(k)})$ respectively. These expressions are greater than the expressions obtained by Wilms if and only if $2^\rho < n$.

Remark 6 The k in the above expressions for the time complexities coming from Section 5 of Wilms (2020) is the number of constituent games in the Boolean combination of these games, while the k in the expressions for the complexities of the algorithms that we present is the number of weight vectors in the representation of the game as a multi-parameter weighted voting game. The number of weight vectors in the multi-parameter weighted voting representation of a game G is in general smaller than the number of constituent games in the representation of G as a Boolean combination of games. (See Sect. 3.3 for an example.) So in general our algorithms will be faster than that of Wilms for the same game.

We note that Section 6 of Wilms (2020) performs an analysis which is specific to the EU voting games. The time complexities resulting from this analysis is smaller than the time complexities that we obtain. So for computing $\mathbf{m}$ and $\mathbf{M}$ for the EU voting game, our algorithms do not improve upon the algorithms in Section 6 of Wilms (2020). Instead, for the EU voting game, our work provides algorithms for computing other statistics which were not considered in Wilms (2020) and other previous works. See Remark 7.

Avoiding exponential blow-up. More generally, our approach (both sequential and parallel) for computing $\mathbf{m}^+, \mathbf{m}^-, \mathbf{m}$ and $\mathbf{M}^+, \mathbf{M}^-, \mathbf{M}$ applies for the class of all multi-parameter weighted voting games. Beyond the special cases of weighted AND games, weighted OR games and games presented in the conjunctive normal form, there is no previous work which provides algorithms for computing $\mathbf{m}^+, \mathbf{m}^-, \mathbf{m}$ and $\mathbf{M}^+, \mathbf{M}^-, \mathbf{M}$ for any other form of games with more than one weight vector. Given a simple game G, one may obtain a representation of G using a weighted AND game and then apply previous algorithms for weighted AND games to compute the required quantities. The problem with this approach is that the number of constituent games in the weighted AND game representation can be exponential in the number of players even when the number of weight vectors in the representation as a multi-parameter weighted voting game is a small integer (see Sect. 3.3 for an example). So converting to weighted AND representation and applying previous algorithms can result in an exponential blow up in the time and space complexities.

Usefulness of the tables $T^{(i)}$ and $C^{(i)}$. Wilms' approach avoids computing the tables $T^{(i)}$ and $C^{(i)}$. We note that these tables are also required for computing the vectors $\boldsymbol{\gamma}$ and $\boldsymbol{\Gamma}$ related to minimal winning coalitions. This is discussed in Sect. 6.1. So the usefulness of computing these tables extend beyond computing $\mathbf{m}$ and $\mathbf{M}$. See Remark 8.

6 Minimal winning coalitions

In this section, we put restrictions on the weight vectors and the decision function *f*. The condition on the weight vectors is that they must be simultaneously decreasing, i.e.,

$$w_1^{(1)} \geq \cdots \geq w_n^{(1)},$$
$$\cdots$$
$$w_1^{(k)} \geq \cdots \geq w_n^{(k)}.$$

The restriction on *f* is that it must be monotone.

The above two conditions simplify the task of counting minimal winning coalitions. Suppose S is a winning coalition. To be a minimal winning coalition it is required that dropping any player from S results in a losing coalition. Let $i = \max S$ and suppose that dropping i from S results in a losing coalition. By the above two conditions, it follows that dropping ℓ from S, with $\ell < i$, instead of i also results in a losing coalition. So to check whether S is a minimal winning coalition, it is sufficient to check whether dropping the highest numbered player from S results in a losing coalition. While the two conditions simplify the task of counting minimal winning coalitions, we do not know whether they are necessary for being able to efficiently compute the number of minimal winning coalitions.

The above conditions on the weight vectors and the decision rule are satisfied by the pre-Brexit Lisbon voting rules in the EU Council. As described in Sect. 3.3, this voting rule can be modelled as a 2-parameter game. The first parameter corresponds to the cardinality of a coalition and the weight vector $\mathbf{w}^{(1)}$ has the entry 1 for all the nations. The second parameter corresponds to the total size of the population of a coalition and the weight vector $\mathbf{w}^{(2)}$ records the populations of the nations. In this game $k = 2$ and $w_1^{(1)} = \cdots = w_n^{(1)} = 1$. So simply arranging the components of $\mathbf{w}^{(2)}$ in descending order ensures that the simultaneous decreasing condition on the weight vectors hold. Further, since f is a monotone Boolean combination of thresholding functions, the property of being monotone on each component also holds for *f*.

We now turn to the problem of determining the number of minimal winning coalitions in a general multi-parameter weighted voting game. In the game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, for $1 \leq i \leq n$, let E_i be the number of minimal winning coalitions S such that i is in S and $S \subseteq \{1, \dots, i\}$. Then

$$\#\mathsf{MW}_G = E_1 + \cdots + E_n. \tag{18}$$

So to obtain the number of minimal winning coalitions in G it is sufficient to obtain E_i.

