Weighted voting games

Spring 2024

1. Weighted voting games

Weighted voting games

- A Weighted voting game (WVG) is a simple game for which there exists a quota q and it is possible to assign to each $i \in N$ a weight w_i, so that

$$X \in \mathcal{W} \text{ iff } \sum_{i \in X} w_i \geq q.$$

Weighted voting games

- A Weighted voting game (WVG) is a simple game for which there exists a quota q and it is possible to assign to each $i \in N$ a weight w_i, so that

$$X \in \mathcal{W} \text{ iff } \sum_{i \in X} w_i \geq q.$$

A explicit representation as $[q; w]$ for a WVG Γ is called a realization of Γ.

Weighted voting games: equivalent realizations

Weighted voting games: equivalent realizations

Weighted voting games: equivalent realizations

- Two realizations with different weights and quotas, might encode the same WVG i.e., have exactly the same set of winning coalitions.

Weighted voting games: equivalent realizations

- Two realizations with different weights and quotas, might encode the same WVG i.e., have exactly the same set of winning coalitions.
- For instance, this is the case if one game is obtained from the other by scaling up all weights and the quota by the same factor.

Weighted voting games: equivalent realizations

- Two realizations with different weights and quotas, might encode the same WVG i.e., have exactly the same set of winning coalitions.
- For instance, this is the case if one game is obtained from the other by scaling up all weights and the quota by the same factor.
- But $\Gamma_1 = [5; 3, 3]$ and $\Gamma_2 = [2; 1, 1]$ also describe the same game.

Weighted voting games: equivalent realizations

- Two realizations with different weights and quotas, might encode the same WVG i.e., have exactly the same set of winning coalitions.
- For instance, this is the case if one game is obtained from the other by scaling up all weights and the quota by the same factor.
- But $\Gamma_1 = [5; 3, 3]$ and $\Gamma_2 = [2; 1, 1]$ also describe the same game.
- Two realizations $[q_1; w_1]$ and $[q_2; w_2]$ on the same set N players are equivalent if, for $S \subseteq N$, $w_1(S) \geq q_1$ iff $w_2(S) \geq q_2$.

Weighted voting games: equivalent realizations

- Two realizations with different weights and quotas, might encode the same WVG i.e., have exactly the same set of winning coalitions.
- For instance, this is the case if one game is obtained from the other by scaling up all weights and the quota by the same factor.
- But $\Gamma_1 = [5; 3, 3]$ and $\Gamma_2 = [2; 1, 1]$ also describe the same game.
- Two realizations $[q_1; w_1]$ and $[q_2; w_2]$ on the same set N players are equivalent if, for $S \subseteq N$, $w_1(S) \geq q_1$ iff $w_2(S) \geq q_2$.
- The notion of equivalence naturally extends to other representations forms for simple games.

Integrality of Weights and Quota

Theorem

For any weighted voting game $\Gamma = [q; w]$ *with* $|N| = n$*, there exists an equivalent weighted voting game* $\Gamma' = [q'; w']$ *such that*

- Γ *and* Γ' *are equivalents,*
- $w' \in (\mathbb{Z}^+)^n$ *and* $q' \in \mathbb{Z}^+$*, and*
- $w'_{max} = O(2^{n \log n})$*.*

[Carreras and Freixas, Math. Soc.Sci., 1996]
It can be deduced from
[S. Muroga. Threshold Logic and its Applications, 1971].

Representation

WVG by $[q; w]$

Representation

WVG by $[q; w]$

- We can assume without loss of generality that all weights and the quota are integers given in binary.

Representation

WVG by $[q; w]$

- We can assume without loss of generality that all weights and the quota are integers given in binary.
- Even though the entries of the weight vector w are exponential in n, they can be represented using $O(nlogn)$ bits

Representation

WVG by $[q; w]$

- We can assume without loss of generality that all weights and the quota are integers given in binary.
- Even though the entries of the weight vector w are exponential in n, they can be represented using $O(nlogn)$ bits
- Thus, a weighted voting game with n players can be described using $poly(n)$ bits.

