UNIVERSITAT DE BARCELONA
SCHOOL OF ECONOMICS

Working Papers

Col·lecció d'Economia E23/441

# THE COMPLEXITY OF POWER INDICES IN VOTING GAMES WITH INCOMPATIBLE PLAYERS

Martí Jané Ballarín

ISSN 1136-8365

UNIVERSITAT DE BARCELONA
SCHOOL OF ECONOMICS

Working Papers

UB Economics *Working Paper No. 441*

**Title:** The complexity of power indices in voting games with incompatible players

**Abstract:** We study the complexity of computing the Banzhaf index in weighted voting games with cooperation restricted by an incompatibility graph. With an existing algorithm as a starting point, we use concepts from complexity theory to show that, for some classes of incompatibility graphs, the problem can be solved efficiently, as long as the players have "small" weights. We also show that for some other class of graphs it is unlikely that we can find efficient algorithms to compute the Banzhaf index in the corresponding restricted game. Finally, we discuss the complexity of deciding whether the index of a player is non-zero.

**JEL Codes:** C71

**Keywords:** Banzhaf index, Graphs, Algorithms.

**Author:**

Martí Jané Ballarín

Universitat de Barcelona

**Email:**
marti.jane.ballarin@gmail.com

**Date:** February 2023

**Acknowledgements:** I acknowledge the financial support of the Research Group in Game Theory and Assignment Markets of the University of Barcelona through a Màster+ UB scholarship. I am grateful to Mikel Álvarez, for his insight and feedback on this paper. A preliminary version of this paper was presented at the XIX annual meeting of the Spanish Social Choice Network (REES) in November 2022.

# 1. Introduction

The use of cooperative games as a tool to study voting power was pioneered by Lloyd Shapley in the early 1950s. In [45], he introduced the value that now carries his name as a way to measure the expected contribution of a player in a transferable utility game. The following year, in [46], the so-called Shapley-Shubik index was used to assess the influence a member of a committee has over the decisions taken by the group. Years later, in 1965, John Banzhaf (a lawyer) introduced his namesake index in [6]. His goal was also related to measuring political power. Namely, he was concerned with legislative bodies in which each member may cast several votes.

Since then, research has been mostly focused on axiomatic characterizations of these and other power indices. In contrast with the Shapley-Shubik index, the Banzhaf index was not initially introduced in this manner; one possible characterization of the index appears in [37]. At the same time, applications of the indices have mostly stayed within the field of political science. In [7], Banzhaf himself used his index to discuss a voter's power in a presidential election in the United States. Others, like [4, 16], analyze the power political parties wield in a legislative chamber.

The latter references also point towards further lines of research. On one hand, some authors have studied how to model partial cooperation among agents, in contrast with the original framework, which assumed full cooperation. In this regard, in [34] Myerson used graphs to introduce restrictions to cooperation. The games resulting from his approach are called communication games. Another graph-theoretical model of restricted cooperation gives rise to so-called incompatibility games ([10]). The Banzhaf and Shapley-Shubik indices in both of these models have been axiomatically characterized. For communication games, such characterizations can be found in [38] and [34], respectively; for incompatibility games, the Banzhaf index is characterized in [5], while the Shapley-Shubik index is characterized in [10].

On the other hand, part of the literature is devoted to developing algorithms to compute the aforementioned power measures. For instance, [11] uses generating functions to compute the Shapley-Shubik index in weighted voting games with unrestricted cooperation, while [31] solves the problem via dynamic programming. There are algorithms to compute the Banzhaf and Shapley-Shubik indices in games with restricted cooperation as well. For the Shapley-Shubik index in communication games, the proecdure in [2] also uses generating functions. The same technique is used in [4] to compute the Banzhaf and Shapley-Shubik indices in incompatibility games.

In later years, the computational problems posed by these power indices have been approached from a complexity theory perspective. Indeed, there are surveys on this matter, such as [22, 31, 41], which study the complexity of solving game-theoretical computational problems. The aforementioned references focus on the unrestricted cooperation cases; the literature becomes scarce for games with restricted cooperation. For communication games, [9, 47] classify the problems of computing the Banzhaf and Shapley-Shubik indices in this framework. They

also identify subclasses of graphs for which the problem for the corresponding restricted games is easier than in the general case.

In this work, we shall study incompatibility games. Namely, we seek results similar to those in [9, 47]. To this end, we will use the algorithm in [4] as a starting point to assess the complexity of computing power indices in incompatibility games. Analogously to communication games, we will show that the problem is hard in general, but there are meaningful particular cases that are easier.

The following section introduces the necessary concepts to state and prove our results. It is divided in three parts. First, we deal with the graph-theoretical notions that shall define the cooperation restriction models we have hinted at. Next, we discuss some basics on complexity theory. For this subsection, [27] is the main reference regarding counting problems. Namely, its second chapter is a useful tool to get acquainted with such problems and their complexity, and for a brief introduction to the complexity of decision problems. For a deeper look, books like [24, 40] are standard references. The former was used to convey the nuances that arise when encoding computational problems, and provides a detailed guide to the concept of NP-complete problems. In the final part of Section 2 we introduce the game-theoretical models we will be working with; in particular, we properly define the Banzhaf and Shapley-Shubik indices.

In Section 3, we assess the complexity of computing power indices in what we shall call "incompatibility games with very few maximal independent sets". Section 4 fulfills the same goal for incompatibility games with few maximal independent sets, which constitute a broader class. In particular, we study the complexity of computing the Banzhaf index when the complement of the incompatibility graph is planar. Finally, in Section 5 we summarize our results and discuss some possible continuations of this research. In doing so, we suggest candidates for subclasses of graphs on which our study problem becomes easier than in the general case.

# 2. Preliminaries

## 2.1 Graph concepts, properties and classes

We denote an undirected graph without loops (henceforth, simply a *graph*) by $G = (V, E)$, where $V$ is a finite set of *vertices*, and $E$ is a set of *edges*, each represented by an unordered pair of distinct vertices. Given an edge $e = \{u, v\}$, we say $u$ and $v$ are its *endpoints*; moreover, we say $u$ and $v$ are *adjacent* (to each other) and $e$ is *incident* to $u$ and $v$. For convenience, we will always take $V = \{1, \ldots, n\}$ for some integer $n \geqslant 1$. If $E$ is the set of all allowed pairs of vertices, we say $G$ is the *complete* graph with $n$ vertices, and write $G = K_n$. If $E = \varnothing$, $G$ is called an *empty graph*.

A vertex is *isolated* when no other vertex is adjacent to it; we say a vertex to which all other vertices are adjacent is *universal* (or *dominant*). The *degree* of a vertex is the number of vertices to which it is adjacent.

