Computing the Public Good Index for Weighted Voting Games with Precoalitions Using Dynamic Programming

Jochen Staudacher

1 Introduction

The theory of cooperative games with transferable utility Chakravarty et al. (2015) deals with the outcomes and benefits which players can gain by forming coalitions. A large number of different approaches to distribute the payoff of the grand coalition have been studied in the literature, with the Shapley value Shapley (1953) being the most widely known solution concept. This article focusses on the Public Good index (abbreviated PGI, also known as Holler index or as Holler–Packel index) which was formally proposed by Manfred Holler (1982) and axiomatized by Holler and Packel (1983) as a solution concept for the special class of simple games, i.e. for games in which coalitions are either winning with value 1 or losing with value 0. The PGI assumes that only minimal winning coalitions are relevant for measuring the relative power of players. Holler and Li (1995) propose a non-normalized version of the PGI measuring absolute power which they call Public Value and present as a solution concept for general cooperative games with transferable utility.

Ideological proximity or common economic interests are just two of many reasons why certain coalitions are more likely than others. Hence, transferable utility games with a partition of the player set into disjoint precoalitions (also known as a priori unions) have become an important branch of cooperative game theory, with the generalization of the Shapley value by Owen (1977) being the most widely known solution concept. As pointed out in the book by Owen (1995), pp. 303, the players in such a precoalition have agreed to keep together, but, even though they will do so in most cases, they are not forced to comply. Therefore, the influence of a player needs to be evaluated in a two-stage process. In the external stage, the power of the precoalition is determined, and in the internal stage, the results for the members of the precoalition are computed. In the terminology of cooperative game theory, this two-stage process translates into an external game (also known as the quotient game) between the precoalitions and an internal game within each precoalition. Together with various coauthors Manfred Holler proposed and investigated a total of

J. Staudacher (✉)
Fakultät Informatik, Hochschule Kempten, Bahnhofstr. 61, 87435 Kempten, Germany
e-mail: jochen.staudacher@hs-kempten.de

M. A. Leroch and F. Rupp (eds.), *Power and Responsibility*,
https://doi.org/10.1007/978-3-031-23015-8_6

six different variations of the PGI with precoalitions in the articles Alonso-Meijide et al. (2010a, b), Holler and Nohn (2009).

This article discusses the efficient computation of the PGI as well as its six variants with precoalitions for weighted voting games, a very important subclass of simple games. Weighted voting games (also known as weighted majority games or weighted games) are an established model for decision-making and voting in committees, panels or boards. There are n players and each player i is allocated a positive weight w_i, which, in some situations, can be interpreted as the number of votes of a voting bloc. A measure or motion gets passed if and only if a certain quota q, normally more than 50 percent of the sum of all weights, is reached or exceeded. Weighted voting games are relevant well beyond classical voting situations in politics, described in Algaba et al. (2007); Kóczy (2021); Kurz (2016). For example, Holler and Rupp employ the PGI and weighted voting games for analysing social networks in a series of recent papers Holler and Rupp (2019, 2020, 2021) whereas the contemporary paper Staudacher et al. (2021) discusses these tools in the context of indirect control power in corporate shareholding structures. These social and economic network applications may involve large numbers of players as well as precoalitions making the need for fast methods for computing the PGI and its variants with precoalitions (expressed in the final paragraph of the paper by Alonso-Meijide et al. (2010b)) a very relevant subject of research.

Power indices for simple games are frequently computed using generating functions, see, e.g. Algaba et al. (2007); Alonso-Meijide et al. (2008); Bilbao et al. (2000), and the paper by Alonso-Meijide and Bowles Alonso-Meijide and Bowles (2005) for the case of precoalitions. If the subsets of players attain only a small number of different weight sums, this method profits Kurz (2016) and fast-access data structures for polynomials with few coefficients in computer algebra systems like Mathematica Tanenbaum (1997) can be employed. In this paper, we use the strongly related, though mathematically less sophisticated, paradigm of dynamic programming for power index computation Staudacher et al. (2021, 2022), Uno (2012). The recent article Staudacher et al. (2021) proposes a new method for computing the PGI for weighted voting games efficiently. Our goal is to extend the algorithm for the PGI to its six variants with precoalitions.

In Sect. 2, we introduce the basic concepts from cooperative game theory, including simple games, the Public Good index (PGI) and its six variants with precoalitions along the lines of Alonso-Meijide et al. (2009, 2010a, b); Holler and Nohn (2009). Section 3 explains how dynamic programming is used to count coalitions efficiently for weighted voting games and discusses the state-of-the-art algorithms for computing the Public Good index from Staudacher et al. (2021) in detail. Section 4 forms the centrepiece of this paper and presents the new algorithms for the six Public Good indices with precoalitions. We point out how our algorithms reflect both the definitions of the indices as well as the different internal division procedures and present a sophisticated new approach for computing the Owen Extended Public Good Index. Section 5 discusses implementations of our new algorithms in C++ including numerical experiments and reports supportive computing times. We end with some concluding remarks and an outlook to open problems in Sect. 6.

2 Preliminaries

In this section, we briefly review some terminology for cooperative games and precoalitions along the lines of the paper by Alonso-Meijide et al. (2009) and define the Public Good index (PGI) and its variants with precoalitions following the articles Alonso-Meijide et al. (2010a, b), Holler and Nohn (2009).

2.1 Cooperative Games, Simple Games and Precoalitions

Let $N = \{1, ..., n\}$ denote a finite set of n players. A group of players $S \subseteq N$ is called a coalition, whereas 2^N symbolizes the set of all subsets of N. $\emptyset$ stands for the *empty coalition* and N is referred to as the *grand coalition*. By $|S|$ we denote the cardinality of a coalition S, i.e. the number of its members, hence $|N| = n$. An n-person cooperative game with transferable utility can be characterized as a pair (N, v) where $v : 2^N \to \mathbb{R}$ is referred to as the *characteristic function* assigning a real value to all coalitions $S \in 2^N$, with $v(\emptyset) = 0$, i.e. in a cooperative game the value of the empty coalition is always zero, see, e.g. one of the books Chakravarty et al. (2015), p. 20, or Owen (1995), p. 213. A cooperative game is *monotone* if for all coalitions $S, T \in 2^N$ the relation $S \subseteq T$ implies $v(S) \leq v(T)$.

We call a cooperative game *simple* if it is monotone and there holds $v(N) = 1$ and $v(S) = 0$ or $v(S) = 1$ for each coalition $S \subset N$. Coalitions for which $v(S) = 1$ are called *winning coalitions* in simple games, whereas coalitions for which $v(S) = 0$ are termed *losing coalitions*. A player i is a *critical player* (also known as a decisive player or swing player) in a winning coalition S if $v(S \backslash \{i\}) = 0$, i.e. the winning coalition S becomes a losing one if player i leaves S. We call a winning coalition S *minimal winning* if it contains only critical players, i.e. if every proper subset of S is a losing coalition.