Representation

WVG by $[q; w]$

- We can assume without loss of generality that all weights and the quota are integers given in binary.
- Even though the entries of the weight vector w are exponential in n, they can be represented using $O(nlogn)$ bits
- Thus, a weighted voting game with n players can be described using $poly(n)$ bits.
- We write $w(C)$ to denote the total weight of a coalition C, i.e., we set $w(C) = \sum_{i \in C} w_i$

Representation

WVG by $[q; w]$

- We can assume without loss of generality that all weights and the quota are integers given in binary.
- Even though the entries of the weight vector w are exponential in n, they can be represented using $O(nlogn)$ bits
- Thus, a weighted voting game with n players can be described using $poly(n)$ bits.
- We write $w(C)$ to denote the total weight of a coalition C, i.e., we set $w(C) = \sum_{i \in C} w_i$
- We set $w_{max} = \max_{i \in N} w_i$

IsSimple and IsStrong

A simple game $(N, \mathcal{W})$ is

- strong if $S \notin \mathcal{W}$ implies $N \setminus S \in \mathcal{W}$.
- proper if $S \in \mathcal{W}$ implies $N \setminus S \notin \mathcal{W}$.

IsSimple and IsStrong

- We analyze the complexity of the IsProper and IsStrong problems when the input game is an integer realization $[q; w]$ of a WVG Γ.

IsSimple and IsStrong

- We analyze the complexity of the IsProper and IsStrong problems when the input game is an integer realization $[q; w]$ of a WVG Γ.
- Some of the hardness proofs are based on reductions from the NP-complete problem:

IsSimple and IsStrong

- We analyze the complexity of the IsProper and IsStrong problems when the input game is an integer realization $[q; w]$ of a WVG Γ.
- Some of the hardness proofs are based on reductions from the NP-complete problem:
  Name: Partition
  Input: n integer values, $x_1, \ldots, x_n$
  Question: Is there $S \subseteq \{1, \ldots, n\}$ *for which*

$$\sum_{i \in S} x_i = \sum_{i \notin S} x_i.$$

IsSimple and IsStrong

- We analyze the complexity of the IsProper and IsStrong problems when the input game is an integer realization $[q; w]$ of a WVG Γ.
- Some of the hardness proofs are based on reductions from the NP-complete problem:
  Name: Partition
  Input: n integer values, $x_1, \ldots, x_n$
  Question: Is there $S \subseteq \{1, \ldots, n\}$ *for which*

$$\sum_{i \in S} x_i = \sum_{i \notin S} x_i.$$

- Observe that, for any instance of the Partition problem in which the sum of the n input numbers is odd, the answer must be no.

IsSimple and IsStrong

Theorem

The ISSTRONG *and the* ISPROPER *problems, when the input is described by an integer realization of a weighted game* $[q; w]$*, are coNP-complete.*

IsSimple and IsStrong

Theorem

The IsStrong *and the* IsProper *problems, when the input is described by an integer realization of a weighted game* $[q; w]$*, are coNP-complete.*

- From the definitions of strong, proper it is straightforward to show that both problems belong to coNP.

IsSimple and IsStrong

Theorem

The IsStrong *and the* IsProper *problems, when the input is described by an integer realization of a weighted game* $[q; w]$*, are coNP-complete.*

- From the definitions of strong, proper it is straightforward to show that both problems belong to coNP.
- Observe that the weighted game with integer representation $(2; 1, 1, 1)$ is both proper and strong.

Hardness

We transform an instance $x = (x_1, \ldots, x_n)$ of PARTITION into a realization of a weighted game according to the following schema

$$f(x) = \begin{cases} (q(x); x) & \text{when } x_1 + \cdots + x_n \text{ is even,} \\ (2; 1, 1, 1) & \text{otherwise.} \end{cases}$$

Hardness

We transform an instance $x = (x_1, \ldots, x_n)$ of PARTITION into a realization of a weighted game according to the following schema

$$f(x) = \begin{cases} (q(x); x) & \text{when } x_1 + \cdots + x_n \text{ is even,} \\ (2; 1, 1, 1) & \text{otherwise.} \end{cases}$$

- Function f can be computed in polynomial time provided q does.