A *subgraph* of $G = (V, E)$ is a graph $H = (S, E')$ such that $S \subseteq V$ and $E'$ is a subset of $E$ such that both endpoints of any $e' \in E'$ are elements of $S$. If $E'$ contains *all* edges in $E$ with both endpoints in $S$, $H$ is the subgraph (of $G$) *induced by* $S$; it is commonplace to denote this graph by $G[S]$. A set of vertices such that no two of them are adjacent (equivalently, the subgraph they induce is empty) is called an *independent set*. An independent set which is not contained in any other such set is called a *maximal* independent set. A set of vertices such that any two of them are adjacent (equivalently, the subgraph they induce is complete) is called a *clique*. Analogously to independent sets, a clique is called maximal if it is not contained in any other clique.

A set of edges such that no two of them share an endpoint is called a *matching*; a matching is called *perfect* when for all vertices of the graph there is an edge in the matching incident to it[1]. A *path* of length $k$ is a sequence of edges $e_1, e_2, \ldots, e_k \in E$ such that there are distinct vertices $v_0, \ldots, v_k \in V$ satisfying $e_i = \{v_{i-1}, v_i\}$, $i \in \{1, \ldots, k\}$. We say such a path joins vertices $v_0$ and $v_k$. A sequence with these same properties but where $v_0 = v_k$ is called a *cycle* (of length $k$).

The *distance* between two vertices of a graph is the length of the shortest path joining them; if no such path exists, the distance is set to infinity. The *diameter* of a graph is the maximum distance between two vertices. A *connected* graph is one such that for any two different vertices, there is a path joining them (equivalently, it has finite diameter). Given $G = (V, E)$, $S \subseteq V$, we say $G[S]$ is a *connected component* of $G$ when $G[S]$ is a connected graph and, for any $T$ such that $S \subsetneq T \subseteq V$, $G[T]$ is not connected. It is a common abuse of notation to say a set of vertices $S \subseteq V$ is connected (or a connected component) when $G[S]$ is connected (or a connected component).

We can also perform some operations involving graphs. Given a graph $G = (V, E)$ and a

[1] In some references, (perfect) matchings are called (maximal) independent edge sets, since the definitions of independent (vertex) sets and matchings are analogous.

subset of vertices $S \subseteq V$, we define $G \setminus S$ as the graph that results from removing all vertices in $S$ from $G$, as well as all edges in $E$ incident to at least one vertex in $S$. Given a subset of edges $E' \subseteq E$, we define $G \setminus E = (V, E \setminus E')$. These two operations are called vertex and edge *deletions*, respectively. We may also *contract* an edge $e = \{u, v\} \in E$. In order to define this operation, let $f$ be a mapping of the vertices of $G$ that maps $u$ and $v$ to the same new vertex, call it $w$, and all other vertices to themselves. The graph resulting from the contraction of edge $e$, denoted by $G/e$, has vertex set $\{f(x) : x \in V\}$ and edge set $\{\{f(x), f(y)\} : \{x, y\} \in E \setminus \{e\}\}$. A graph $H$ that can be obtained by successively applying vertex deletions, edge deletions and edge contractions to $G$ is called a *minor* of $G$.

The *complement* of a graph $G$ is a new graph $\overline{G}$ with the same vertices of $G$, any two of them being adjacent in $\overline{G}$ if and only if they are not adjacent in $G$. It is immediate to see that the properties below hold.

PROPERTY 2.1. Given a graph $G = (V, E)$ and a subset of vertices $S \subseteq V$, $\overline{G[S]} = \overline{G}[S]$. In plain words, the complement of the subgraph of $G$ induced by $S$ is the subgraph of $\overline{G}$ induced by $S$.

PROPERTY 2.2. A set of vertices of a graph $G = (V, E)$ forms a (maximal) independent set in $G$ if and only if it forms a (maximal) clique in $\overline{G}$.

Given a class of graphs $\mathcal{C}$, we will follow [12] and denote by co-$\mathcal{C}$ the class of graphs whose complement is in $\mathcal{C}$.

The *line graph* of $G$, denoted by $L(G)$, is a new graph with the edges of $G$ as vertices, adjacent to each other if and only if they share an endpoint as edges of $G$.[2] This definition, along with those of matchings and independent sets, implies the following fact.

PROPERTY 2.3. There is a bijection between (perfect) matchings in a graph $G$ and (maximal) independent sets in its line graph, $L(G)$.

In order to define some binary graph operations, let $G = (V, E)$, $H = (W, F)$ be two disjoint graphs, that is, with disjoint sets of vertices[3]. The *union* of these two graphs is defined as $G \cup H = (V \cup W, E \cup F)$. The *join* of two graphs is a new graph $G + H$ with the same vertices and edges as $G \cup H$, with an additional edge $\{v, w\}$ for each pair of vertices such that $v \in V$, $w \in W$. If $H$ has only one vertex $w$, we write the union as $G \cup w$ and the join as $G + w$; note that $G \cup w$ is simply adding a new isolated vertex to $G$, while $G + w$ is adding a new universal vertex.

Finally, we introduce some graph classes that will be used in later sections. For a start, we define a *tree* as a connected graph that has no cycles. A graph $G = (V, E)$ is called $k$-*partite*, $k \geqslant 1$, when its vertices can be partitioned into $k$ disjoint independent sets, call them $V_1, \ldots, V_k$. A $k$-partite graph is called complete when it has the maximum possible amount

[2] Some references, such as [44], refer to line graphs as *interchange graphs*.

[3] The operations that follow can also be defined for non-disjoint graphs.

of edges, that is, when any two vertices from different independent sets of the partition[4] are adjacent. Analogously to complete graphs, a complete $k$-partite graph such that the sets of the partition are of sizes $n_1, \ldots, n_k$ is denoted by $K_{n_1,\ldots,n_k}$. If $k = 2$ or $k = 3$, $k$-partite graphs are called *bipartite* or *tripartite*, respectively. A (complete) multipartite graph is a graph that is (complete) $k$-partite for some $k \geqslant 2$.

The *chromatic number* of a graph $G$ is the minimum $k \geqslant 1$ such that $G$ is $k$-partite. A graph $G = (V, E)$ is called *perfect* if for all $S \subseteq V$ the chromatic number of $G[S]$ is equal to the size of the largest clique in $G[S]$. As such, every induced subgraph of a perfect graph is itself a perfect graph. The following properties of graphs of this class are also of interest.

PROPERTY 2.4. Let $G = (V, E)$ be a graph.

(i) If $G$ is bipartite, then both $G$ and $L(G)$ are perfect graphs (the latter due to König's line coloring theorem; see Chapter 1.4 of [30]).

(ii) The complement of $G$ is a perfect graph if and only if $G$ is a perfect graph ([29]).

(iii) Let $w \notin V$. If $G$ is a perfect graph, then so are $G + w$ and $G \cup w$ ([8]).

(iv) Let $v \in V$, $w \notin V$. If $G$ is a perfect graph, then so is $H = (V \cup \{w\}, E \cup \{\{v, w\}\})$. Since $w$ is adjacent to only one of the vertices in $V$, it is called a *pendant vertex* of $H$. Thus, in words, the operation of adding a pendant vertex preserves perfection of a graph ([8]).