Weighted voting games (also known as weighted majority games or weighted games) are probably the most important subclass of simple games. They are employed as models in a large number of practical applications Algaba et al. (2007); Holler and Rupp (2019, 2020, 2021); Kóczy (2021); Kurz (2016); Staudacher et al. (2021). An n-player weighted voting game is specified by n non-negative real weights w_i, $i = 1, \ldots, n$, and a non-negative real quota q, normally $q > \frac{1}{2}\sum_{i=1}^{n} w_i$. Its characteristic function $v : 2^N \to \{0, 1\}$ takes the value $v(S) = 1$ for every winning coalition S, i.e. $w(S) = \sum_{i \in S} w_i \geq q$, and $v(S) = 0$ otherwise, implying that coalition S is losing.

Let us create an external division of our set of players $N = \{1, ..., n\}$ into precoalitions (also known as a priori unions). Let $\mathcal{P}(N)$ denote for the set of all partitions of N, with a partition being a set of non-empty subsets of N satisfying the constraint that N is a disjoint union of these subsets. We call an element $P \in \mathcal{P}(N)$ a *coalition structure* (also known as a system of unions) of the set N. A simple game with a coalition structure can be written as a triplet (N, v, P). Following Alonso-Meijide et al. (2009), we write our coalition structure in the form $P = \{P_1, \ldots, P_l\}$, i.e. we have l precoalitions $P_1, \ldots, P_l$ and the set $L = \{1, \ldots, l\}$ serves as the index set of the partition P. For a weighted voting game with a coalition structure P, the *external game* (also known as the quotient game) is defined as the weighted voting game v^P played between the l precoalitions Alonso-Meijide et al. (2009); Malawski (2004). The external game is formally defined as the weighted voting game $[q; w(P_1), \ldots, w(P_l)]$, i.e. it is characterized by the (unmodified) quota q and the weights $w(P_1), \ldots, w(P_l)$ of the l precoalitions, where $w(P_k) = \sum_{i \in P_k} w_i$ with $k \in L$, $L = \{1, \ldots, l\}$.

2.2 The Public Good Index and Its Variants with Precoalitions

A function f that receives an n-person simple game (N, v) specified by its player set N and its characteristic function v as its input, and passes a unique vector $f(N, v) = (f_1(N, v), \ldots, f_n(N, v))$ as its output is called a *power index*. The literature offers an array of power indices, including the Shapley–Shubik index Shapley and Shubik (1954), the Banzhaf index Banzhaf III (1964), the Johnston index Johnston (1978) and the Deegan–Packel index Deegan and Packel (1978). In this paper, we focus entirely on the Public Good Index (PGI) formally defined by Manfred Holler (1982) and refer the reader to the overview article by Bertini et al. (2013) for a deeper discussion of power indices.

Given a simple game (N, v) with n players, let M denote the set of its minimal winning coalitions and M_i the set of minimal winning coalitions containing player $i \in \{1, \ldots, n\}$. The *Public Good index* (PGI) δ_i of player i is given as

$$\delta_i(N, v) = \frac{|M_i|}{\sum_{j=1}^{n} |M_j|}. \tag{1}$$

A *coalitional power index* g is a function which gets an n-person simple game with a coalition structure (N, v, P) specified by its player set N, its characteristic function v and a partition P as its input and delivers a unique vector $g(N, v, P) = (g_1(N, v, P), \ldots, g_n(N, v, P))$ as its output.

Taking up the idea of the external game v^P played between the precoalitions introduced at the end of Sect. 2.1, we can measure the power of a union $Q \in P = \{P_1, \ldots, P_l\}$ in terms of the PGI. We denote the set of minimal winning coalitions in the external game by M^P and by M_Q^P the set of those minimal winning coalitions in the external game containing precoalition $Q \in P$. Following Alonso-Meijide et al. (2010a), we can write

$$\delta_Q(P, v^P) = \frac{|M_Q^P|}{\sum_k |M_{P_k}^P|}. \tag{2}$$

With the definitions of the PGI on the levels of individual players (1) and precoalitions (2) in place, we can define the six variants of the PGI with precoalitions.

The *Solidarity PGI* Alonso-Meijide et al. (2010b) assigns power to each precoalition according to its PGI in the external game (2) in the first step. In the second step, the Solidarity PGI stresses the public good property and attributes equal power to each member of the same precoalition. We can formally define the Solidarity PGI Υ_i of player $i \in P(i)$, i.e. player i contained in union $P(i)$, as follows:

$$\Upsilon_i(N, v, P) = \delta_{P(i)}(P, v^P)\frac{1}{|P(i)|}. \tag{3}$$

The *Union PGI* Holler and Nohn (2009) reflects the spirit of the original PGI by assuming that the coalitional value is a public good and only minimal winning coalitions (with respect to the coalition structure P) are relevant. The power of an individual player $i \in P(i)$, i.e. player i contained in union $P(i)$, is thus proportional to the number of minimal winning coalitions her precoalition belongs to in the external game. We can formally define the Union PGI Λ_i of player $i \in P(i)$, as follows:

$$\Lambda_i(N, v, P) = \frac{|M^P_{P(i)}|}{\sum_k |P_k||M^P_{P_k}|}. \tag{4}$$

As for the Solidarity PGI, players within the same precoalition obtain the identical Union PGI. We note that among the six extensions of the PGI for precoalitions, the Union PGI is the only solution concept that does not attribute power to precoalitions based on the PGI in the external game (2).

Holler and Nohn (2009) propose three different approaches for reflecting an individual player's threat power to leave the union. In all three cases, the total power attributed to a precoalition Q is given by its external PGI (2). That power is then distributed internally among individual members of unions in threat games.

For the *Threat PGI 1* (TPGI 1) Holler and Nohn (2009), only a minimal degree of stability of the precoalition structure P is assumed. As soon as a single player i leaves her coalition $P(i)$, then not only that union $P(i)$, but the complete precoalition structure P breaks apart. In terms of intra-union allocation of power, this model implies that subsets of a union are not only able to cooperate with other precoalitions, but also with subsets of these precoalitions. Hence, we define the Threat PGI 1 T^1_i of player $i \in P(i)$ as follows:

$$T^1_i(N, v, P) = \delta_{P(i)}(P, v^P)\frac{\delta_i(N, v)}{\sum_{j\in P(i)} \delta_j(N, v)}, \tag{5}$$

whenever $\sum_{j\in P(i)} \delta_j(N, v) > 0$ and $T^1_i(N, v, P) = 0$ otherwise.