Hardness

We transform an instance $x = (x_1, \ldots, x_n)$ of PARTITION into a realization of a weighted game according to the following schema

$$f(x) = \begin{cases} (q(x); x) & \text{when } x_1 + \cdots + x_n \text{ is even,} \\ (2; 1, 1, 1) & \text{otherwise.} \end{cases}$$

- Function f can be computed in polynomial time provided q does.
- Independently of q, when $x_1 + \cdots + x_n$ is *odd*, x is a NO input for partition, but $f(x)$ is a YES instance of ISSTRONG or ISPROPER.

IsStrong

Assume that $x_1 + \cdots + x_n$ is *even*.
Let $s = (x_1 + \cdots + x_n)/2$ and $N = \{1, \ldots, n\}$.
Set $q(x) = s + 1$.

IsStrong

Assume that $x_1 + \cdots + x_n$ is *even*.
Let $s = (x_1 + \cdots + x_n)/2$ and $N = \{1, \ldots, n\}$.
Set $q(x) = s + 1$.

- If there is $S \subset N$ such that $\sum_{i \in S} x_i = s$, then $\sum_{i \notin S} x_i = s$, thus both S and $N \setminus S$ are losing coalitions and $f(x)$ is not strong.

IsStrong

Assume that $x_1 + \cdots + x_n$ is *even*.
Let $s = (x_1 + \cdots + x_n)/2$ and $N = \{1, \ldots, n\}$.
Set $q(x) = s + 1$.

- If there is $S \subset N$ such that $\sum_{i \in S} x_i = s$, then $\sum_{i \notin S} x_i = s$, thus both S and $N \setminus S$ are losing coalitions and $f(x)$ is not strong.
- If S and $N \setminus S$ are losing coalitions in $f(x)$.
  If $\sum_{i \in S} x_i < s$ then $\sum_{i \notin S} x_i \geq s + 1$, $N \setminus S$ should be winning.
  Thus $\sum_{i \in S} x_i = \sum_{i \notin S} x_i = s$, and there exists a partition of x.

IsProper

Assume that $x_1 + \cdots + x_n$ is *even*.
Let $s = (x_1 + \cdots + x_n)/2$ and $N = \{1, \ldots, n\}$.
Set $q(x) = s$.

IsProper

Assume that $x_1 + \cdots + x_n$ is *even*.
Let $s = (x_1 + \cdots + x_n)/2$ and $N = \{1, \ldots, n\}$.
Set $q(x) = s$.

- If there is $S \subset N$ such that $\sum_{i \in S} x_i = s$, then $\sum_{i \notin S} x_i = s$, thus both S and $N \setminus S$ are winning coalitions and $f(x)$ is not proper.

IsProper

Assume that $x_1 + \cdots + x_n$ is *even*.
Let $s = (x_1 + \cdots + x_n)/2$ and $N = \{1, \ldots, n\}$.
Set $q(x) = s$.

- If there is $S \subset N$ such that $\sum_{i \in S} x_i = s$, then $\sum_{i \notin S} x_i = s$, thus both S and $N \setminus S$ are winning coalitions and $f(x)$ is not proper.
- When $f(x)$ is not proper

$$\exists S \subseteq N : \sum_{i \in S} x_i \geq s \wedge \sum_{i \notin S} x_i \geq s,$$

  and thus $\sum_{i \in S} x_i = s$.

Power and weight

- We have argued that the power of player i in a coalitional game can be measured by the Shapley value ϕ_i or Banzhaf index β_i.

Power and weight

- We have argued that the power of player i in a coalitional game can be measured by the Shapley value ϕ_i or Banzhaf index β_i.
- If the game in question is a WVG, one may expect that ϕ_i is closely related to w_i.

Power and weight

- We have argued that the power of player i in a coalitional game can be measured by the Shapley value ϕ_i or Banzhaf index β_i.
- If the game in question is a WVG, one may expect that ϕ_i is closely related to w_i.
- It is not hard to show that power is monotone in weight, i.e., for any weighted voting game $\Gamma = [q; w]$ and any two players $i, j \in N$, we have $\phi_i(\Gamma) \leq \phi_j(\Gamma)$ iff $w_i \leq w_j$.