Proposition 10 In the game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, the value of E_i, $1 \leq i \leq n$, is given by

$$\sum_{\substack{0 \le s_1 \le \sigma_i^{(1)}, \\ \cdots \\ 0 \le s_k \le \sigma_i^{(k)}}} \left(f(s_1, \dots, s_k) - f(s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)})\right) \cdot T(i-1, s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}), \tag{19}$$

where for $1 \le j \le k, \sigma_i^{(j)} = w_1^{(j)} + \cdots + w_i^{(j)}$.

Proof For the computation of E_i, it is sufficient to consider only the set of players $\{1, \dots, i\}$. Let $\Omega_{|_i}$ be as defined in the proof of Proposition 6.

So in (19), f is applied to k-tuples in $\Omega_{|_i}$. Now note that if either $(s_1, \dots, s_k) \notin \Omega_{|_i}$, or $(s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}) \notin \Omega_{|_i}$, then $(s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}) \notin \Omega_{|_{i-1}}$ which implies that $T(i-1, s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}) = 0$ and so the corresponding terms in (19) are 0 (see Remark 4).

The coalitions S such that $i \in S$, $S \subseteq \{1, \dots, i\}$ and $w_S^{(1)} = s_1, \dots, w_S^{(k)} = s_k$ are counted in the expression $T(i, s_1, \dots, s_k) - T(i-1, s_1, \dots, s_k)$ which is equal to $T(i-1, s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)})$ (see (8)). Now suppose $f(s_1, \dots, s_k) = 1$ and $f(s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}) = 0$, i.e., S is a winning coalition and dropping i from S results in a losing coalition. By the simultaneous decreasing property of the weight vectors and the component-wise monotone property of f it follows that dropping any player ℓ with $\ell < i$ instead of i also results in a losing coalition. So such an S is a minimal winning coalition and is counted in E_i. This shows that E_i is equal to

$$\sum_{\substack{0 \le s_1 \le \sigma_i^{(1)}, \\ \cdots \\ 0 \le s_k \le \sigma_i^{(k)}}} f(s_1, \dots, s_k) \cdot \left(1 - f(s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)})\right) \cdot T(i-1, s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}). \tag{20}$$

Since f is monotone, the expression for E_i given by (19) follows from (20) and (1). □

Next we consider the number of minimal winning coalitions containing a fixed player i, i.e. the value of γ_i. The minimal winning coalitions counted in E_i certainly contain i and the minimal winning coalitions counted in E_ℓ for $\ell < i$ certainly do not contain i. It is, however, possible that for $\ell > i$, a minimal winning coalition counted in E_ℓ also contains i. For $\ell > i$, let $E_{i,\ell}$ be the number of minimal winning coalitions S such that $i, \ell \in S$ and $S \subseteq \{1, \dots, \ell\}$. Then

$$\gamma_i = E_i + E_{i,i+1} + \cdots + E_{i,n}. \tag{21}$$

Proposition 10 already shows how to obtain E_i. The next result shows how to obtain $E_{i,\ell}$ for $\ell = i+1, \dots, n$.

Proposition 11 In the game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, the value of $E_{i,\ell}$, for $1 \le i \le n$ and $i+1 \le \ell \le n$, is given by

$$\sum_{\substack{0\le s_1\le \sigma_\ell^{(1)}-w_i^{(1)},\\ \dots\\ 0\le s_k\le \sigma_\ell^{(k)}-w_i^{(k)}}} \Big(f(s_1',\dots,s_k')-f(s_1'-w_\ell^{(1)},\dots,s_k'-w_\ell^{(k)})\Big)\cdot T^{(i)}(\ell-1,s_1-w_\ell^{(1)},\dots,s_k-w_\ell^{(k)}), \tag{22}$$

where $s_1' = s_1 + w_i^{(1)},\dots,s_k' = s_k + w_i^{(k)}$.

Proof For the computation of $E_{i,\ell}$ it is sufficient to restrict to the set of players $\{1,\dots,\ell\}$. Let $\Omega_{|_\ell}$ be defined as in the proof of Proposition 6 and $\Omega^{(i)}_{|_\ell}$ be defined from $\Omega_{|_\ell}$ in a manner similar to the definition of $\Omega^{(i)}$ from Ω in the proof of Proposition 8. We note that $(s_1+w_i^{(1)},\dots,s_k+w_i^{(k)})\notin\Omega_{|_\ell}$ if and only if $(s_1+w_i^{(1)}-w_\ell^{(1)},\dots,s_k+w_i^{(k)}-w_\ell^{(k)})\notin\Omega_{|_{\ell-1}}$. Further, if $(s_1+w_i^{(1)}-w_\ell^{(1)},\dots,s_k+w_i^{(k)}-w_\ell^{(k)})\notin\Omega_{|_{\ell-1}}$, then $(s_1-w_\ell^{(1)},\dots,s_k-w_\ell^{(k)})\notin\Omega^{(i)}_{|_{\ell-1}}$ implying that $T^{(i)}(\ell-1,s_1-w_\ell^{(1)},\dots,s_k-w_\ell^{(k)})=0$. Consequently, if either $(s_1+w_i^{(1)},\dots,s_k+w_i^{(k)})\notin\Omega_{|_\ell}$ or $(s_1+w_i^{(1)}-w_\ell^{(1)},\dots,s_k+w_i^{(k)}-w_\ell^{(k)})\notin\Omega_{|_{\ell-1}}$, then the corresponding terms in (22) are 0 (see Remark 4).