Given an integer $d \geqslant 1$, a graph is called $d$-regular when all its vertices have degree $d$. A $k$-partite graph is said to be $(d_1, \ldots, d_k)$-regular when all vertices in the $j$-th set of the partition have degree $d_j$.

A *planar* graph is a graph that admits a planar embedding, that is, a representation of the graph on the plane so that its edges may intersect only at their endpoints. Note that the operations vertex deletion, edge deletion and edge contraction preserve the planarity of a graph; in other words, all minors of a planar graph are planar. In fact, planar graphs can be characterized as those from which there is a set of graphs that cannot be obtained by means of these operations.

PROPERTY 2.5 ([54]). A graph $G$ is planar if and only if neither the complete graph with five vertices, $K_5$, nor the complete bipartite graph $K_{3,3}$ (known as the utility graph) are minors of $G$.

## 2.2 Complexity of decision and counting problems

The main objects of interest in complexity theory are decision problems. A *decision problem* is a computational problem (or question) the solution (or answer) to which is either "yes" or

[4] In general, there may be multiple ways to partition the vertices of a $k$-partite graph into $k$ independent sets; however, such a partition is unique if the graph is complete $k$-partite.

"no". More formally, it is a function $\varphi : \Sigma^* \to \{0, 1\}$ where $\Sigma$ is an *alphabet* in which instances of a problem are encoded; $\Sigma^*$ denotes the set of finite strings of symbols in $\Sigma$.

We say a decision problem $\varphi$ is *decidable* (or *solvable*, or *computable*) when there exists a deterministic Turing machine[5] (or DTM, for short) $M$ that, given an input string $x \in \Sigma^*$ encoding an instance of $\varphi$, halts in a finite number of steps and returns $M(x) = 1$ (or *accepts*) if $\varphi(x) = 1$, and $M(x) = 0$ (or *rejects*) if $\varphi(x) = 0$. In such case, we say $M$ decides (or solves, or computes) $\varphi$.

If the computation of a DTM $M$ halts on a given input $x \in \Sigma^*$, we measure the computation time of $M$ on $x$ as the number of steps it takes for the computation to stop. Given a DTM $M$ that halts on every input $x \in \Sigma^*$, we define its *time complexity function* $T_M : \mathbb{N} \to \mathbb{N}$ so that $T_M(n)$ is the maximum $m \in \mathbb{N}$ for which there is some $x \in \Sigma^*$ of size $n$ such that the computation of $M$ on $x$ takes time $m$. As stated in [24], "the time complexity function [of a Turing machine] expresses its time requirements by giving, for each possible input length, the largest amount of time needed by [the machine] to solve a problem instance of that size".

The *length* $|x|$ of an input string $x \in \Sigma^*$ encoding an instance of some problem is defined as the total number of symbols in $x$. Note that this definition is sensitive to the choice of a preestablished encoding scheme. For example, if the input instance of a problem is a graph, the length of the input string will depend on how this graph is encoded. This motivates the introduction of the more informal notion of *size*.

We define "size" as an encoding independent measure intended to reflect the amount of input data needed to describe an instance of the problem. For the sake of formality, we will consider each problem $\varphi$ associated to a function $\text{Size} : D_\varphi \to \mathbb{N}$ that provides the size of a given instance, where $D_\varphi$ represents the set of instances of problem $\varphi$. Although this function is encoding independent, we require that, under "reasonable" encoding schemes[6], the size of an instance is bounded above by a polynomial function of the length of the string that encodes it, and vice versa. For computational problems whose instances are graphs, the size of an instance is usually defined as the number of vertices of the input graph.

Given two functions $f, g : \mathbb{N} \to \mathbb{N}$, we say $f$ is *big-O* of $g$, and write $f(n) = O(g(n))$, when there exist some $n_0 \in \mathbb{N}$ and a constant $c > 0$ such that $f(n) \leqslant c \cdot g(n)\ \forall n \geqslant n_0$. This notation conveys that the asymptotic growth rate of $f$ is bounded above by that of $g$. A *polynomial time* deterministic Turing machine is one that halts on every input and for which $T_M(n) = O(p(n))$ for some polynomial $p$. Equivalently, for every input instance $I$, $M$ halts in time $O\left(p\left(\text{Size}(I)\right)\right)$. The complexity class $\mathsf{P}$ is defined as the class of decision problems $\varphi$ for which there exists a polynomial time DTM $M$ that solves $\varphi$.

Example 2.1. The question "Is $G$ a bipartite graph?" can be solved in polynomial time in $n$

[5]For our purposes, a deterministic Turing machine can be regarded as a step-by-step procedure used to solve (computational) problems, i.e. an *algorithm*. In spite of this, in this section we will continue to use the phrase "Turing machine" so as not to alter the terminology used in the definitions. The full computational model of the Turing machine, along with some possible augmentations, is detailed in [24, 40].

[6]A fully detailed version of this discussion can be found in [24], along with a loose idea of what constitutes a reasonable encoding scheme.

given an $n$-vertex graph $G = (V, E)$.

EXAMPLE 2.2. There is no known polynomial time algorithm to decide whether a given graph $G = (V, E)$ is $k$-partite for $k > 2$. However, given sets $V_1, \ldots, V_k \subseteq V$, we can check in polynomial time whether they provide a partition of $V$ into $k$ disjoint independent sets.

The complexity class NP is defined as the class of decision problems $\varphi$ for which there exist a polynomial time DTM $M$ and a polynomial $p$, so that $\varphi(x) = 1$ if and only if there is some $y \in \Sigma^*$ of length $|y| \leqslant p(|x|)$ such that $M(\langle x, y\rangle) = 1$, where $\langle x, y\rangle$ denotes the concatenation of strings $x$ and $y$. In this context, we say $y$ is a *witness* for $x$. For $k > 2$, the problem of deciding whether a graph is $k$-partite is in NP. Recall that a graph is $k$-partite if and only if there is a partition of its vertices into at most $k$ disjoint independent sets; we argued in Example 2.2 that, given a candidate partition (a witness), we can decide whether it displays the desired property in polynomial time in the length of the partition.

Note that $\mathsf{P} \subseteq \mathsf{NP}$. However, it is unknown whether $\mathsf{P} = \mathsf{NP}$, in other words, if all decision problems whose solutions can be verified in polynomial time can also be solved in polynomial time. It is conjectured that $\mathsf{P} \neq \mathsf{NP}$; from now on, we will always assume this conjecture to be true.

Under this assumption, we can identify a class that, in a sense, contains the most difficult problems within NP. In order to define this class, we need to introduce a way to relate two problems in terms of their complexity. From [24], "the principal technique used for demonstrating that two decision problems are related is that of *reducing* one to the other, by giving a constructive transformation of any instance of the first problem into an equivalent instance of the second". More formally, we say a decision problem $\varphi_1$ reduces to another problem $\varphi_2$ if there is a transformation $f : \Sigma^* \to \Sigma^*$ such that $\forall x \in \Sigma^*$, $\varphi_1(x) = 1$ if and only if $\varphi_2(f(x)) = 1$.