The *Threat PGI 2* (TPGI 2) Holler and Nohn (2009) assumes a greater degree of stability of the precoalition structure P. As soon as a single player i leaves her coalition $P(i)$, then only that union $P(i)$ breaks apart into singletons, but the rest of the precoalition structure remains intact. In terms of intra-union allocation of power, this model implies that subsets of a union are only allowed to cooperate with other precoalitions, but not with subsets of these precoalitions. Following Holler and Nohn (2009), for union $Q \in P$, let $P/Q = P \backslash \{Q\} \cup \{\{i\}| i \in Q\}$ stand for the precoalition structure after Q breaks up into singletons $\{i\}$, $i \in Q$. We define the Threat PGI 2 T_i^2 of player $i \in P(i)$ as follows:

$$T_i^2(N, v, P) = \delta_{P(i)}(P, v^P) \frac{\delta_i(P/P(i), v^{P/P(i)})}{\sum_{j \in P(i)} \delta_j(P/P(i), v^{P/P(i)})}, \tag{6}$$

whenever $\sum_{j \in P(i)} \delta_j(P/P(i), v^{P/P(i)}) > 0$ and $T_i^2(N, v, P) = 0$ otherwise.

The *Threat PGI 3* (TPGI 3) Holler and Nohn (2009) assumes a maximal degree of stability of the precoalition structure P. In case a single player i leaves her coalition $P(i)$, then the rest of that union $P(i)$ remains intact as do all the other precoalitions. Following Holler and Nohn (2009), let $P/i = P \backslash \{P(i)\} \cup \{\{i\}, P(i) \backslash \{\{i\}\}$ stand for the precoalition structure after player i breaks away from her union $P(i)$ and plays on her own. We define the Threat PGI 3 T_i^3 of player $i \in P(i)$ as follows:

$$T_i^3(N, v, P) = \delta_{P(i)}(P, v^P) \frac{\delta_i(P/i, v^{P/i})}{\sum_{j \in P(i)} \delta_j(P/j, v^{P/j})}, \tag{7}$$

whenever $\sum_{j \in P(i)} \delta_j(P/j, v^{P/j}) > 0$ and $T_i^3(N, v, P) = \frac{\delta_{P(i)}(P, v^P)}{|P(i)|}$ otherwise.

The *Owen Extended PGI* Alonso-Meijide et al. (2010a, b), Holler and Nohn (2009), distributes power within precoalitions according to the possibilities which the subsets of this precoalitions possess to form winning coalitions with other precoalitions. We call a subset $S \subseteq Q$ of a precoalition Q an *essential part* with respect to a minimal winning coalition $R \in M_Q^P$ (on the external level) if $S \cup \bigcup_{Q' \in R \backslash \{Q\}} Q'$ is a winning coalition in (N, v) and $T \cup \bigcup_{Q' \in R \backslash \{Q\}} Q'$ is a losing coalition for all true subsets $T \subset S$. We denote the *set of essential parts* with respect to a minimal winning coalition $R \in M^P$ (on the external level) containing player i by $E_i^R(N, v, P)$. The Owen Extended PGI Γ_i of player $i \in P(i)$ is defined as follows:

$$\Gamma_i(N, v, P) = \delta_{P(i)}(P, v^P) \sum_{R \in M_{P(i)}^P} \frac{1}{|M_{P(i)}^P|} \frac{|E_i^R(N, v, P)|}{\sum_{j \in P(i)} |E_j^R(N, v, P)|}. \tag{8}$$

We note that the definition (8) coincides with the "counting PGI" from the work by Malawski (2004) and stress that the Owen Extended PGI manages to be as close as possible in spirit to the extension of the Shapley value to games with a coalition structure proposed by Owen (1977). For the coalition structures $P^n = \{\{1\}, \{2\}, \ldots, \{n\}\}$ and $P^N = \{N\}$, the Owen Extended PGI reduces to the PGI in both cases (just as the Owen value Owen (1977) reduces to the Shapley value in both cases).

3 Dynamic Programming for Computing the Public Good Index

In this section, we introduce the technique of dynamic programming for counting coalitions in weighted voting games efficiently and present the state-of-the-art algorithms for computing the PGI for weighted voting games from the paper Staudacher et al. (2021). Given that Staudacher et al. (2021) is partly a survey paper, the discussion of the PGI in Staudacher et al. (2021) is rather brief and thus the new algorithm merits a more detailed presentation in this section of our article. In other words, we are not computing any PGIs with precoalitions in this section, but prepare the groundwork for doing so in Sect. 4.

3.1 Counting Winning and Losing Coalitions via Dynamic Programming

Every weighted voting game allows for an integer representation Kurz (2016). Therefore, we assume that the weights w_i of the n players in our weighted voting game as well as the quota q are positive integers for the rest of the article. We set $\tilde{w} = w(N) = \sum_{i=1}^{n} w_i$ and assume $q \le \tilde{w}$. We stress that the algorithms presented in this and the next section are valid for any integer quota with $1 \le q \le \tilde{w}$.

Dynamic programming is an algorithmic paradigm based on two pillars. We aim to solve a problem algorithmically by dividing it into subproblems and storing intermediate results efficiently. We employ this paradigm to find out how many subsets $S \subseteq \{1, 2, \dots, n\}$ there are with weight x, i.e. $w(S) = \sum_{i \in S} w_i = x$, for $x \in \{0, 1, \dots, \tilde{w}\}$.

Theorem 1 *(see Chakravarty et al. (2015), p. 229). Let $T(i, x)$ be the number of possibilities to write the integer x as a sum of the first i weights $w_1, \dots, w_i$. For all $i \in \{0, \dots, n\}$ and all $x \in \{0, 1, \dots, \tilde{w}\}$, the following recursion delivers $T(i, x)$:*

$$\begin{aligned} T(i,0) &= 1 && \textit{for}\quad 0 \le i \le n \\ T(0,x) &= 0 && \textit{for}\quad x > 0 \\ T(i,x) &= T(i-1,x) + T(i-1,x-w_i) && \textit{otherwise.} \end{aligned}$$

Note that the above equations can be interpreted as a boundary condition stating that we can obtain the sum 0 in exactly one way, i.e. via the empty set, as another boundary condition stating that we cannot obtain any sum $x > 0$ without any term and an actual recursion mirroring that the first i weights can deliver a sum $x > 0$ either with or without player i. In practice, T is normally not stored as a two-dimensional table, but efficiently as a vector updated from $T(i-1, x)$ to $T(i, x)$. In our discussion of memory space requirements, we follow the convention by Uno (2012) throughout our paper and omit the need to store the n weights and the corresponding n values of the power indices. This facilitates a clearer and more concise presentation. According to this convention, Theorem 1 enables us to compute the vector $T(n, x)$ for $x \in \{0, 1, \dots, \tilde{w}\}$ in $O(n\tilde{w})$ time and $O(\tilde{w})$ memory space. We finally note that it is as simple to update from $T(i-1, x)$ to $T(i, x)$ in Theorem 1 as it is to "downdate" the vector $T(i+1, x)$ to $T(i, x)$ via

$$T(i, x) = T(i+1, x) - T(i, x - w_i). \tag{9}$$

3.2 *Computing the Public Good Index via Dynamic Programming*

The recent article Staudacher et al. (2021) proposes a new algorithm with a favourable pseudopolynomial complexity for computing the PGI of weighted voting games. We present this algorithm in more detail. In this subsection, we assume the positive integer weights of our n players to be in a descending order, i.e. $w_1 \geq \cdots \geq w_n$.