Power and weight

- We have argued that the power of player i in a coalitional game can be measured by the Shapley value ϕ_i or Banzhaf index β_i.
- If the game in question is a WVG, one may expect that ϕ_i is closely related to w_i.
- It is not hard to show that power is monotone in weight, i.e., for any weighted voting game $\Gamma = [q; w]$ and any two players $i, j \in N$, we have $\phi_i(\Gamma) \leq \phi_j(\Gamma)$ iff $w_i \leq w_j$.
- However, two agents may have identical voting power even if their weights differ considerably.

Power and weight

- After the May 2010 elections in the UK, the Conservative Party had 307 seats, the Labour Party had 258 seats, the Liberal Democrats (LibDems) had 57 seats, and all other parties shared the remaining 28 seats (with the most powerful of them getting 8 seats).

Power and weight

- After the May 2010 elections in the UK, the Conservative Party had 307 seats, the Labour Party had 258 seats, the Liberal Democrats (LibDems) had 57 seats, and all other parties shared the remaining 28 seats (with the most powerful of them getting 8 seats).
- It is easy to see that in this weighted voting game there are two two-party coalitions (Conservatives+Labour and Conservatives+LibDems) that can get a majority of seats.

Power and weight

- After the May 2010 elections in the UK, the Conservative Party had 307 seats, the Labour Party had 258 seats, the Liberal Democrats (LibDems) had 57 seats, and all other parties shared the remaining 28 seats (with the most powerful of them getting 8 seats).
- It is easy to see that in this weighted voting game there are two two-party coalitions (Conservatives+Labour and Conservatives+LibDems) that can get a majority of seats.
- Moreover, if Labour or LibDems want to form a coalition that does not include Conservatives, they need each other (as well as a few minor parties).

Power and weight

- After the May 2010 elections in the UK, the Conservative Party had 307 seats, the Labour Party had 258 seats, the Liberal Democrats (LibDems) had 57 seats, and all other parties shared the remaining 28 seats (with the most powerful of them getting 8 seats).
- It is easy to see that in this weighted voting game there are two two-party coalitions (Conservatives+Labour and Conservatives+LibDems) that can get a majority of seats.
- Moreover, if Labour or LibDems want to form a coalition that does not include Conservatives, they need each other (as well as a few minor parties).
- Thus, Labour and LibDems have the same Shapley value, despite being vastly different in size.

Power and weight

- The phenomenon illustrated in the previous example explains why major parties often end up making concessions to smaller parties in order to form a winning coalition: the small parties may wield substantial voting power.

Power and weight

- The phenomenon illustrated in the previous example explains why major parties often end up making concessions to smaller parties in order to form a winning coalition: the small parties may wield substantial voting power.
- To determine a player's power, we have to take into account the distribution of the other players' weights as well as the quota.

Power and weight

Power and weight

- Consider a weighted voting game with $w = (4, 4, 1, 1)$.

Power and weight

- Consider a weighted voting game with $w = (4, 4, 1, 1)$.
- Setting $q = 10$, only the grand coalition wins, so all Shapley values are $1/4$.

Power and weight

- Consider a weighted voting game with $w = (4, 4, 1, 1)$.
- Setting $q = 10$, only the grand coalition wins, so all Shapley values are $1/4$.
- Setting $q = 8$, the smaller players are dummies, so their Shapley value is 0.

Power and weight

- Consider a weighted voting game with $w = (4, 4, 1, 1)$.
- Setting $q = 10$, only the grand coalition wins, so all Shapley values are $1/4$.
- Setting $q = 8$, the smaller players are dummies, so their Shapley value is 0.
- Setting $q = 5$, a player of weight 1 is pivotal only if it appears in the second position, and a player of weight 4 appears in the first position. There are four permutations that satisfy this condition, so the Shapley value of each of the smaller players is $1/6$.

Power and weight:duality

The dual of a game $\Gamma = (N, \mathcal{W})$ is the game $\Gamma^d = (N, \mathcal{W}^d)$ where $\mathcal{W}^d = \{S \subseteq N \mid N \setminus S \notin \mathcal{W}\}$.