To be clear, this defines a particular type of reduction, called *many-to-one* reductions; we will omit references to the specific kind of reduction as long as no confusion may arise. If $\varphi_1$ many-to-one reduces to $\varphi_2$ through a transformation $f$ that can be computed by a polynomial time DTM, we say that $\varphi_1$ *polynomial time many-to-one reduces* to $\varphi_2$, and write $\varphi_1 \leqslant_m^{\mathsf{P}} \varphi_2$. If $\varphi_1 \in \mathsf{P}$ and $\varphi_1 \leqslant_m^{\mathsf{P}} \varphi_2$, then $\varphi_2 \in \mathsf{P}$; equivalently, if $\varphi_2 \notin \mathsf{P}$ and $\varphi_1 \leqslant_m^{\mathsf{P}} \varphi_2$, then $\varphi_1 \notin \mathsf{P}$. Thus, we can interpret $\varphi_1 \leqslant_m^{\mathsf{P}} \varphi_2$ as "$\varphi_2$ is at least as hard as $\varphi_1$". If $\varphi_1 \leqslant_m^{\mathsf{P}} \varphi_2$ and $\varphi_2 \leqslant_m^{\mathsf{P}} \varphi_1$, we say $\varphi_1$ and $\varphi_2$ are *polynomially related*, and write $\varphi_1 \sim^{\mathsf{P}} \varphi_2$. Note that $\sim^{\mathsf{P}}$ is an equivalence relation, and $\leqslant_m^{\mathsf{P}}$ induces a partial order on the resulting equivalence classes of decision problems. Among these classes, P corresponds to that of the "easiest" problems; in particular, P is closed under polynomial time many-to-one reductions.

We say a decision problem $\varphi$ is NP-*hard* when every problem $\varphi' \in \mathsf{NP}$ polynomial time reduces to $\varphi$. The class NP-*complete* is the class of problems in NP that are NP-hard. Observe that if some NP-complete problem could be solved in polynomial time, then $\mathsf{P} = \mathsf{NP}$; equivalently, if $\mathsf{P} \neq \mathsf{NP}$, no NP-complete problem can be solved in polynomial time. Moreover, NP-complete is another equivalence class of the relation $\sim^{\mathsf{P}}$. In summary, under the assumption that $\mathsf{P} \neq \mathsf{NP}$, the class NP-complete captures the desired notion of the most difficult problems in NP.

Since the relation $\leqslant_m^{\mathsf{P}}$ is transitive, in order to show that some $\varphi \in \mathsf{NP}$ is NP-complete, it suffices to find an NP-complete problem that polynomial time reduces to $\varphi$. The first problem to be shown to be NP-complete was that of deciding whether a Boolean formula is satisfiable ([19]).

Using this result, many other problems were shown to also be in this class in [28]; furthermore, there is a whole chapter in [24] devoted to providing an extensive list of NP-complete problems.

The framework hitherto described can be extended to counting problems. A *counting problem* is a function $f : \Sigma^* \to \mathbb{N}$ mapping (encodings of) problem instances to natural numbers. The class FP (read, "function polynomial time") is defined as that of counting problems that can be solved by a polynomial time DTM. In other words, $f \in \mathsf{FP}$ if and only if there are some polynomial $p$ and a DTM $M$ such that for every $n \in \mathbb{N}$, $M$ halts on every input of size $n$ in time $O(p(n))$ and $M(x) = f(x)\ \forall x \in \Sigma^*$. A counting problem is said to belong to the complexity class #P (read "sharp P") if there exist a polynomial time DTM $M$ and a polynomial $p$, so that for every input $x \in \Sigma^*$ we have $f(x) = |\{y \in \Sigma^* : |y| \leqslant p(|x|) \text{ and } M(\langle x, y\rangle) = 1\}|$.

It is worth pointing out that the previous expression for a function in #P states that $f(x)$ is the number of witnesses for $x$, but these are not required to be listed. We shall refer to problems that involve listing all objects satisfying a certain property as *enumeration problems*[7].

Going back to our discussion regarding counting problems, we point out the following properties of functions in #P.

PROPERTY 2.6. Let $n \in \mathbb{N} \setminus \{0\}$, $f, g \in \#\mathsf{P}$.

(i) $n \cdot f \in \#\mathsf{P}$, where $(n \cdot f)(x) = n \cdot f(x)\ \forall x \in \Sigma^*$.

(ii) $f + g \in \#\mathsf{P}$, where $(f + g)(x) = f(x) + g(x)\ \forall x \in \Sigma^*$.

(iii) $f \cdot g \in \#\mathsf{P}$, where $(f \cdot g)(x) = f(x) \cdot g(x)\ \forall x \in \Sigma^*$.

Note that FP and #P are the counting analogues to P and NP, respectively. In particular, the question of whether $\mathsf{FP} = \#\mathsf{P}$ is unresolved; in what follows, we assume $\mathsf{FP} \neq \#\mathsf{P}$.[8] It is also natural to seek to define the notion of #P-completeness. To do so, a new type of reduction need be introduced.

An *oracle* to a function $f$ over $\Sigma^*$ is an addition to a DTM that allows it to compute $f(x)$, $x \in \Sigma^*$, in just one computational step. A Turing machine with an oracle to $f$ is an augmentation of a Turing machine so that it is allowed to make calls to an oracle to $f$.[9] Given two functions $f, g : \Sigma^* \to \mathbb{N}$, we say $g$ *polynomial time Turing reduces to* $f$, and write $g \leqslant_T f$, if there is a polynomial time Turing machine with an oracle to $f$ that computes $g$. Note that $\leqslant_T$ is a transitive relation and FP is closed under $\leqslant_T$. A counting problem $f$ is #P-complete if and only if $f \in \#\mathsf{P}$ and any other $g \in \#\mathsf{P}$ polynomial time Turing reduces to $f$. Analogously to NP-complete problems, if $\mathsf{FP} \neq \#\mathsf{P}$, no #P-complete counting problem can be solved in polynomial time; thus, analogously to NP-complete, the class of #P-complete problems contains the most difficult problems within #P.

[7] In [24], the term "enumeration problem" is used to describe what we have here called counting problems.

[8] This assumption is redundant, as $\mathsf{P} \neq \mathsf{NP}$ already implies $\mathsf{FP} \neq \#\mathsf{P}$, but added for the sake of clarity. Be that as it may, it is interesting that it may be the case that $\mathsf{P} = \mathsf{NP}$ but not all counting problems can be solved in polynomial time, i.e. $\mathsf{FP} \neq \#\mathsf{P}$ (see [27]).

[9] In more informal terms, we think of an oracle as a subroutine added to an algorithm, the computation time of which is discounted from the total computation time of the algorithm ([27]).

There are relatively few problems known to be in FP. For our purposes, it is interesting to point out that, as shown in [27], the number of perfect matchings in a planar graph can be computed in polynomial time, while the problem of counting the number of *all* matchings in such a graph is #P-complete.