As pointed out in the textbook Chakravarty et al. (2015), p. 235, it is relatively simple to find the total number $|M|$ of minimal winning coalitions in a weighted voting game via dynamic programming in $O(qn)$ time. We observe

$$|M| = \sum_{i=1}^{n} \sum_{x=q-w_i}^{q-1} T(i-1, x)$$

and note that $\sum_{x=q-w_i}^{q-1} T(i-1, x)$ counts the number of minimal winning coalitions with player i being the player with largest index in the minimal winning coalition.

The recent paper Staudacher et al. (2021) shows that not only $|M|$, but also the cardinalities $|M_i|$ of the sets of minimal winning coalitions containing player i can be found in $O(qn)$ time and $O(q)$ memory space for all players. Following Staudacher et al. (2021), we define the operator $d(S)$ removing the player with largest index from a coalition S. Further, let $w(S) = \sum_{i \in S} w_i$ stand for the weight of coalition S and

$$B(i, x) = |\{S \in 2^N | S \subseteq \{i, \dots, n\}, i \in S, w(d(S)) < x \leq w(S)\}| \tag{10}$$

for all players $i \in \{1, \dots, n\}$ and all weights $1 \leq x \leq q$. $B(i, x)$ is the number of coalitions S with i as the player with smallest index such that any coalition S has weight greater or equal x whereas S without its player with largest index has weight less than x. We observe $|M_1| = B(1, q)$. Apart from that observation, Eq. (10) may seem awkward at first, but it helps count $|M_i|$ efficiently without any need to know the player with largest index in any minimal winning coalition contained in M_i. We can obtain B by looping for i from n to 1, as follows.

Theorem 2 *(see Staudacher et al. (2021)) For all $1 \leq i \leq n$ and all weights $1 \leq x \leq q$ there holds*

$$B(i,x) = \begin{cases} 1 & \text{for } 1 \leq x \leq w_i \\ B(i+1, x-w_i) + B(i+1, x-w_i+w_{i+1}) & \text{for } x > w_i, i < n \\ 0 & \text{otherwise.} \end{cases}$$

Proof By definition of $B(i,x)$ from (10), there holds $B(i,x) = 1$ for $1 \leq x \leq w_i$ as we count the singleton coalition consisting of player i. Furthermore, the statement holds true for $i = n$. In all other cases, i.e. for $x > w_i$ and $i < n$, the recursion means that either player $i+1$ is part of a coalition counted in $B(i,x)$ (first term) or player $i+1$ is not part of a coalition counted in $B(i,x)$ (second term).

$$\begin{aligned} B(i,x) = &|\{S \in 2^N | S \subseteq \{i+1,\ldots,n\}, (i+1) \in S, w(d(S)) < x - w_i \leq w(S)\}| \\ &+|\{S \in 2^N | S \subseteq \{i+1,\ldots,n\}, (i+1) \notin S, w(d(S)) < x - w_i \leq w(S)\}| \\ &=|\{S \in 2^N | S \subseteq \{i+1,\ldots,n\}, (i+1) \in S, w(d(S)) < x - w_i \leq w(S)\}| \\ &+|\{S \in 2^N | S \subseteq \{i+1,\ldots,n\}, (i+1) \in S, w(d(S)) < x - w_i + w_{i+1} \leq w(S)\}| \\ &= B(i+1, x-w_i) + B(i+1, x-w_i+w_{i+1}). \end{aligned}$$

Theorem 3 *(see Staudacher et al. (2021)) Let v be an n-player weighted voting game with positive integer weights sorted in a descending order. With the help of the quantities $T(i,x)$ from Theorem 1 and $B(i,x)$ from Theorem 2, the Public Good index can be computed in $O(qn)$ time and $O(q)$ memory space for all players as there holds*

$$|M_i| = \sum_{x=0}^{q-1} T(i-1,x) \cdot B(i, q-x).$$

Proof The statement is true for $i = 1$ as $|M_1| = T(0,0) \cdot B(1,q) = 1 \cdot B(1,q) = B(1,q)$. For $i \geq 2$ we find

$$\begin{aligned} |M_i| = &|\{S \in 2^N | i \in S, w(d(S)) < q \leq w(S)\}| \\ &=|\{S \in 2^N | S = S_1 \cup S_2, S_1 \subseteq \{1,\ldots,i-1\}, S_2 \subseteq \{i,\ldots,n\}, i \in S_2, \\ &\quad w(d(S_2)) < q - w(S_1) \leq w(S_2)\}| \\ &=\sum_{x=0}^{q-1} |\{S_1 \subseteq \{1,\ldots,i-1\} | w(S_1) = x\}| \cdot \\ &\qquad |\{S_2 \subseteq \{i,\ldots,n\} | w(d(S_2)) < q - x \leq w(S_2)\}| \\ &=\sum_{x=0}^{q-1} T(i-1,x) \cdot B(i, q-x). \end{aligned}$$

Since Staudacher et al. (2021) does not list any algorithms, we conclude this section with Algorithm 1 for computing the PGI for a weighted voting game specified by its number of players n, its quota q and its vector w of n weights in $O(qn)$ time and $O(q)$ space.

Algorithm 1 Computing the PGI for weighted voting games

```
1: procedure PGI(n, q, w)
2:     Compute vector $T(x) = T(n-1, x)$ for $x \in [0, q-1]$ according to Theorem 1
3:     Prepare vector $B(x) = B(n, x)$ for $x \in [1, q]$ according to Theorem 2
4:     for $i$ from $n$ to 1 do
5:         $|M_i| = \sum_{x=0}^{q-1} T(x) \cdot B(q-x)$.
6:         if $i > 1$ then
7:             Update vector $B(x) = B(i-1, x)$ according to Theorem 2
8:             Downdate vector $T(x) = T(i-2, x)$ according to Equation (9)
9:         end if
10:    end for
11:    for $i$ from 1 to $n$ do
12:        Compute $\delta_i = \frac{|M_i|}{\sum_{j=1}^{n} |M_j|}$
13:    end for
14:    Return vector $\delta$
15: end procedure
```

4 Computing Public Good Indices with Precoalitions via Dynamic Programming

This section forms the centrepiece of the article. It discusses new dynamic programming algorithms for the six Public Good indices introduced in Sect. 2.2. Thereby, we complement the recent work Staudacher et al. (2022). While Staudacher et al. (2022) generalizes the state-of-the-art algorithms for computing the Banzhaf index Banzhaf III (1964) and the Shapley–Shubik index Shapley (1953); Shapley and Shubik (1954) from the papers by Uno (2012) and Kurz (2016) to the Banzhaf-Owen (1981), Owen (1977) and Symmetric Coalitional Banzhaf Alonso-Meijide and Fiestras-Janeiro (2002) indices via two-level procedures, we hereby extend our algorithm for the PGI from the previous section to its six variants with precoalitions.