A coalition S is blocking if $N \setminus S \notin \mathcal{W}$

Lemma

Given a WVG $\Gamma = [q; w]$, we have

- $[w(N) + 1 - q; w]$ *is a representation of* Γ^d.
- *for each* $i \in N$, $\phi_i(\Gamma) = \phi_i(\Gamma^d)$

Power and weight

Proof.

Let us see that $[w(N) + 1 - q; w]$ is a representation of Γ^d.

Power and weight

Proof.

Let us see that $[w(N) + 1 - q; w]$ is a representation of Γ^d.

- Assume that S is a blocking coalition, we know that $N \setminus S$ loses in Γ.

Power and weight

Proof.

Let us see that $[w(N) + 1 - q; w]$ is a representation of Γ^d.

- Assume that S is a blocking coalition, we know that $N \setminus S$ loses in Γ.
- i.e., $w(N \setminus S) < q$, i.e., $w(N) - w(S) < q$.

Power and weight

Proof.

Let us see that $[w(N) + 1 - q; w]$ is a representation of Γ^d.

- Assume that S is a blocking coalition, we know that $N \setminus S$ loses in Γ.
- i.e., $w(N \setminus S) < q$, i.e., $w(N) - w(S) < q$.
- giving $w(S) > w(N) - q$,

Power and weight

Proof.

Let us see that $[w(N) + 1 - q; w]$ is a representation of Γ^d.

- Assume that S is a blocking coalition, we know that $N \setminus S$ loses in Γ.
- i.e., $w(N \setminus S) < q$, i.e., $w(N) - w(S) < q$.
- giving $w(S) > w(N) - q$,
- but as all the numbers are integers, equivalently, $w(S) \geq 1 + w(N) - q$

Power and weight

Proof.

Let us see that $[w(N)+1-q; w]$ is a representation of Γ^d.

- Assume that S is a blocking coalition, we know that $N \setminus S$ loses in Γ.
- i.e., $w(N \setminus S) < q$, i.e., $w(N) - w(S) < q$.
- giving $w(S) > w(N) - q$,
- but as all the numbers are integers, equivalently, $w(S) \geq 1 + w(N) - q$
- So, $S \in \mathcal{W}^d$ iff $w(S) \geq 1 + w(N) - q$.

Power and weight

Let us prove that, for each $i \in N$, $\phi_i(\Gamma) = \phi_i(\Gamma^d)$

Power and weight

Let us prove that, for each $i \in N$, $\phi_i(\Gamma) = \phi_i(\Gamma^d)$

- Suppose, that for a permutation π and a player i, $w(S_\pi(i)) < q$ and $w(S_\pi(i) \cup \{i\}) \geq q$.

Power and weight

Let us prove that, for each $i \in N$, $\phi_i(\Gamma) = \phi_i(\Gamma^d)$

- Suppose, that for a permutation π and a player i, $w(S_\pi(i)) < q$ and $w(S_\pi(i) \cup \{i\}) \geq q$.
- Let π' be the permutation obtained by reversing π, we have $w(S_{\pi'}(i)) = w(N) - w(S_\pi(i)) - w_i \leq w(N) - q < w(N) - q + 1$, $w(S_{\pi'}(i) \cup \{i\}) = w(N) - w(S_\pi(i)) > w(N) - q \geq w(N) - q + 1$.

Power and weight

Let us prove that, for each $i \in N$, $\phi_i(\Gamma) = \phi_i(\Gamma^d)$

- Suppose, that for a permutation π and a player i, $w(S_\pi(i)) < q$ and $w(S_\pi(i) \cup \{i\}) \geq q$.
- Let π' be the permutation obtained by reversing π, we have $w(S_{\pi'}(i)) = w(N) - w(S_\pi(i)) - w_i \leq w(N) - q < w(N) - q + 1$, $w(S_{\pi'}(i) \cup \{i\}) = w(N) - w(S_\pi(i)) > w(N) - q \geq w(N) - q + 1$.
- Hence, i is pivotal for the permutation π' in the game Γ^d.