Finally for this section, we will introduce an important distinction within NP-complete problems. Namely, we will focus on how NP-completeness behaves when applied to problems the instances of which contain (integer) numbers. In order to study these problems, we introduce a new encoding independent function $\mathrm{Max} : D_\varphi \to \mathbb{N}$ intended to map instances of a decision problem $\varphi$ to the "magnitude" of the largest integer in the instance. In some cases, the numbers that may appear in an instance $I \in D_\varphi$ are "small" in terms of the size of the instance, in the sense that there is a polynomial $p$ such that $\mathrm{Max}(I) \leqslant p(\mathrm{Size}(I)) \ \forall I \in D_\varphi$. We say a decision problem is a *number problem* if there is no polynomial $p$ with this property.

Example 2.3. The problem PARTITION, which takes a list of positive integers $w_1, \ldots, w_n$ as input, and the answer to which is 1 if there is a subset $S \subseteq N = \{1, \ldots, n\}$ such that $\sum_{i \in S} w_i = \sum_{i \in N \setminus S} w_i$, and 0 otherwise, is a number problem. Indeed, for a given list of numbers forming an instance $I$ of the problem let $\mathrm{Max}(I) = W = \sum_{i \in N} w_i$. Assuming the numbers to be encoded in binary, each $w_i$ takes $\log_2(w_i)$ bits of space. Thus, we may take $\mathrm{Size}(I) = \sum_{i=1}^{n} \log_2(w_i)$. There is no polynomial function of $\mathrm{Size}(I)$ that bounds $\mathrm{Max}(I)$ above.

Example 2.4. Consider the problem of deciding whether an $n$-vertex graph contains a clique of size larger than $k$ is not a number problem. We previously mentioned that, for problems whose instances $I$ are graphs, it is commonplace to take $\mathrm{Size}(I) = n$, the number of vertices of the input graph. On the other hand, the only number appearing in an instance of this problem is $k$, so we take $\mathrm{Max}(I) = k$. For the question to be meaningful, it must be the case that $k \leqslant n$; in particular, the problem at issue is not a number problem.

We say a problem $\varphi$ can be solved in *pseudopolynomial time* when there is a DTM $M$ that solves it, and, for every input instance $I$ of the problem, halts in time $O(p(\mathrm{Size}(I), \mathrm{Max}(I)))$, where $p$ is a polynomial. Note that if $\varphi$ is not a number problem, there is no distinction between polynomial time and pseudopolynomial time solvability. Now, given a decision problem $\varphi$ and a polynomial $p$, let $\varphi_p$ be the restriction of $\varphi$ to instances such that $\mathrm{Max}(I) \leqslant p(\mathrm{Size}(I))$. By definition, $\varphi_p$ is never a number problem. We say $\varphi$ is NP-complete *in the strong sense* (or *strongly* NP-complete) if there is some polynomial $p$ such that $\varphi_p$ is NP-complete. If a number problem $\varphi$ is NP-complete but not strongly NP-complete (or, at least, not known to be strongly NP-complete), we say $\varphi$ is NP-complete *in the weak sense* (or *weakly* NP-complete). The notions of weak and strong completeness can also be extended to counting problems. Indeed, the previous discussion still applies if one changes NP for #P.

The problems in Example 2.3 and Example 2.4 are weakly NP-complete and strongly NP-complete, respectively. Their associated counting problems, i.e. those of counting the solutions

to an instance of PARTITION and the number of cliques in a graph of size larger than $k$, are weakly #P-complete and strongly #P-complete, respectively. It is worth noting that under the assumption $\mathsf{P} \neq \mathsf{NP}$ (respectively, $\mathsf{FP} \neq \#\mathsf{P}$), pseudopolynomial time algorithms for a weakly NP-complete (respectively, #P-complete) problem may exist, but are not guaranteed to. Be that as it may, [24] provides a pseudopolynomial time algorithm that solves PARTITION. The associated counting problem can also be solved in pseudopolynomial time: it reduces to the computation of the Shapley-Shubik index of a weighted voting game ([24]), which is discussed in the following subsection.

### 2.3 Power indices in games with restricted cooperation

A *TU game* (short for cooperative transferable utility game) is a pair $\Gamma = (N, v)$ where $N$ is a finite set and $v$ is a function taking real values over the elements of $2^N = \{S : S \subseteq N\}$, with the only constraint being $v(\varnothing) = 0$. In this context, the elements of $N$ are called *players* and a set of players is called a *coalition*; moreover, we refer to $v$ as the *characteristic function* of the game. Unless otherwise stated, we will always take $N = \{1, \ldots, n\}$ for some positive integer $n$. Moreover, as long as there is no confusion regarding the player set, we will identify a TU game with its characteristic function.

We say a game $v$ is *simple* when $v(S) \in \{0, 1\}\ \forall S \subseteq N$, $v(N) = 1$, and $v$ is *monotonic*, that is, for any pair of coalitions $S, T \subseteq N$ such that $S \subseteq T$, we have $v(S) \leqslant v(T)$. In a simple game, a coalition $S$ for which $v(S) = 1$ is called *winning*; if $v(S) = 0$, we say $S$ is a *losing* coalition.

A simple game can thus be defined via its set of winning coalitions, $\mathcal{W}(v) = \{S \subseteq N : v(S) = 1\}$. We say a coalition $S \subseteq N \setminus \{i\}$ is a *swing* for player $i \in N$ when $S$ is losing and $S \cup \{i\}$ is winning. The number of swings for player $i \in N$ in the game is denoted by $\eta_i(v)$. A coalition $W \in \mathcal{W}(v)$ is said to be a *minimal winning coalition* when any coalition $T \subsetneq W$ is losing. The set of minimal winning coalitions is denoted by $\mathcal{W}^m(v)$.

A *weighted voting* game is a simple game for which there exists a list of (non-negative integer) *weights* $w_1, \ldots, w_n$, and a (positive integer) *quota* $q$ such that

$$v(S) = \begin{cases} 1 \text{ if } w(S) \geqslant q \\ 0 \text{ otherwise} \end{cases}$$

where for $S \subseteq N$, $w(S) = \sum_{i \in S} w_i$. If a game admits such a representation, we will compactly write $\Gamma = [q;\ w_1, \ldots, w_n]$.[10]

A *value* on a class $\mathcal{C}$ of TU games is a function $f$ that maps each characteristic function $v$ of a game in $\mathcal{C}$ to an $n$-dimensional real vector, $f(v)$. We refer to the $i$-th component of this vector as the value of player $i$ in $v$ according to $f$. In what follows we will focus mostly on the *Banzhaf* value and, to a lesser extent, the *Shapley* value, which are defined for the whole class of TU games. The Banzhaf value of a player in $v$ is defined by

[10] Nonetheless, it is still common to identify a weighted voting game with the characteristic function it induces, as defined above.

$$\beta_i(v) = 2^{1-n} \sum_{S \subseteq N} (v(S \cup \{i\}) - v(S))$$

If $v$ is a simple game, we will refer to $\beta_i(v)$ as the Banzhaf *index* of $i$. Note that we can define the Banzhaf index in terms of the number of swings for each player; indeed, if $v$ is a simple game, $\beta_i(v) = 2^{1-n}\eta_i(v)$. The Shapley value of $i$ is defined by the formula

$$\varphi_i(v) = \frac{1}{n!} \sum_{S \subseteq N} \gamma_{s,n} \cdot (v(S \cup \{i\}) - v(S))$$

where $s = |S|$ and $\gamma_{s,n} = s!(n-s-1)!$. If $v$ is a simple game, the value is called the Shapley-Shubik index ([46]), and the expression above can be rewritten as

$$\varphi_i(v) = \frac{1}{n!} \sum_{s=0}^{n-1} \gamma_{s,n} \cdot \eta_i^s(v)$$

where $\eta_i^s(v)$ denotes the number of swings $S$ for player $i$ in $v$ such that $|S| = s$.