As we stated in Sect. 2.1, we assume that there are l precoalitions $P_1, \ldots, P_l$. The *external game* (also known as the quotient game) is defined as the weighted voting game played between the precoalitions Alonso-Meijide et al. (2009); Malawski (2004), i.e. a weighted voting game represented by the (unmodified) quota q and the m weights $w(P_1), \ldots, w(P_l)$ where $w(P_k) = \sum_{i \in P_k} w_i$ with $k \in L$, $L = \{1, \ldots, l\}$. Furthermore, we define p as the maximal size of a precoalition, i.e. $p = \max_{k \in L} |P_k|$, and r as the maximal weight of a precoalition.

In the following, we point out how the ideas and algorithms from the previous section translate into external and internal weighted voting games and algorithms for the six Public Good indices with precoalitions from Sect. 2.2 possessing fairly attractive pseudopolynomial computing times and storage requirements.

Theorem 4 *For a weighted voting game with positive integer weights and a precoalition structure $P = \{P_1, \dots, P_l\}$, both the Solidarity Public Good and Union Public Good indices can be computed in $O(lq)$ time and $O(q)$ space for all players.*

Proof The proof is trivial. As we can see from the definitions of the Solidarity PGI (3) and the Union PGI (4), the only computational challenge is to find $|M_Q^P|$ for all precoalitions $Q \in P$ on the external level using Algorithm 1. For l unions, this can be achieved in $O(lq)$ computing time and $O(q)$ space. $\square$

Next, we devote one theorem to the three Threat PGIs from Sect. 2.2 each.

Theorem 5 *For a weighted voting game with positive integer weights and a precoalition structure $P = \{P_1, \dots, P_l\}$ the Threat Public Good indices T^1 can be computed in $O((l + n)q)$ time and $O(q)$ space for all players.*

Proof The definition of the TGPI 1 indices (5) reveals the claim. We need two computations of PGIs using Algorithm 1, one on the level of the l precoalitions, another on the level of the n individual players. This can be achieved in $O((l + n)q)$ computing time and $O(q)$ space. $\square$

For the TPGI 2 indices (6), more PGI computations are needed.

Theorem 6 *For a weighted voting game with positive integer weights and a precoalition structure $P = \{P_1, \dots, P_l\}$, the Threat Public Good indices T^2 can be computed in $O(l(l + p)q)$ time and $O(q)$ space for all players.*

Proof The definition of the TGPI 2 indices (6) reveals that we need $l + 1$ computations of PGIs using Algorithm 1, one on the level of the l precoalitions and another l simulating the break-up of each precoalition into singletons. The number of entities in a PGI computation is bounded by $l + p - 1$ with p being the maximal size of a precoalition. According to the conventions of the O-notation, we may state that the computation can be performed in $O(l(l + p)q)$ time and $O(q)$ memory space. $\square$

Computing the TPGI 3 indices (7) means more effort as players break away as singletons leading to one internal game per player.

Theorem 7 *For a weighted voting game with positive integer weights and a precoalition structure $P = \{P_1, \dots, P_l\}$, the Threat Public Good indices T^3 can be computed in $O(nlq)$ time and $O(q)$ space for all players.*

Proof The definition of the TGPI 3 indices (7) reveals that we need $n + 1$ computations of PGIs using Algorithm 1, one on the level of the l precoalitions and another n internal PGIs when each player breaks away individually. The number of entities in a PGI computation in any internal threat game is $l + 1$. According to the conventions of the O-notation, we may state that the computation can be performed in $O(nlq)$ time and $O(q)$ memory space. $\square$

Computing the Owen Extended PGI for weighted voting games is more complicated than for the other five PGI variants with precoalitions. From its definition (8), we know that in the internal games we need to work out how many times player $i \in P(i)$ is part of a minimal winning coalition S formed together with other players from $P(i)$ and other precoalitions from the set $P \backslash P(i)$. We formulate a theorem with worst-case estimates for computing the Owen Extended PGI. Its proof describes our algorithm.

Theorem 8 *For a weighted voting game with positive integer weights and a precoalition structure $P = \{P_1, \ldots, P_l\}$, the Owen Extended Public Good indices Γ can be computed in $O(l(l+p)q + lpr^2)$ time and $O(q + pr)$ space for all players.*

Proof We assume the weights of the l precoalitions to be in a descending order, i.e. $w(P_1) \geq \cdots \geq w(P_l)$. The case $r = w(P_1) \geq q$ is fairly simple. There are $\tilde{l} \leq l$ precoalitions with $w(P_k) \geq q$ for all $k = 1, \ldots, \tilde{l}$. We obtain the Owen Extended Public Good indices Γ_i for all players i contained in precoalition P_j for $j = 1, \ldots, l$ by first computing the PGIs for the internal games with the weights $w^{int} = (w_1, \ldots, w_{|P_j|})$ and quota q and then dividing the results by $\tilde{l}$.
For $r = w(P_1) < q$, it is more challenging to compute Γ_i for all players i contained in precoalition P_j for $j = 1, \ldots, l$. Equation (8) tells us to find the relative frequencies with which player $i \in P_j$ is contained in an essential part of a minimal winning coalition $R \in M^P_{P_j}$ on external level. We initialize a vector f with $f(i) = 0$ for $i = 1, \ldots, |P_j|$ for summing these relative frequencies. By $w^{int} = (w_1, \ldots, w_{|P_j|})$ we denote the vector of the individual weights of the members of union P_j and by $w(P_j) = \sum_{i=1}^{|P_j|} w_i^{int}$ its sum. Since $w(\bigcup_{Q' \in R \backslash \{P_j\}} Q') < q$, we start with a preprocessing step and use Algorithm 1 to compute

$$h(i, x) = PGI(|P_j|, x, w^{int}) \tag{11}$$

for all $1 \leq i \leq |P_j|$ and all $1 \leq x \leq w(P_j)$. We first look at minimal winning coalitions $R \in M^P_{P_j}$ on external level such that unions $P_{j+1}, \ldots, P_l$ are not contained in R. Let $T^{ext}(j-1, x)$ for $0 \leq x \leq q-1$ be the vector obtained from Theorem 1 for the weights of the first $j-1$ unions, i.e. from $w(P_1), \ldots, w(P_{j-1})$. We update