Power and weight

Let us prove that, for each $i \in N$, $\phi_i(\Gamma) = \phi_i(\Gamma^d)$

- Suppose, that for a permutation π and a player i, $w(S_\pi(i)) < q$ and $w(S_\pi(i) \cup \{i\}) \geq q$.
- Let π' be the permutation obtained by reversing π, we have $w(S_{\pi'}(i)) = w(N) - w(S_\pi(i)) - w_i \leq w(N) - q < w(N) - q + 1$, $w(S_{\pi'}(i) \cup \{i\}) = w(N) - w(S_\pi(i)) > w(N) - q \geq w(N) - q + 1$.
- Hence, i is pivotal for the permutation π' in the game Γ^d.
- By symmetry, the converse is also true.

Power and weight

Let us prove that, for each $i \in N$, $\phi_i(\Gamma) = \phi_i(\Gamma^d)$

- Suppose, that for a permutation π and a player i, $w(S_\pi(i)) < q$ and $w(S_\pi(i) \cup \{i\}) \geq q$.
- Let π' be the permutation obtained by reversing π, we have $w(S_{\pi'}(i)) = w(N) - w(S_\pi(i)) - w_i \leq w(N) - q < w(N) - q + 1$, $w(S_{\pi'}(i) \cup \{i\}) = w(N) - w(S_\pi(i)) > w(N) - q \geq w(N) - q + 1$.
- Hence, i is pivotal for the permutation π' in the game Γ^d.
- By symmetry, the converse is also true.
- Thus, we have established a bijection between the set of permutations that i is pivotal for in Γ and the set of permutations that i is pivotal for in Γ^d.

EndProof.

Auxiliary results

- For n integer values $w = (w_1, \ldots, w_n)$ and an integer x, let $T_w(i, x)$ be the number of possibilities to write the integer x as the sum of some subset of the first i weights.

Auxiliary results

- For n integer values $w = (w_1, \ldots, w_n)$ and an integer x, let $T_w(i, x)$ be the number of possibilities to write the integer x as the sum of some subset of the first i weights.
- Let $\bar{w} = \sum_{i=1}^{n} w_i$

Auxiliary results

- For n integer values $w = (w_1, \dots, w_n)$ and an integer x, let $T_w(i, x)$ be the number of possibilities to write the integer x as the sum of some subset of the first i weights.
- Let $\bar{w} = \sum_{i=1}^{n} w_i$
- When $x > \bar{w}$, $T_w(i, x) = 0$

Auxiliary results

Given w, the values $T_w(i,x)$, for $0 \leq i \leq n$ and $0 \leq x \leq \bar{w}$, can be computed in time $O(xn)$

Auxiliary results

Given w, the values $T_w(i, x)$, for $0 \leq i \leq n$ and $0 \leq x \leq \bar{w}$, can be computed in time $O(xn)$

- We can use dynamic programming over the following recurrence.

Auxiliary results

Given w, the values $T_w(i,x)$, for $0 \leq i \leq n$ and $0 \leq x \leq \bar{w}$, can be computed in time $O(xn)$

- We can use dynamic programming over the following recurrence.

$$T_w(i,x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{if } x > 0 \text{ and } i = 0 \\ T_w(i-1,x) & \text{if } x < w_i \text{ and } i > 0 \\ T_w(i-1,x) + T_w(i-1,x-w_i) & \text{otherwise} \end{cases}$$

Auxiliary results

Given w, the values $T_w(i,x)$, for $0 \le i \le n$ and $0 \le x \le \bar{w}$, can be computed in time $O(xn)$

- We can use dynamic programming over the following recurrence.

$$T_w(i,x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{if } x > 0 \text{ and } i = 0 \\ T_w(i-1,x) & \text{if } x < w_i \text{ and } i > 0 \\ T_w(i-1,x) + T_w(i-1,x-w_i) & \text{otherwise} \end{cases}$$

- The table has size $O(xn)$ and each element can be computed in $O(1)$ filling the table by rows.

Auxiliary results

- For integers x, c, let $C_w(i, x)$ be the number of possibilities to write the integer x as the sum of some subset with cardinality c of the first i weights.