Several authors have studied the complexity of the computation of these indices and other related problems; we summarize the known results below.

PROPERTY 2.7. Given a weighted voting game $\Gamma = [q;\ w_1, \ldots, w_n]$ with characteristic function $v$ and a distinguished player $i \in N$ as inputs:

(i) The problem of deciding whether $\eta_i(v) > 0$ is weakly NP-complete. The problem of computing $\eta_i(v)$ exactly is weakly #P-complete ([41]).

(ii) The problem of deciding whether $\beta_i(v) > 0$ is weakly NP-complete. The problem of computing $\beta_i(v)$ exactly is weakly #P-complete ([41]).[11]

(iii) Given also $s \in \{0, \ldots, n-1\}$, the problem of deciding whether $\eta_i^s(v) > 0$ is weakly NP-complete. The problem of computing $\eta_i^s(v)$ exactly is weakly #P-complete ([24]).

(iv) The problem of deciding whether $\varphi_i(v) > 0$ is weakly NP-complete. The problem of computing $\varphi_i(v)$ exactly is weakly #P-complete ([22, 24]).

From now on, we will refer to the problem of computing $\eta_i(v)$ ($\eta_i^s(v)$) exactly as counting swings (of size $s$); analogously, we will refer to the problem of deciding whether $\eta_i(v)$ ($\eta_i^s(v)$) is non-zero as deciding whether there exists a swing (of size $s$).

In Property 2.7 above, the inputs $q$, $w_1$, ..., $w_n$, $i$, are assumed to be numbers encoded in binary. Throughout this work, weighted voting games will always be encoded in this manner.

[11] Note that $\beta_i(v)$ is a rational number, while the class #P only contains functions that take values in $\mathbb{N}$. However, the Banzhaf index only differs from a problem in #P (the computation of the number of swings) by a factor of a power of 2. Multiplying by such a number can be done in linear time, hence why we will abuse notation and say that the exact computation of the Banzhaf index is in #P. This is loosely acknowledged in [41], although the class #P is not properly defined there. The situation is similar for the Shapley-Shubik index. In this case, as argued in [22], the computation of $n!\varphi_i(v)$ is in #P; the factorial of $n$ can be computed in polynomial time, and so can a division by this number.

As such, we may take $\text{Max}(I) = W = \sum_{i=1}^{n} w_i$ as the magnitude of an instance, and $\text{Size}(I) = n \log_2 W$. Note that an algorithm to solve the problems at issue runs in pseudopolynomial time if and only if it halts in $O(p(n, W))$ steps for some polynomial $p$.

Now, the known results establish completeness in the weak sense in all cases, so there is hope for pseudopolynomial time algorithms to exist for the problems hereby introduced. In fact, such algorithms exist for all of them: [31] and [11] present pseudopolynomial time algorithms to compute the Banzhaf and Shapley-Shubik indices, based on dynamic programming and generating functions, respectively.

The aforementioned algorithms can also be applied to solve the other problems in Property 2.7 as well. In particular, counting swings and computing the Banzhaf index are polynomially related problems: recall that they only differ by a factor of a power of 2, and such operation can be performed efficiently. As such, other references tend to assess the complexity of the computation of the Banzhaf index via that of counting swings; we will follow this approach in subsequent sections. The relation between the Shapley-Shubik index and counting swings of a particular size is not as symmetric. Indeed, while the formula given for the latter is a linear combination of the number of swings for each size, knowing the Shapley-Shubik index of a player does not provide any information on the number of swings for this player of any size.

In the values defined so far, it is assumed that all players communicate without constraints, that is, that all coalitions can form with equal probability. In reality, this is often far from the case; this gives rise to several models of restrictions to cooperation. We focus on two graph based models, which, for the sake of simplicity, we will only apply to simple games; for the remainder of this section, let $G = (N, E)$ be a graph.

Given a simple game $v$ with player set $N$, consider the game $v^A$ defined by

$$v^A(S) = \begin{cases} 1 \text{ if } \exists T \subseteq S : T \in \mathcal{W}(v) \text{ and } G[T] \text{ is connected} \\ 0 \text{ otherwise} \end{cases}$$

for all $S \subseteq N$, where $G[T]$ denotes the subgraph of $G$ induced by the players in coalition $T$, as defined in Subsection 2.1. We say $v^A$ is a *communication game*, and refer to $G$ as its communication graph; more explicitly, we may say $v^A$ is $v$ with cooperation restricted by communication graph $G$. It can be shown that $v^A$ is either a simple game or the null game[12].

[12]The *null game* (over a set of players $N$) has characteristic function $v(S) = 0 \ \forall S \subseteq N$.

The complexity of computing the Banzhaf and Shapley-Shubik indices when the original game is a weighted voting game has since been studied in [9, 47]. These references provide pseudopolynomial time algorithms to compute these power indices long as the communication graph $G$ is within some particular class of graphs. It is also shown in [9] that, for generic communication graphs, the problem of computing these indices is strongly #P-complete.

In the communication games model, we have seen that players cooperate when they are adjacent in $G$. The following approach is based on the opposite idea, but it is not fully complementary. Given a simple game $v$ with player set $N$, consider the game with characteristic function $v^I$ defined[13] by

$$v^I(S) = \begin{cases} 1 \text{ if } \exists T \subseteq S : \ T \in \mathcal{W}(v) \text{ and } G[T] \text{ is empty} \\ 0 \text{ otherwise} \end{cases}$$

for all $S \subseteq N$. Analogously to communication games, we say $v^I$ is an *incompatibility game*, refer to $G$ as its incompatibility graph; alternatively, more explicitly, we may say $v^I$ is $v$ with cooperation restricted by $G$ as an incompatibility graph. Again, it can be shown that $v^I$ is either a simple game or the null game. The algorithms in [4, 35] compute the Banzhaf and Shapley-Shubik indices in incompatibility games provided that the original game is a weighted voting game.

[13] This is not the definition used in [4], but it is mentioned there that the definition presented here is better so as to preserve behavior over the class of simple games, hence why we use it.