Recall that the table $T^{(i)}$ is constructed from the weight vectors obtained by dropping the player i. The number of coalitions $S\subseteq\{1,\dots,\ell\}\setminus\{i\}$ containing ℓ such that $w_S^{(1)}=s_1,\dots,w_S^{(k)}=s_k$ is given by $T^{(i)}(\ell,s_1,\dots,s_k)-T^{(i)}(\ell-1,s_1,\dots,s_k)$ which is equal to $T^{(i)}(\ell-1,s_1-w_\ell^{(1)},\dots,s_k-w_\ell^{(k)})$. By the simultaneous decreasing condition on the weights and the component-wise monotone property of f, such a coalition is counted in $E_{i,\ell}$ if and only if $S\cup\{i\}$ is winning and $(S\cup\{i\})\setminus\{\ell\}$ is losing. The last condition is equivalent to $f(s_1+w_i^{(1)},\dots,s_k+w_i^{(k)})=1$ and $f(s_1+w_i^{(1)}-w_\ell^{(1)},\dots,s_k+w_i^{(k)}-w_\ell^{(k)})=0$. So we have $E_{i,\ell}$ to be equal to

$$\sum_{\substack{0\le s_1\le \sigma_\ell^{(1)}-w_i^{(1)},\\ \dots\\ 0\le s_k\le \sigma_\ell^{(k)}-w_i^{(k)}}} f(s_1',\dots,s_k')\Big(1-f(s_1'-w_\ell^{(1)},\dots,s_k'-w_\ell^{(k)})\Big)\cdot T^{(i)}(\ell-1,s_1-w_\ell^{(1)},\dots,s_k-w_\ell^{(k)}). \tag{23}$$

Since f is monotone, the expression for $E_{i,\ell}$ given by (22) follows from (23) and (1). □

Next we consider the cardinalities of the minimal winning coalitions containing a player i. Let $F_{i,c}$ be the number of minimal winning coalitions S such that $i\in S$, $\#S=c$ and $S\subseteq\{1,\dots,i\}$. Similarly, for $\ell>i$, let $F_{i,\ell,c}$ be the number of minimal winning coalitions S such that $\#S=c, i,\ell\in S$ and $S\subseteq\{1,\dots,\ell\}$. So by an argument similar to the one for the number of minimal winning coalitions containing i, we have that the number of minimal winning coalitions of cardinality c containing player i is

$$\gamma_{i,c}=F_{i,c}+F_{i,i+1,c}+\cdots+F_{i,n,c}. \tag{24}$$

So to obtain the number of minimal winning coalitions of a particular cardinality c containing a player i, it suffices to obtain $F_{i,c}$ and $F_{i,\ell,c}$ for $\ell > i$. These expressions are obtained in a manner similar to that for E_i and $E_{i,\ell}$ and are stated in the following result.

Proposition 12 In the game $G = (N, \mathbf{w}^{(1)}, \dots, \mathbf{w}^{(k)}, f)$, the value of $F_{i,c}$, for $1 \le i \le n$ and $0 \le c \le n$, is given by

$$\sum_{\substack{0 \le s_1 \le \sigma_i^{(1)},\\ \dots\\ 0 \le s_k \le \sigma_i^{(k)}}} \left(f(s_1, \dots, s_k) - f(s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)})\right) \cdot C(i-1, c-1, s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}). \tag{25}$$

Further, the value of $F_{i,\ell,c}$, for $1 \le i \le n, i+1 \le \ell \le n$ and $0 \le c \le n$, is given by

$$\sum_{\substack{0 \le s_1 \le \sigma_\ell^{(1)} - w_i^{(1)},\\ \dots\\ 0 \le s_k \le \sigma_\ell^{(k)} - w_i^{(k)}}} \left(f(s'_1, \dots, s'_k) - f(s'_1 - w_\ell^{(1)}, \dots, s'_k - w_\ell^{(k)})\right) \cdot C^{(i)}(\ell-1, c-1, s_1 - w_\ell^{(1)}, \dots, s_k - w_\ell^{(k)}), \tag{26}$$

where $s'_1 = s_1 + w_i^{(1)}, \dots, s'_k = s_k + w_i^{(k)}$.

6.1 Computing #MW, $\boldsymbol{\gamma}$ and $\boldsymbol{\Gamma}$

From (18), computing #MW reduces to the problem of computing E_i for $i = 1, \dots, n$. From Proposition 10, computing E_i requires the computation of the table T. From Proposition 3, the table T can be computed in time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(\sigma^{(1)} \cdots \sigma^{(k)})$. From the values of E_i, #MW can be computed in $O(n)$ time. So #MW can be computed in time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(\sigma^{(1)} \cdots \sigma^{(k)})$.