$$f(i) = f(i) + \sum_{x=q-w(P_j)}^{q-1} T^{ext}(i-1, x) \cdot h(i, q-x) \tag{12}$$

for all $1 \leq i \leq |P_j|$. Next, we loop for k from $j+1$ to $l-1$ and deal with minimal winning coalitions $R \in M^P_{P_j}$ on external level such that unions $P_{k+1}, \ldots, P_l$ are not contained in R. We update the vector $T^{ext}(j-1, x)$ for $0 \leq x \leq q-1$ successively while omitting the weight $w(P_j)$ of precoalition j, i.e. we compute $T^{ext}_{-j}(k-1, x)$ for $0 \leq x \leq q-1$ using Theorem 1 with the weights $w(P_1), \ldots, w(P_{j-1}), w(P_{j+1}), \ldots, w(P_{k-1})$. We update

$$f(i) = f(i) + \sum_{x=q-w(P_j)-w(P_k)}^{q-w(P_j)-1} T^{ext}_{-j}(k-1, x) \cdot h(i, q - w(P_k) - x) \tag{13}$$

for all $1 \leq i \leq |P_j|$. Finally, with $\delta_{P_j}(P, v^P)$ from (2) and $|M^P_{P_j}|$ we find

$$\Gamma_i(N, v, P) = \delta_{P_j}(P, v^P) \frac{f(i)}{|M^P_{P_j}|}.$$

Computing the values $h(i, x)$ in (11) for all $1 \leq i \leq |P_j|$ and all $1 \leq x \leq w(P_j)$ can be done in at most $O(pr^2)$ time and $O(pr)$ memory space as there are at most $p \geq |P_j|$ members of a precoalition and as the weight of a precoalition is at most $r \geq w(P_j)$. For all l precoalitions, the preprocessing step (11) costs at most $O(lpr^2)$ time and $O(pr)$ memory space. The vector T^{ext} needs additional $O(q)$ memory space justifying the estimate $O(q + pr)$ for storage. As for computing time, we also need to consider the outer loop for $j = 1, \dots, l$ over all precoalitions as well as the inner loops for $k = j + 1, \dots, l$ for updating the vector f in (12) and (13). This can be done in $O(l(l + p)q)$ time justifying our total estimate $O(l(l + p)q + lpr^2)$. □

There is one major difference between Algorithm 1 for computing the PGI and our new algorithm for the Owen Extended PGI from Theorem 8. In Algorithm 1 for the PGI, we do not need to know the concrete indices of the players with largest index in a minimal winning coalition (assuming that weights are in a descending order). However, our algorithm for the Owen Extended PGI from Theorem 8 not only assumes that precoalitions are in a descending order by their weights, but it also needs the information on the precoalition with largest index in a minimal winning coalition on the external level. The latter requires an additional loop over precoalitions.

5 Numerical Results and Software

In the recent articles Staudacher et al. (2021, 2022), a powerful software package named EPIC (Efficient Power Index Computation) providing efficient C++ implementations of various power indices (both with and without precoalitions) for weighted voting games was introduced. EPIC is freely available via https://github.com/jhstaudacher/EPIC/. We integrated our new algorithms for the six PGI variants with precoalitions in EPIC. For further details on the internal workings of EPIC, readers are referred to Staudacher et al. (2021, 2022).

For testing our new algorithms, we created a number of games, publicly available at https://github.com/jhstaudacher/EPIC/tree/master/test_cases/precoalitions.

Tables 1 and 2 list computing times of the PGI without precoalitions (PGI), the Solidarity PGI (SPGI), the Union PGI (UPGI), the three Threat PGIs (TPGI1, TPGI2, TPGI3), and the Owen Extended PGI (OPGI) for some of these example problems. The numerical results in Tables 1 and 2 were obtained under Ubuntu 20.04 focal (64

Table 1 Computing times for three test problems with quotas equal to 50 % plus 1 vote

	Problem 1: 741 players, $\tilde{w} = 37064$, 40 precoalitions, max. coal. size 40, avg. coal. size 18, max. prec. weight 2049	Problem 2: 3034 players, $\tilde{w} = 152098$, 60 precoalitions, max. coal. size 78, avg. coal. size 50, max. prec. weight 4150	Problem 3: 3434 players, $\tilde{w} = 72068$, 200 precoalitions, max. coal. size 54, avg. coal. size 17, max. prec. weight 1488
Index	Time (sec)	Time (sec)	Time (sec)
PGI	0.973	95.148	56.413
SPGI	0.42	0.252	0.394
UPGI	0.42	0.252	0.396
TPGI1	1.022	96.985	57.891
TPGI2	2.056	25.887	80.330
TPGI3	26.987	747.713	1274.111
OPGI	38.967	624.871	65.645

bit) on an Intel Core(tm) i7-6600U CPU with a clock speed of 2.60 GHz and 16 GB RAM, i.e. on a standard laptop PC.

In Table 1, we compare three different problems with quotas equal to 50 % plus 1 vote in each case. As reported in Staudacher et al. (2021), the new algorithm for the PGI is very fast, handling problems with more than 3000 players and large quotas in less than two minutes. Not surprisingly, the computing times of the SPGI and the UPGI coincide and are very small since sophisticated computations are performed only on precoalition level. Computing times for TPGI1 are only a little larger than for the PGI confirming our claims from Theorem 5. Comparing computing times for TPGI2, more than three times as many precoalitions (and more players) in problem 3 outweigh the fact that the quota in problem 2 is more than twice the quota in problem 3. As predicted by Theorem 7, the computing times for TPGI3 are much larger than for TPGI2 mirroring the internal games for each individual player. Our new algorithm for OPGI proves to be applicable for large problems and outperforms TPGI3 for test problems 2 and 3. Comparing the computing times of OPGI and TPGI2 for problems 1, 2 and 3 underlines the influence of the preprocessing steps (11) and the maximal weight of a precoalition on our algorithm from Theorem 8.

In Table 2, we study the effects of the quota for another test problem. For PGI, SPGI, UPGI, TPGI1, TPGI2 and TPGI3, these effects on computing times are as predicted by Theorems 3, 4, 5, 6 and 7. For OPGI, the quota has hardly any impact on computing times underlining the fact that the preprocessing steps in (11) are by far the most time-consuming part of the algorithm.