## 3. Hardness on graphs with very few maximal independent sets

In this section we will discuss the complexity of the algorithm introduced in [4] to compute the Banzhaf index in weighted voting games with cooperation restricted by an incompatibility graph. Henceforth, as long as no confusion arises, we shall refer to these as restricted games. Moreover, if $G$ is the incompatibility graph associated with such a game, we will call it "the $G$ restricted game". Similarly, in order to refer to restricted games where the incompatibility graph is of a certain class $\mathcal{C}$, we will use the terminology "$\mathcal{C}$ restricted games".

It is also important to determine how we will encode instances of the problems we study. An instance of a restricted game is formed by a weighted voting game and a graph. In Section 2, we discussed how to encode a weighted voting game. In particular, we established its magnitude to be $W$, the sum of the weights of its $n$ players, and its size to be $n \log_2 W$. We also mentioned that the size of an $n$-vertex graph input is taken as $n$. Since $n$ is bounded above by $n \log_2 W$, we take the size and magnitude of an instance of a restricted game to be the same as for unrestricted voting games.

Having made these clarifications, we proceed with our analysis of the algorithm. The algorithm at issue uses *generating functions*, which are a way to represent sequences of numbers. Namely, given a sequence $\{a(n)\}_{n\in\mathbb{N}}$, we say the power series $f(x) = \sum\limits_{n\in\mathbb{N}} a(n)x^n$ is the generating function of the sequence. The concept can be extended to sequences with multiple indices; thus,

$$f(x_1, \ldots, x_m) = \sum_{n_1\in\mathbb{N}} \cdots \sum_{n_m\in\mathbb{N}} a(n_1, \ldots, n_m) x_1^{n_1} \cdots x_m^{n_m}$$

is the generating function that represents the sequence $\{a(n_1, \ldots, n_m) : n_1 \in \mathbb{N}, \ldots, n_m \in \mathbb{N}\}$.

The complexity of algorithms that use generating functions is usually assessed via the time needed to compute the generating function(s) of interest. For the sake of completeness, we summarize below the results presented as Propositions 9 through 11 in [4], regarding the algorithm we seek to study.

PROPOSITION 3.1. Let $\Gamma$ be the weighted voting game given by $[q;\ w_1, \ldots, w_n]$, and $G = (V, E)$ an incompatibility graph. Let $v^I$ be the characteristic function of the $G$-restriction of $\Gamma$, denote by $\mathcal{M} = \{M_1, \ldots, M_\mu\}$ the family of maximal independent sets in $G$, and let $K = \{1, \ldots, \mu\}$. Then, for each $i \in N = \{1, \ldots, n\}$,

(i) The number of swings for player $i$ is given by

$$\eta_i(v^I) = \sum_{l\in K_i} \sum_{r_l=q-w_i}^{q-1} \sum_{\substack{m<l \\ i\in M_m}} \sum_{r_m=0}^{q-w_i-1} \sum_{\substack{m'<l \\ i\notin M_{m'}}} \sum_{r_{m'}=0}^{q-1} \sum_{j>l} \sum_{r_j=0}^{q-1} A^i(r_1, \ldots, r_\mu)$$

where $K_i = \{l \in K : i \in M_l\}$ and $A^i(r_1, \ldots, r_\mu)$ is the number of coalitions $S \subseteq N \setminus \{i\}$ such that $w(S \cap M_l) = \sum\limits_{j\in S\cap M_l} w_j = r_l\ \forall l \in K$.

(ii) The generating functions for the numbers $A^i(r_1, \ldots, r_\mu)$, $r_1, \ldots, r_\mu \geqslant 0$ are given by

$$BC_i(x_1, \ldots, x_\mu) = \prod_{j \neq i} \left(1 + \prod_{l \in K_j} x_l^{w_j}\right)$$

(iii) The number of non-zero terms of $BC_i(x_1, \ldots, x_\mu)$ is bounded above by $\prod_{l=1}^{\mu} (w(M_l) + 1)$.

(iv) A time $O(nW^\mu)$, where $W = w(N)$, is required to expand the polynomial $BC_i(x_1, \ldots, x_\mu)$.

We will focus on two aspects of these results. First, the inputs required to compute the generating functions $BC_i(x_1, \ldots, x_\mu)$; then, the nature of the asymptotic bound on the number of operations needed to expand this function given in (iv). These issues are both related to the number of maximal independent sets in the incompatibility graph.

Besides the original weighted voting game itself and the incompatibility graph $G$, Proposition 3.1 assumes the collection of maximal independent sets in $G$ to be given as well. As such, since we only take the restricted game itself as the input, computing this collection is required before computing the generating function. This is acknowledged in Remark 15 in [4], where it is also stated that "there are efficient algorithms to [enumerate the maximal independent sets in a graph]", and the reader is referred to [1, 13, 49] .

However, enumeration algorithms such as these are "output-sensitive". In this case, this means that their complexity is given in terms of the time per maximal independent set needed to list all such sets. In particular, they may be polynomial time algorithms only if $G$ has a polynomial amount of maximal independent sets in terms of the number of vertices of the graph. In this sense, there are graphs that have exponentially many maximal independent sets.

Example 3.1. Let $k \geqslant 1$, $G$ be the disjoint union of $k$ copies of the complete graph $K_3$ (see Figure 1). Thus, $G$ has $n = 3k$ vertices. Note that a set of vertices of $G$ is a maximal independent set if and only if it contains exactly one vertex from each copy of $K_3$. Indeed, vertices from different copies of $K_3$ are not adjacent, so any maximal independent set must contain at least one vertex from each copy; on the other hand, two vertices in the same copy of $K_3$ are adjacent, and so cannot be in a same independent set. Hence, $G$ has $3^k = 3^{\frac{n}{3}}$ maximal independent sets.

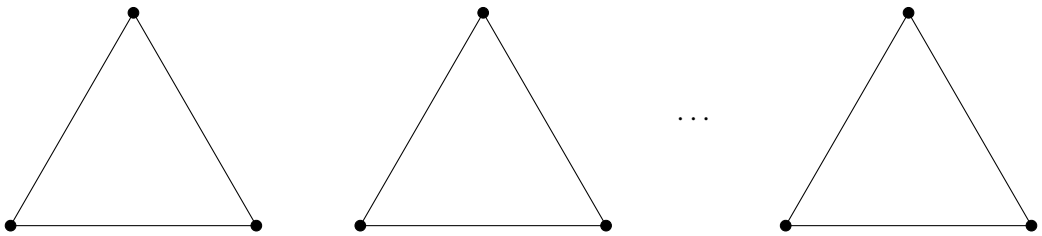


Figure 1: Illustration of the graph described in Example 3.1.

It can be shown that if $n$ is a multiple of 3, $G$ is the $n$-vertex graph with the maximum number of maximal independent sets. See [33] for details, including the construction of graphs

that maximize the number of maximal independent sets with respect to their number of vertices $n$ when $n$ is not a multiple of 3.