From (21), γ_i, $1 \le i \le n$, can be computed from E_i and $E_{i,\ell}$, for $\ell = i+1, \dots, n$. From Proposition 11, the computation of $E_{i,\ell}$ requires the computation of $T^{(i)}$. The computation of $T^{(i)}$ is similar to the computation of T and hence from Proposition 3, the computation of $T^{(i)}$ can be done in time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(\sigma^{(1)} \cdots \sigma^{(k)})$. From E_i and $E_{i,\ell}$, for $\ell = i+1, \dots, n$, the computation of γ_i can be done in $O(n)$ time. So the computation of γ_i, for any particular i can be done in time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(\sigma^{(1)} \cdots \sigma^{(k)})$. By sequentially repeating the procedure for all $i \in \{1, \dots, n\}$, the computation of the vector $\boldsymbol{\gamma}$ requires time $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(\sigma^{(1)} \cdots \sigma^{(k)})$.

From (24), the computation of $\gamma_{i,c}$, $1 \le i \le n$ and $0 \le c \le n$, requires the computation of $F_{i,c}$ and $F_{i,\ell,c}$ for $\ell = i+1, \dots, n$. From Proposition 12, the computation of $F_{i,c}$ for $1 \le i \le n$ and for all $c \in \{0, \dots, n\}$ requires the computation of the table C, while for any particular i the computation of $F_{i,\ell,c}$ for all $c \in \{0, \dots, n\}$ requires the computation of the table $C^{(i)}$. From Proposition 4, the computation of C or $C^{(i)}$ requires time $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(n\sigma^{(1)} \cdots \sigma^{(k)})$. For any particular i, using $F_{i,c}$ and $F_{i,c}$ for $1 \leq i \leq n$, the computation of $\gamma_{i,c}$ for all $c \in \{0, \dots, n\}$ requires $O(n^2)$ time. So for any particular i, the computation of $\gamma_{i,c}$ for all $c \in \{0, \dots, n\}$ requires time $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(n\sigma^{(1)} \cdots \sigma^{(k)})$. By sequentially repeating the procedure for computing $C^{(i)}$ for $i = 1, \dots, n$, the computation of $\mathbf{\Gamma}$ can be done in time $O(n^3\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(n\sigma^{(1)} \cdots \sigma^{(k)})$.

For the computation of $\boldsymbol{\gamma}$ and $\mathbf{\Gamma}$, the above description considers sequential algorithms. In a manner similar to the computation of $\mathbf{m}$ and $\mathbf{M}$, parallel algorithms provide a factor of n speed-up at a cost of increasing space by a factor of n. For computing $\boldsymbol{\gamma}$, the idea is to have a total of $n + 1$ parallel processes. One process computes the table T and hence obtains the values $E_1, \dots, E_n$. Each of the other n processes computes exactly one of the tables $T^{(i)}$ and obtains the values of $E_{i,\ell}$ for $\ell = i + 1, \dots, n$. So each of the processes requires time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(\sigma^{(1)} \cdots \sigma^{(k)})$. This provides all the values $E_1, \dots, E_n$ and $E_{i,\ell}$ for $1 \leq i \leq n$ and $\ell = i + 1, \dots, n$. All of these values can be put together in $O(n^2)$ time to obtain $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_n)$. So the total time required for the entire procedure consists of the parallel time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ required by each of the processes plus the final $O(n^2)$ time at the end, for a total of $O(n^2 + n\sigma^{(1)} \cdots \sigma^{(k)})$ time. Since the processes run in parallel and each process requires $O(\sigma^{(1)} \cdots \sigma^{(k)})$ space, all the $n + 1$ processes require $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ space in total.

In a similar manner, the computation of $\mathbf{\Gamma}$ can be done using $n + 1$ parallel processes, where the first process computes the table C (and hence the values of $F_{i,c}$ for $1 \leq i \leq n$ and $0 \leq c \leq n$), while each of the other n processes computes exactly one of the tables $C^{(i)}$ (and hence the values of $F_{i,\ell,c}$ for $i + 1 \leq \ell \leq n$ and $0 \leq c \leq n$). Each process requires time $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ and space $O(n\sigma^{(1)} \cdots \sigma^{(k)})$. The outputs $F_{i,c}$ for $1 \leq i \leq n$ and $0 \leq c \leq n$ and $F_{i,\ell,c}$ for $1 \leq i \leq n, i + 1 \leq \ell \leq n$ and $0 \leq c \leq n$ of the $n + 1$ processes can be put together to compute the matrix $\mathbf{\Gamma} = ((\gamma_{i,c}))$ in $O(n^3)$ time. So the total time required for the entire procedure consists of the parallel time $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ required by each of the processes plus the final $O(n^3)$ time at the end, for a total of $O(n^3 + n^2\sigma^{(1)} \cdots \sigma^{(k)})$ time. As above, the total space required is $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$.