Auxiliary results

- For integers x, c, let $C_w(i, x)$ be the number of possibilities to write the integer x as the sum of some subset with cardinality c of the first i weights.
- When $x > \bar{w}$, or $c > n$, the values are 0.

Auxiliary results

Given w, the values $C_w(i,x)$, for $0 \leq i \leq n$, $0 \leq x \leq \bar{w}$, and $0 \leq c \leq n$ can be computed in time $O(xn^2)$

Auxiliary results

Given w, the values $C_w(i,x)$, for $0 \leq i \leq n$, $0 \leq x \leq \bar{w}$, and $0 \leq c \leq n$ can be computed in time $O(xn^2)$

- We can use dynamic programming over the following recurrence.

$$C(i,x,c) = \begin{cases} 1 & \text{for } x = 0 \\ 0 & \text{if } x > 0 \text{ and } i = 0 \\ 0 & \text{if } x > 0 \; i > 0 \text{ and } c = 0 \\ C(i-1,x,c) & \text{for } 1 \leq i \leq n,\ 1 \leq x < w_i, \\ & \text{and } 1 \leq c \leq n \\ C(i-1,x,c) + C(i-1,x-w_i,c-1) & \text{otherwise} \end{cases}$$

Auxiliary results

Given w, the values $C_w(i,x)$, for $0 \leq i \leq n$, $0 \leq x \leq \bar{w}$, and $0 \leq c \leq n$ can be computed in time $O(xn^2)$

- We can use dynamic programming over the following recurrence.

$$C(i,x,c) = \begin{cases} 1 & \text{for } x = 0 \\ 0 & \text{if } x > 0 \text{ and } i = 0 \\ 0 & \text{if } x > 0 \ i > 0 \text{ and } c = 0 \\ C(i-1,x,c) & \text{for } 1 \leq i \leq n,\ 1 \leq x < w_i, \\ & \text{and } 1 \leq c \leq n \\ C(i-1,x,c) + C(i-1,x-w_i,c-1) & \text{otherwise} \end{cases}$$

- The table has size $O(xn^2)$ and each element can be computed in $O(1)$ filling the table in an adequate order.

Auxiliary results

- Note that in T once a row is computed, we do not need any of the previous rows to compute the next row.

Auxiliary results

- Note that in T once a row is computed, we do not need any of the previous rows to compute the next row.
- This allows for the design of algorithms that consume the values as they are computed but do no require to store the complete table.

Definitions

For a simple game $\Gamma = (N, \mathcal{W})$,

- player i is critical for coalition S if $S \in \mathcal{W}$ and $S - \{i\} \in \mathcal{L}$

Definitions

For a simple game $\Gamma = (N, \mathcal{W})$,

- player i is critical for coalition S if $S \in \mathcal{W}$ and $S - \{i\} \in \mathcal{L}$
- $\eta_i(\Gamma)$ is the number of coalitions for which i is critical.

Definitions

For a simple game $\Gamma = (N, \mathcal{W})$,

- player i is critical for coalition S if $S \in \mathcal{W}$ and $S - \{i\} \in \mathcal{L}$
- $\eta_i(\Gamma)$ is the number of coalitions for which i is critical.
- $\mathcal{W}_i$ is the set of winning coalitions containing i

Definitions

For a simple game $\Gamma = (N, \mathcal{W})$,

- player i is critical for coalition S if $S \in \mathcal{W}$ and $S - \{i\} \in \mathcal{L}$
- $\eta_i(\Gamma)$ is the number of coalitions for which i is critical.
- $\mathcal{W}_i$ is the set of winning coalitions containing i
- The Banzhaf value is $\beta_i(\Gamma) = \eta_i(\Gamma)/2^{n-1}$

Computing the Banzhaf value

Lemma

For a WVG $\Gamma = (N, \mathcal{W})$ given by an integer realization $[q; w]$, the quantites $\eta_i(\Gamma)$ and $|\mathcal{W}_i|$, for $i \in N$, and $|\mathcal{W}|$ can be computed in $O(\Delta n)$ time and $O(\Delta)$ space, where $\Delta = \min(q, \bar{w} - q + 1)$.