Table 2 Computing times for a test problems with 1031 players, $\tilde{w} = 22031$, 60 precoalitions, max. coal. size 52, avg. coal. size 18 and max. prec. weight 1493 for three different quotas

	$q = 5508(\approx 25\%)$	$q = 11016(\approx 50\%)$	$q = 16524(\approx 75\%)$
Index	Time (sec)	Time (sec)	Time (sec)
PGI	0.447	0.875	1.286
SPGI	0.021	0.037	0.052
UPGI	0.021	0.037	0.052
TPGI1	0.473	0.945	1.330
TPGI2	1.214	2.401	3.536
TPGI3	16.304	32.340	47.582
OPGI	14.482	14.536	14.563

6 Outlook and Conclusions

This article proposes new dynamic programming algorithms for six variants of the PGI with precoalitions Alonso-Meijide et al. (2010a, b); Holler and Nohn (2009) for weighted voting games. All the new algorithms employ an algorithm for computing the PGI efficiently which was recently proposed in Staudacher et al. (2021) and supersedes the previous state-of-the-art approach from Matsui and Matsui (2000). Even though further algorithmic improvements for both the Threat Public Good indices 2 and 3 and the Owen Extended PGI (OPGI) might be possible, we emphasize that our current C++ implementations can be applied for large numbers of players. With the software EPIC Staudacher et al. (2021, 2022) for weighted voting games (with and without precoalitions) and the R package CoopGame Staudacher and Anwander (2021) providing a prototypical implementation of the Public Value by Holler and Li (1995) for cooperative games with transferable utility, there is now publicly available software for many solution concepts proposed by Manfred Holler.

Still, open questions in the context of power index computation abound. Overcoming the limitations of weighted voting games and solving other classes of simple games efficiently is one of them. Wilms (2020) presents dynamic programming algorithms for computing Banzhaf and Shapley–Shubik indices for conjunctions and disjunctions of weighted voting games. It appears rewarding to expand these ideas to other power indices, including the PGI and power indices with precoalitions. Wilms (2020) compares his algorithms with quasi-ordered binary decision diagrams (QOBDDs) Bolus (2011, 2012), i.e. a recent technique based on relational algebra, which has not yet been extended to power indices with precoalitions. These binary decision diagrams provide representations of weighted voting games which come with computational costs for their generation, but are independent of the integer weights and quota. Hence, Wilms (2020) speaks of a minimum quota effect meaning that when using QOBDDs the same bounds for the quota and the sum of all weights can be used as for a minimum sum representation Freixas and Molinero (2009) of the weighted voting game. As we confirmed in Sect. 5, our new algorithms benefit from small integer weights and quotas. In all cases except for OPGI, a smaller quota implies lower storage requirements.

As stated in our introduction, dynamic programming and generating functions Algaba et al. (2007); Alonso-Meijide and Bowles (2005); Alonso-Meijide et al. (2008); Bilbao et al. (2000) are strongly related. The publicly available doctoral dissertation by Lindner (2004) points out in detail how we can use the recursion from Theorem 1 to find the coefficients of a corresponding generating function and elaborates relations between generating functions and recursions used in dynamic programming algorithms for the Banzhaf and Shapley–Shubik indices. While clearly beyond the scope of this article, studying the mathematical relations between dynamic programming and generating functions for power index computation, including the PGI and its variants with precoalitions, appears to be a very promising field of future research. It bears the potential to lead to even faster algorithms. Furthermore, it could be interesting to study whether a recent result by Koiliaris and Xu (2019) can be used to make the dynamic programming algorithms from Staudacher et al. (2021, 2022) even faster.

Another promising task could be the parallel computation of power indices. Our new algorithms for PGIs with precoalitions presented in the proofs of Theorems 6, 7 and 8 appear to be suitable for parallel processing, given that one can compute internal games independently. We also note that dynamic programming algorithms for the Deegan–Packel index with precoalitions Alonso-Meijide et al. (2011) and the Johnston index with precoalitions Mercik and Ramsey (2017) for weighted voting games have yet to be developed. In terms of practical applications, the author hopes that the new algorithms proposed in this work will prove useful for extending existing studies of social and economic networks Holler and Rupp (2019, 2020, 2021); Staudacher et al. (2021) to precoalitions among the player set and large numbers of players.

Acknowledgements The author thanks his former student Jan Filipp for his advice and assistance in implementing the six new PGIs with precoalitions in C++ and an anonymous reviewer whose comments and suggestions helped improve the paper.

References

Algaba, E., Bilbao, J., & M.,& Fernández-García, J. R. (2007). The distribution of power in the European Constitution. *European Journal of Operational Research,176*(3), 1752–1766. https://doi.org/10.1016/j.ejor.2005.12.002.

Alonso-Meijide, J. M., & Bowles, C. (2005). Generating functions for coalitional power indices: An application to the IMF. *Annals of Operations Research, 137*(1), 21–44. https://doi.org/10.1007/s10479-005-2242-y.

Alonso-Meijide, J. M., & Fiestras-Janeiro, M. G. (2002). Modification of the Banzhaf value for games with a coalition structure. *Annals of Operations Research, 109*(1), 213–227. https://doi.org/10.1023/A:1016356303622.

Alonso-Meijide, J. M., Casas-Mendez, B., Holler, M. J., & Lorenzo-Freire, S. (2008). Computing power indices: Multilinear extensions and new characterizations. *European Journal of Operational Research, 188*(2), 540–554. https://doi.org/10.1016/j.ejor.2007.04.019.

Alonso-Meijide, J. M., Bowles, C., Holler, M. J., & Napel, S. (2009). Monotonicity of power in games with a priori unions. *Theory & Decision, 66*(1), 17–37. https://doi.org/10.1007/s11238-008-9114-2.

Alonso-Meijide, J. M., Casas-Méndez, B., Fiestras-Janeiro, M. G., Holler, M. J., & Nohn, A. (2010a). Axiomatizations of public good indices with a priori unions. *Social Choice and Welfare, 35*(3), 517–533. https://doi.org/10.1007/s00355-010-0451-z.

Alonso-Meijide, J. M., Casas-Méndez, B., Fiestras-Janeiro, M. G., & Holler, M. J. (2010b). Two variations of the Public Good Index for games with a priori unions. *Control & Cybernetics, 39,* 839–855.

Alonso-Meijide, J. M., Casas-Méndez, B., Fiestras-Janeiro, M. G., & Holler, M. J. (2011). The Deegan-Packel index for simple games with a priori unions. *Quality & Quantity, 45*(2), 425–439. https://doi.org/10.1007/s11135-009-9306-z.

Banzhaf, J. F., III. (1964). Weighted voting doesn't work: A mathematical analysis. *Rutgers L. Rev., 19,* 317–343.

Bertini, C., Freixas, J., Gambarelli, G., & Stach, I. (2013). Comparing power indices. *International Game Theory Review, 15,* 1340004. https://doi.org/10.1142/S0219198913400045.

Bilbao, J. M., Fernández-García, J. F., Jiménez Losada, A., & López, J. J. (2000). Generating functions for computing power indices efficiently. *TOP, 8*(2), 191–213. https://doi.org/10.1007/BF02628555.

Bolus, S. (2011). Power indices of simple games and vector-weighted majority games by means of binary decision diagrams. *European Journal of Operational Research, 210*(2), 258–272. https://doi.org/10.1016/j.ejor.2010.09.020.