Remark 7 To the best of our knowledge, there is no previous work in the literature which provides algorithms for computing $\boldsymbol{\gamma}$ and $\mathbf{\Gamma}$ even for the special cases where a game can be represented using either a weighted AND game, or a weighted OR game, or a monotone combination of games. In particular, for the EU voting game, we provide the first algorithms for computing $\boldsymbol{\gamma}$ and $\mathbf{\Gamma}$.

Remark 8 Note that using the table T, it is possible to compute the number of winning coalitions and the number of minimal winning coalitions. From the tables $T^{(i)}, i = 1, \dots, n$, it is possible to compute the vector $\mathbf{m}$, and from T along with $T^{(i)}$, $i = 1, \dots, n$, it is possible to compute $\boldsymbol{\gamma}$. From the tables $C^{(i)}$, it is possible to compute $\mathbf{M}$, and from C along with the tables $C^{(i)}, i = 1, \dots, n$, it is possible to compute $\mathbf{\Gamma}$. So the tables T, $T^{(i)}, i = 1, \dots, n$ and the tables C, $C^{(i)}, i = 1, \dots, n$ provide a wealth of information about the underlying game.

7 Conclusion

In this paper we introduced a new model of simple voting games defined from multiple weight vectors. The previously proposed models of weighted AND games, weighted OR games and monotone Boolean combination of weighted games can be seen as particular cases of the new model that we introduce. We introduced the notion of hyperplane voting games and show that such games can be compactly modelled using the new framework, but not necessarily so using the previously known frameworks. Further, we show that there are games which can be compactly expressed in the new model, but require an exponential number of constituent games in the weighted AND model. For the new model of games that we introduced we described dynamic programming techniques to compute various important parameters of a game from which it is possible to compute the well known voting power indices. The unified approach and the ability to compactly represent new voting scenarios make the new model of games to be of both theoretical and practical interest. We hope that this class of games will attract further attention from researchers.

The focus of study in this paper was simple games, i.e. games where each player votes either yes or no. It is possible that in real-life a player abstains from voting. This leads to the consideration of voting games with abstention. More generally, a voting game may allow the players to have more than two options and also the outcome of the game may also have more than two options. Such general games were introduced by Freixas and Zwicker (2003) and have been later studied by a number of authors (see for example Freixas et al. 2014; Guemmegne and Pongou 2014). It would be interesting to extend the framework that we introduce to voting games with abstention and more generally to voting games with multiple options and multiple levels. We leave this as possible future work.

Appendix A: The cases of weighted AND and weighted OR games

In this section, we show how the results obtained by Wilms (2020) for the cases of weighted AND and weighted OR games can be derived from the general framework that we have introduced.

Let $N_i = \{1, \dots, i\}$ and $\overline{N}_i = N \backslash N_i = \{i+1, \dots, n\}$. Then m_i^+ is given as follows.

$$\begin{aligned} m_i^+ &= \#\{S \subseteq N \setminus \{i\} : f(w_S^{(1)}, \dots, w_S^{(k)}) = 0, f(w_S^{(1)} + w_i^{(1)}, \dots, w_S^{(k)} + w_i^{(k)}) = 1\} \\ &= \#\{(S_1, S_2) : S_1 \subseteq N_{i-1}, S_2 \subseteq \overline{N}_i, \\ &\qquad f(w_{S_1}^{(1)} + w_{S_2}^{(1)}, \dots, w_{S_1}^{(k)} + w_{S_2}^{(k)}) = 0, f(w_{S_1}^{(1)} + w_{S_2}^{(1)} + w_i^{(1)}, \dots, w_{S_1}^{(k)} + w_{S_2}^{(k)} + w_i^{(k)}) = 1\} \\ &= \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)}, \\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} \#Y(i-1, s_1, \dots, s_k) \cdot \#X(i, s_1, \dots, s_k), \end{aligned}$$

where

$$
\begin{aligned}
Y(i-1,s_1,\dots,s_k) =& \{S_1 \subseteq N_{i-1} \,:\, w^{(1)}_{S_1} = s_1,\dots,w^{(k)}_{S_1} = s_k\},\\
X(i,s_1,\dots,s_k) =& \Big\{ S_2 \subseteq \overline{N}_i \,: f(s_1 + w^{(1)}_{S_2},\dots,s_k + w^{(k)}_{S_2}) = 0,\\
& f(s_1 + w^{(1)}_{S_2} + w^{(1)}_i,\dots,s_k + w^{(k)}_{S_2} + w^{(k)}_i) = 1 \Big\}.
\end{aligned}
$$

Note that $\#Y(i-1,s_1,\dots,s_k) = T(i-1,s_1,\dots,s_k)$. Let

$$
\begin{aligned}
A =& \{S_2 \subseteq \overline{N}_i \,: f(s_1 + w^{(1)}_{S_2} + w^{(1)}_i,\dots,s_k + w^{(k)}_{S_2} + w^{(k)}_i) = 1\} \subseteq 2^{\overline{N}_i},\\
B =& \{S_2 \subseteq \overline{N}_i \,: f(s_1 + w^{(1)}_{S_2},\dots,s_k + w^{(k)}_{S_2}) = 1\} \subseteq 2^{\overline{N}_i}.
\end{aligned}
\tag{27}
$$