Computing the Banzhaf value

Proof.

Case $q = \min(q, \bar{w} - q + 1)$ $(q \leq (\bar{w} + 1)/2)$

Computing the Banzhaf value

Proof.

Case $q = \min(q, \bar{w} - q + 1)$ $(q \leq (\bar{w} + 1)/2)$

- We compute the vector $T(n, x) = T_w(n, x)$, for $0 \leq x \leq q - 1$

Computing the Banzhaf value

Proof.

Case $q = \min(q, \bar{w} - q + 1)$ $(q \leq (\bar{w} + 1)/2)$

- We compute the vector $T(n, x) = T_w(n, x)$, for $0 \leq x \leq q - 1$
- Let $T_{-i}(x)$ be the number of losing coalitions $S \in \mathcal{L}$, with $w(S) = x$ and $i \notin S$.

Computing the Banzhaf value

Proof.

Case $q = \min(q, \bar{w} - q + 1)$ $(q \leq (\bar{w} + 1)/2)$

- We compute the vector $T(n, x) = T_w(n, x)$, for $0 \leq x \leq q - 1$
- Let $T_{-i}(x)$ be the number of losing coalitions $S \in \mathcal{L}$, with $w(S) = x$ and $i \notin S$. These values can be computed recursively as

$$T_{-i}(x) = T(n, x) - T_{-i}(n, x - w_i)$$

Computing the Banzhaf value

Proof.

Case $q = \min(q, \bar{w} - q + 1)$ $(q \leq (\bar{w} + 1)/2)$

- We compute the vector $T(n, x) = T_w(n, x)$, for $0 \leq x \leq q - 1$
- Let $T_{-i}(x)$ be the number of losing coalitions $S \in \mathcal{L}$, with $w(S) = x$ and $i \notin S$. These values can be computed recursively as

$$T_{-i}(x) = T(n, x) - T_{-i}(n, x - w_i)$$

- Then,

$$\eta_i(\Gamma) = \sum_{x=q-w_i}^{q-1} T_{-i}(x)$$

Computing the Banzhaf value

Proof.

Case $q = \min(q, \bar{w} - q + 1)$ $(q \leq (\bar{w} + 1)/2)$

- We compute the vector $T(n, x) = T_w(n, x)$, for $0 \leq x \leq q - 1$
- Let $T_{-i}(x)$ be the number of losing coalitions $S \in \mathcal{L}$, with $w(S) = x$ and $i \notin S$. These values can be computed recursively as

$$T_{-i}(x) = T(n, x) - T_{-i}(n, x - w_i)$$

- Then,

$$\eta_i(\Gamma) = \sum_{x=q-w_i}^{q-1} T_{-i}(x)$$

- All the computation can be done in the desired time bounds.

Computing the Banzhaf value

- When $q > (\bar{w}+1)/2$, we compute $T(n,x)$, for $q \leq x \leq \bar{w}$ indexing the sets by their complements.

Computing the Banzhaf value

- When $q > (\bar{w}+1)/2$, we compute $T(n,x)$, for $q \leq x \leq \bar{w}$ indexing the sets by their complements.
- With a symmetric definition of $T_{+i}(x)$ we can express $\eta_i(\Gamma)$ in a similar way.

Computing the Banzhaf value

- When $q > (\bar{w}+1)/2$, we compute $T(n,x)$, for $q \leq x \leq \bar{w}$ indexing the sets by their complements.
- With a symmetric definition of $T_{+i}(x)$ we can express $\eta_i(\Gamma)$ in a similar way.
- The other values can be expressed as sums of $T_{-i}(x)$ and/or $T_{+i}(x)$

EndProof.

Other power indices

- In a similar way, it can be shown that the Shapley-Shubick index can be computed in $O(\Delta n^2)$ time using $O(\Delta n)$ memory.

Other power indices

- In a similar way, it can be shown that the Shapley-Shubick index can be computed in $O(\Delta n^2)$ time using $O(\Delta n)$ memory.
- Other power indices can be computed using similar techniques, see [Staudacher et al., Operations research and decisions 2:123–145, 2021]
- CoopGame is a R-package implementing most of the results in the paper.