Bolus, S. (2012). *A QOBDD-based approach to simple games*. Ph.D. thesis, Christian-Albrechts Universität Kiel, Kiel, Germany. Retrieved May 17, 2022 from https://macau.uni-kiel.de/receive/diss_mods_00009114.

Chakravarty, S. R., Mitra, M., & Sarkar, P. (2015). *A Course on Cooperative Game Theory*. Cambridge University Press.

Deegan, J., & Packel, E. W. (1978). A new index of power for simple n-person games. *International Journal of Game Theory, 7*(2), 113–123.

Freixas, J., & Molinero, X. (2009). On the existence of a minimum integer representation for weighted voting systems. *Annals of Operations Research, 166*(1), 243–260. https://doi.org/10.1007/s10479-008-0422-2.

Holler, M. J., & Rupp, F. (2019). Power in networks: A PGI analysis of Krackhardt's Kite network. In N. Nguyen, R. Kowalczyk, J. Mercik, & A. Motylska-Kuźma (Eds.), *Transactions on Computational Collective Intelligence XXXIV* (Vol. 11890, pp. 21–34). Lecture Notes in Computer Science. Springer. https://doi.org/10.1007/978-3-662-60555-4_2.

Holler, M. J., & Rupp, F. (2020). Power in networks and the urban space. In E. Macrì, V. Morea, & M. Trimarchi M. (Eds.), *Cultural Commons and Urban Dynamics* (pp. 37–52). Springer. https://doi.org/10.1007/978-3-030-54418-8_4.

Holler, M. J. (1982). Forming coalitions and measuring voting power. *Political Studies, 30*(2), 262–271.

Holler, M. J., & Li, X. (1995). From public good index to public value: An axiomatic approach and generalization. *Control & Cybernetics, 24,* 257–270.

Holler, M. J., & Nohn, A. (2009). The Public Good Index with threats in a priori unions. *Essays in Honor of Hannu Nurmi, 1,* 393–401.

Holler, M. J., & Packel, E. W. (1983). Power, luck and the right index. *Zeitschrift für Nationalökonomie, 43*(1), 21–29.

Holler, M. J., & Rupp, F. (2021). Power in networks: The Medici. *Homo Oeconomicus, 38,* 1–17. https://doi.org/10.1007/s41412-021-00108-1.

Johnston, R. J. (1978). On the measurement of power: Some reactions to Laver. *Environment & Planning A*, *10*(8), 907–914.

Kóczy, L. Á. (2021). Brexit and power in the council of the European Union. *Games, 12*(2), 51. https://doi.org/10.3390/g12020051.

Koiliaris, K., & Xu, C. (2019). Faster pseudopolynomial time algorithms for subset sum. *ACM Transactions on Algorithms (TALG), 15*(3), 1–20. https://doi.org/10.1145/3329863.

Kurz, S. (2016). Computing the power distribution in the IMF. arXiv Preprint arXiv: 1603.01443.

Lindner, I. (2004). *Power measures in large weighted voting games*. Ph.D. thesis, Universität Hamburg, Hamburg, Germany. Retrieved May 17, 2022 from https://ediss.sub.uni-hamburg.de/bitstream/ediss/715/1/Dissertation.pdf.

Malawski, M. (2004). “Counting” power indices for games with a priori unions. In G. Gambarelli (Ed.) *Essays in cooperative games: Theory and decision library* (Vol. 36, pp. 125–140). Springer. https://doi.org/10.1007/978-1-4020-2936-3_10.

Matsui, T., & Matsui, Y. (2000). A survey of algorithms for calculating power indices of weighted majority games. *Journal of the Operations Research Society of Japan, 2000*(43), 71–86.

Mercik, J., & Ramsey, D.M. (2017). The effect of Brexit on the balance of power in the European Union Council: An approach based on pre-coalitions. In N. Nguyen, R. Kowalczyk, & J. Mercik, (Eds.) *Transactions on Computational Collective Intelligence XXVII* (Vol. 10480, pp. 87–107). Lecture Notes in Computer Science. Springer. https://doi.org/10.1007/978-3-319-70647-4_7.

Owen, G. (1977). Values of games with a priori unions. In R. Henn, & O. Moeschlin (Eds.), *Mathematical Economics and Game Theory* (Vol. 141, pp. 76–88). Lecture Notes in Economics and Mathematical Systems. Springer. https://doi.org/10.1007/978-3-642-45494-3_7.

Owen, G. (1981). Modification of the Banzhaf-Coleman index for games with a priori unions. In M. J. Holler (Ed.) *Power, Voting, and Voting Power* (pp. 232–238). Physica. https://doi.org/10.1007/978-3-662-00411-1_17.

Owen, G. (1995). *Game theory* (3rd ed.). Academic Press.

Shapley, L. S. (1953). A value for n-person games. *Contributions to the Theory of Games, 28*(2), 307–317.

Shapley, L. S., & Shubik, M. (1954). A method for evaluating the distribution of power in a committee system. *American Political Science Review, 48*(3), 787–792.

Staudacher, J., & Anwander J. (2021). Using the R package CoopGame for the analysis, solution and visualization of cooperative games with transferable utility. Retrieved May 17, 2022 from https://cran.r-project.org/package=CoopGame. R Vignette for package version 0.2.2.

Staudacher, J., Olsson, L., & Stach, I. (2021). Implicit power indices for measuring indirect control in corporate structures. In N. Nguyen, R. Kowalczyk, J. Mercik, & A. Motylska-Kuźma (Eds.), *Transactions on Computational Collective Intelligence XXXVI* (Vol. 13010, pp. 73–93). Lecture Notes in Computer Science. Springer. https://doi.org/10.1007/978-3-662-64563-5_4.

Staudacher, J., Kóczy, L. Á., Stach, I., Filipp, J., Kramer, M., Noffke, T., et al. (2021). Computing power indices for weighted voting games via dynamic programming. *Operations Research and Decisions, 31*(2), 123–145. https://doi.org/10.37190/ord210206.

Staudacher, J., Wagner, F., & Filipp, J. (2022). Dynamic programming for computing power indices for weighted voting games with precoalitions. *Games, 13*(1), 6. https://doi.org/10.3390/g13010006.

Tanenbaum, P. (1997). Power in weighted voting games. *Mathematica Journal, 7,* 58–63.

Uno, T. (2012). Efficient computation of power indices for weighted majority games. In K. M. Chao, T. Hsu, & D. T. Lee (Eds.), *International Symposium on Algorithms and Computation. ISAAC 2012* (Vol. 7676, pp. 679–689). Lecture Notes in Computer Science. Springer. https://doi.org/10.1007/978-3-642-35261-4_70.

Wilms, I. (2020). Dynamic programming algorithms for computing power indices in weighted multi-tier games. *Mathematical Social Sciences, 108,* 175–192. https://doi.org/10.1016/j.mathsocsci.2020.06.004.