It follows that $X(i,s_1,\dots,s_k) = A \cap \overline{B}$. We now continue the analysis for monotone games. So f is monotone and we have

$$
m_i = m_i^+ = \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w^{(1)}_i,\\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w^{(k)}_i}} T(i-1,s_1,\dots,s_k)\cdot \#X(i,s_1,\dots,s_k).
$$

Further, since f is monotone, $f(s_1 + w^{(1)}_{S_2},\dots,s_k + w^{(k)}_{S_2}) = 1$ implies $f(s_1 + w^{(1)}_{S_2} + w^{(1)}_i,\dots,s_k + w^{(k)}_{S_2} + w^{(k)}_i) = 1$. Consequently, we have $B \subseteq A$. Then $A = A \cap (B \cup \overline{B}) = (A \cap B) \cup (A \cap \overline{B}) = B \cup (A \cap \overline{B})$, so that $\#X(i,s_1,\dots,s_k) = \#(A \cap \overline{B}) = \#A - \#B = \#\overline{B} - \#\overline{A}$.

Case of weighted AND games. In this case, $f(s_1,\dots,s_k) = 1$ if and only if $s_1 \ge q^{(1)},\dots,s_k \ge q^{(k)}$. Let $H^A(i,s_1,\dots,s_k) = \#\{S_2 \subseteq \overline{N}_{i-1} \,:\, w^{(1)}_{S_2} \ge s_1,\dots,w^{(k)}_{S_2} \ge s_k\}$. Then

$$
\begin{aligned}
\#A =& \#\Big\{ S_2 \subseteq \overline{N}_i \,: f(s_1 + w^{(1)}_{S_2} + w^{(1)}_i,\dots,s_k + w^{(k)}_{S_2} + w^{(k)}_i) = 1 \Big\}\\
=& \#\Big\{ S_2 \subseteq \overline{N}_i \,:\, s_1 + w^{(1)}_{S_2} + w^{(1)}_i \ge q^{(1)},\dots,s_k + w^{(k)}_{S_2} + w^{(k)}_i \ge q^{(k)} \Big\}\\
=& H^A(i+1, q^{(1)} - w^{(1)}_i - s_1,\dots,q^{(k)} - w^{(k)}_i - s_k),
\end{aligned}
\tag{28}
$$

$$
\begin{aligned}
\#B =& \#\Big\{ S_2 \subseteq \overline{N}_i \,: f(s_1 + w^{(1)}_{S_2},\dots,s_k + w^{(k)}_{S_2}) = 1 \Big\}\\
=& \#\Big\{ S_2 \subseteq \overline{N}_i \,:\, s_1 + w^{(1)}_{S_2} \ge q^{(1)},\dots,s_k + w^{(k)}_{S_2} \ge q^{(k)} \Big\}\\
=& H^A(i+1, q^{(1)} - s_1,\dots,q^{(k)} - s_k).
\end{aligned}
\tag{29}
$$

So

$$
\begin{aligned}
m_i = \sum T(i-1,s_1,\dots,s_k)\cdot \Big(& H^A(i+1, q^{(1)} - w^{(1)}_i - s_1,\dots,q^{(k)} - w^{(k)}_i - s_k)\\
& - H^A(i+1, q^{(1)} - s_1,\dots,q^{(k)} - s_k) \Big),
\end{aligned}
$$

where the sum is over $0 \le s_1 \le \sigma^{(1)} - w_i^{(1)}, \dots, 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}$.

Case of weighted OR games. In this case, $f(s_1, \dots, s_k) = 0$ if and only if $s_1 \le q^{(1)}, \dots, s_k \le q^{(k)}$. Let $H^O(i, s_1, \dots, s_k) = \#\{S_2 \subseteq \overline{N}_{i-1} : w_{S_2}^{(1)} \le s_1, \dots, w_{S_2}^{(k)} \le s_k\}$. Then

$$\begin{aligned}\#\overline{A} =&\#\left\{ S_2 \subseteq \overline{N}_i : f(s_1 + w_{S_2}^{(1)} + w_i^{(1)}, \dots, s_k + w_{S_2}^{(k)} + w_i^{(k)}) = 0 \right\} \\ =&\#\left\{ S_2 \subseteq \overline{N}_i : s_1 + w_{S_2}^{(1)} + w_i^{(1)} \le q^{(1)}, \dots, s_k + w_{S_2}^{(k)} + w_i^{(k)} \le q^{(k)} \right\} \\ =&H^O(i+1, q^{(1)} - w_i^{(1)} - s_1, \dots, q^{(k)} - w_i^{(k)} - s_k),\end{aligned} \tag{30}$$

$$\begin{aligned}\#\overline{B} =&\#\left\{ S_2 \subseteq \overline{N}_i : f(s_1 + w_{S_2}^{(1)}, \dots, s_k + w_{S_2}^{(k)}) = 0 \right\} \\ =&\#\left\{ S_2 \subseteq \overline{N}_i : s_1 + w_{S_2}^{(1)} \le q^{(1)}, \dots, s_k + w_{S_2}^{(k)} \le q^{(k)} \right\} \\ =&H^O(i+1, q^{(1)} - s_1, \dots, q^{(k)} - s_k).\end{aligned} \tag{31}$$

So

$$\begin{aligned}m_i = \sum T(i-1, s_1, \dots, s_k) \cdot \big(&H^O(i+1, q^{(1)} - s_1, \dots, q^{(k)} - s_k) \\ &-H^O(i+1, q^{(1)} - w_i^{(1)} - s_1, \dots, q^{(k)} - w_i^{(k)} - s_k)\big),\end{aligned}$$

where the sum is over $0 \le s_1 \le \sigma^{(1)} - w_i^{(1)}, \dots, 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}$.

Note that

$$\begin{aligned}H^A(i, s_1, \dots, s_k) =&H^A(i+1, s_1, \dots, s_k) + H^A(i+1, s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}), \\ H^O(i, s_1, \dots, s_k) =&H^O(i+1, s_1, \dots, s_k) + H^O(i+1, s_1 - w_i^{(1)}, \dots, s_k - w_i^{(k)}),\end{aligned} \tag{32}$$

with appropriate boundary conditions. Using these recurrences, both $H^A(i, \dots)$ and $H^O(i, \dots)$ can be computed in $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ time. Recall that $T()$ can also be computed in $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ time. With the appropriate values of T and H^A available, the value of m_i for the weighted AND game can be computed in $O(\sigma^{(1)} \cdots \sigma^{(k)})$ time. Similarly, with the appropriate values of T and H^O available, the value of m_i for the weighted OR game can be computed in $O(\sigma^{(1)} \cdots \sigma^{(k)})$ time. So for both weighted AND and weighted OR games, the values of m_i, $i = 1, \dots, n$ can be computed in $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ time.

The tables H^A and H^O were introduced by Wilms (2020) who also identified the recurrences (32) and established the results that the swings for all the players for the weighted AND and weighted OR games can be computed in $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ time and $O(\sigma^{(1)} \cdots \sigma^{(k)})$ space.

The specialisation of the general model to the particular case of weighted AND and weighted OR games for computing **m** can be carried out in a similar manner for computing **M**. This results in $O(n^2\sigma^{(1)} \cdots \sigma^{(k)})$ time and $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ space complexity for computing **M**. These results were also established by Wilms (2020).

A.1 Winning coalitions containing a player and the set of blockers

For the number of winning coalitions containing player i, we have

$$\begin{aligned}
\#W_G(i) =& \{S : S \subseteq N \setminus \{i\}, f(w_S^{(1)} + w_i^{(1)}, \dots, w_S^{(k)} + w_i^{(k)}) = 1\} \\
=& \{(S_1, S_2) : S_1 \subseteq N_{i-1}, S_2 \subseteq \overline{N}_i, f(w_{S_1}^{(1)} + f(w_{S_2}^{(1)}) + w_i^{(1)}, \dots, w_{S_1}^{(k)} + w_{S_2}^{(k)} + w_i^{(k)}) = 1\} \\
=& \sum_{\substack{0 \le s_1 \le \sigma^{(1)} - w_i^{(1)}, \\ \dots \\ 0 \le s_k \le \sigma^{(k)} - w_i^{(k)}}} T(i-1, s_1, \dots, s_k) \cdot \#A,
\end{aligned}$$

where A is defined in (27). From (28), we obtain $\#A$ in terms of H^A, while from (30), we obtain $\#A$ in terms of H^O. It follows that for both weighted AND and weighted OR games, $\#W_G(i)$ for any particular i can be obtained in time $O(\sigma^{(1)} \cdots \sigma^{(k)})$ and so $\#W_G(i)$, for $i = 1, \dots, n$ can be obtained in time $O(n\sigma^{(1)} \cdots \sigma^{(k)})$.

Player i is a blocker if and only if $\#W_G(i) = \#W_G$. Since $\#W_G$ can be computed in $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ time, from the above analysis, it follows that the set of blockers for weighted AND and weighted OR games can be computed in $O(n\sigma^{(1)} \cdots \sigma^{(k)})$ time.

The computation of $\#W_G(i)$, $i = 1, \dots, n$, and the determination of the set of blockers were not considered by Wilms (2020).

Acknowledgements We thank the reviewers for their comments which have helped in improving the paper.

Data availability No data has been used for this research.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

Algaba EJM, Bilbao JM, García F, López JJ (2003) Computing power indices in weighted multiple majority games. Math Soc Sci 46:63–80

Algaba E, Bilbao JM, Fernández J (2007) The distribution of power in the European constitution. Eur J Oper Res 176(3):1752–1766

Alonso-Meijide JM, Bilbao J, Casas-Méndez B, Fernández J (2009) Weighted multiple majority games with unions: generating functions and applications to the European union. Eur J Oper Res 198(2):530–544

Aziz H, Paterson MS, Leech D (2007) Combinatorial and computational aspects of multiple weighted voting games. Working Paper, Warwick economic research papers, University of Warwick, Department of Economics. http://wrap.warwick.ac.uk/1393

Banzhaf JF (1965) Weighted voting doesn't work: a mathematical analysis. Rutgers Law Rev 19:317–343

Bhattacherjee S, Sarkar P (2019) Correlation and inequality in weighted majority voting games. In: Dasgupta I, Mitra M (eds) Deprivation, inequality and polarization: essays in honor of S.R. Chakravarty. Springer, New York, pp 161–191

Bolus S (2011) Power indices of simple games and vector-weighted majority games by means of binary decision diagrams. Eur J Oper Res 210(2):258–272

Chakravarty SR, Mitra M, Sarkar P (2015) A course on cooperative game theory. Cambridge University Press, Cambridge

Chen Y, Cheung WS, Ng TW (2019) An upper bound on the dimension of the voting system of the European union council under the Lisbon rules. https://arxiv.org/abs/1907.09711

Coleman JS (1971) Control of collectives and the power of a collectivity to act. In: Lieberman B (ed) Social choice. Gordon and Breach, New York, pp 269–298

Deegan J, Packel EW (1978) A new index of power for simple n-person games. Int J Game Theory 7(2):113–123

Deĭneko VG, Woeginger GJGJ (2006) On the dimension of simple monotonic games. Eur J Oper Res 170:315–318

Einy E, Peleg B (1991) Linear measures of inequality for cooperative games. J Econ Theory 53:328–344

Faliszewski P, Elkind E, Wooldridge M (2009) Boolean combinations of weighted voting games. In: Sichmann S, Castelfranchi (eds) Proceedings of 8th international conference on autonomous agents and multiagent systems (AAMAS), Budapest, Hungary, pp 185–192

Felsenthal DS, Machover M (1998) The measurement of voting power. Edward Elgar, Cheltenham

Freixas J, Marciniak D (2010) On the notion of dimension and codimension. In: Petrosyan LA, Zenkevich NA (eds) Proceedings of the 3rd international conference game theory and management. The International Society of Dynamic Games (Russian Chapter), pp 67–81

Freixas J, Zwicker WS (2003) Weighted voting, abstention, and multiple levels of approval. Soc Choice Welf 21:399–431

Freixas J, Tchantcho B, Tedjeugang N (2014) Voting games with abstention: linking completeness and weightedness. Decis Support Syst 57:172–177

Guemmegne JT, Pongou R (2014) A policy-based rationalization of collective rules: dimensionality, specialized houses, and decentralized authority. J Math Econ 52:182–193

Holler MJ (1982) Forming coalitions and measuring voting power. Polit Stud 30:262–271

Johnston RJ (1978) On the measurement of power: some reactions to Laver. Environ Plan A 10:907–914

Kober S, Weltge S (2021) Improved lower bound on the dimension of the EU council's voting rules. Optim Lett 15:1293–1302

Kurz S, Napel S (2016) Dimension of the Lisbon voting rules in the EU council: a challenge and new world record. Optim Lett 10:1245–1256

Laruelle A, Valenciano F (2004) Inequality in voting power. Soc Choice Welf 22:413–431

Laruelle A, Valenciano F (2008) Voting and collective decision-making: bargaining and power. Cambridge University Press, Cambridge

Leech D (2002) Designing the voting system for the council of the European Union. Public Choice 113:437–464

Maschler M, Owen G (1989) The consistent Shapley value for hyperplane games. Int J Game Theory 18:389–407

Matsui T, Matsui Y (2000) A survey of algorithms for calculating power indices for weighted majority games. J Oper Res Soc Jpn 43:71–86

Olsen M, Kurz S, Molinero X (2016) On the construction of high-dimensional simple games. In: ECAI'16: proceedings of the twenty-second European conference on artificial intelligence. IOS Press, The Hague, The Netherlands, pp 880–885

Penrose LS (1946) The elementary statistics of majority voting. J R Stat Soc 109(1):53–57

Rae D (1969) Decision rules and individual values in constitutional choice. Am Polit Sci Rev 63:40–56

Shapley LS, Shubik MJ (1954) A method for evaluating the distribution of power in a committee system. Am Polit Sci Rev 48:787–792

Taylor AD, Zwicker WS (1993) Weighted voting, multicameral representation, and power. Games Econ Behav 5:170–181

Taylor AD, Zwicker WS (1995) Simple games and magic squares. J Comb Theory Ser A 71:67–88

Taylor AD, Zwicker WS (1999) Simple games: desirability relations, trading, and pseudoweightings. Princeton University Press, Princeton

Uno T (2012) Efficient computation of power indices for weighted majority games. In: Chao K-M, Sheng Hsu T, Lee D-T (eds) Algorithms and computation. Springer, Berlin, pp 679–689

Weber M (2016) Two-tier voting: measuring inequality and specifying the inverse power problem. Math Soc Sci 79:40–45

Wilms I (2020) Dynamic programming algorithms for computing power indices in weighted multi-tier games. Math Soc Sci 108:175–192
Yu C (2022) Hyperplane games, prize games and NTU values. Theory Decis 93:359–370