Definition 3.1. Given a graph $G$, let $\mu(G)$ be the number of maximal independent sets in $G$. Let $\mathcal{C}$ be a class of graphs. We say $\mathcal{C}$ has *few* maximal independent sets if and only if there exists a polynomial $p$ such that for every $n$-vertex graph $G_n$ of class $\mathcal{C}$, $\mu(G_n) \leqslant p(n)$ ([42]).

We say $\mathcal{C}$ has *very few* maximal independent sets if and only if there exists a constant $k$ such that no graph of class $\mathcal{C}$ has more than $k$ maximal independent sets, i.e. $\mu(G) \leqslant k \ \forall G \in \mathcal{C}$.

We will say a graph has (very) few maximal independent sets when it belongs to a class that has (very) few maximal independent sets.

In a slight abuse of language, we will refer to a restricted game such that its incompatibility graph has (very) few maximal independent sets as a "restricted game with (very) few maximal independent sets".

Now, even though Example 3.1 shows that an $n$-vertex graph may have a number of maximal independent sets that is not bounded by any polynomial function of $n$, there are wide classes of graphs which have few maximal independent sets. These include, for instance, co-planar graphs ([42]). See [43] for more examples of such classes of graphs, along with some problems that are hard on general graphs, but are polynomial time solvable when the input is restricted to having few maximal independent sets.

In particular, an algorithm like the one described in [50] (which is a variation on those in [1] and [13]) lists the maximal independent sets in graphs that have few maximal independent sets in polynomial time, since it takes time $O(n^3)$ per output[14]. Consider also the result below, which appears as Proposition 6 in [4].

Lemma 3.1. Let $v$ be a simple game with player set $N$, $G = (N, E)$ be a graph, and let $v^I$ be the $G$-restriction of $v$. Suppose there is some minimal winning coalition $W \in \mathcal{W}^m(v)$ such that no two players in $W$ are adjacent as vertices of $G$.[15] Then, the set of minimal winning coalitions of the restricted game is precisely the collection of minimal winning coalitions of $v$ with this property, that is, $\mathcal{W}^m(v^I) = \{W \in \mathcal{W}^m(v) : \{i, j\} \notin E \ \forall i, j \in W\}$.

Remark 3.1. It is interesting to point out that the analogue of Lemma 3.1 for communication games does not hold in general. Indeed, one can find examples of communication games such that no minimal winning coalition of it is a minimal winning coalition of the original game, while none of the latter coalitions is a winning coalition in the communication game. In essence, this is because if $S$ is an independent set in a graph, then so is every subset of $S$; in general, this is not the case if $S$ is connected. See [15] for details.

In plain words, Lemma 3.1 states that the minimal winning coalitions of a $G$ restricted game are exactly those minimal winning coalitions of the original game such that, as sets of vertices of the incompatibility graph $G$, are independent sets. For our purposes, this is relevant because, combined with the discussion on the enumeration of the maximal independent sets in graphs with few such sets, yields the following result.

[14] There are more efficient algorithms to list the maximal independent sets of a graph. The algorithm in [49] is shown to take time $O(3^{\frac{n}{3}})$ in the worst-case.

[15] Otherwise, $v^I$ is the null game.

PROPOSITION 3.2. Let $\Gamma = [q;\, w_1, \ldots, w_n]$ be a weighted voting game, $G$ be a graph with few maximal independent sets. Consider the $G$-restriction of $\Gamma$, and suppose the hypotheses of Lemma 3.1 are satisfied. Given a player $i \in N$, the problem of deciding whether there is a swing $S \subseteq N \setminus \{i\}$ for $i$ is in $\mathsf{NP}$. Furthermore, the associated counting problem (counting swings in a restricted game) is in $\#\mathsf{P}$.

*Proof.* It will suffice to show that, given an appropriate witness, we can check whether it encodes a swing for $i$ in polynomial time in its length. We will see that the list of weights of the players in a candidate swing $S \subseteq N$ provides a suitable witness. The length of this list is bounded by $n \log_2 W$, where $W = \sum_{j \in N} w_j$, which is the size of the full instance. Moreover, we can verify in linear time whether $i \notin S$, as well as whether the given weights are indeed those of the players in $S$. All in all, it only remains to be shown that we can decide whether $S \subseteq N \setminus \{i\}$ is a swing for $i$ in polynomial time with respect to $n \log_2 W$.

To this end, let $\mathcal{M} = \{M_1, \ldots, M_\mu\}$ be the family of maximal independent sets in $G$, $\mathcal{M}_i \subseteq \mathcal{M}$ be the collection of those that contain a given player $i \in N$. By Lemma 3.1, every minimal winning coalition of the restricted game is a subset of some (not necessarily unique) maximal independent set in $G$. Moreover, by definition, every winning coalition is a superset of some (not necessarily unique) minimal winning coalition. Thus, $S \subseteq N$ is winning in the restricted game if and only if there is some (not necessarily unique) $M \in \mathcal{M}$ such that $w(S \cap M) \geqslant q$. Furthermore, $S \subseteq N \setminus \{i\}$ is a swing for $i$ in the restricted game if and only if there is some (not necessarily unique) $M \in \mathcal{M}_i$ such that $q - w_i \leqslant w(S \cap M) \leqslant q - 1$, while $w(S \cap M') \leqslant q - 1 \; \forall M' \neq M$.

Per the discussion preceding Lemma 3.1, we can construct $\mathcal{M}$ in polynomial time. Now, given $S \subseteq N \setminus \{i\}$ consider an auxiliary boolean variable, call it $z$, to record whether some $M \in \mathcal{M}_i$ such that $q - w_i \leqslant w(S \cap M) \leqslant q - 1$ has been found; initially set $z$ to "False", and apply the following procedure. For each $l \in K$, check whether $i \in M_l$, compute $w(S \cap M_l)$ and check if it exceeds $q - 1$. If so, stop and conclude that $S$ is not a swing for $i$ (because in this case $S$ would be winning).

Otherwise, if $i \in M_l$ and $z$ is "False", check whether $q - w_i \leqslant w(S \cap M_l)$; if so, update $z$ to "True". If all maximal independent sets are processed and $z$ is set to "True" when this occurs, we conclude that $S$ is a swing for $i$; otherwise, conclude that it is not.

Since the maximal independent sets in $G$ cannot have more than $n$ elements, for each $l \in K$, we can check whether $i \in M_l$ and compute $S \cap M_l$ in linear time. Moreover, operating bit-wise as usual, the weights of the players in $S \cap M_l$ can be added in time $O(n \log_2 W)$, and the required comparisons of integer numbers take $O(\log_2 W)$ time each, and there is a constant amount of them. All in all, processing each maximal independent set takes $O(n \log_2 W)$ time. By hypothesis, the number of maximal independent sets in $G$ is bounded by a polinomial function of $n$, call it $p$, so the whole procedure takes $O(p(n) n \log_2 W)$ time. □

COROLLARY 3.1. Let $\Gamma$ and $G$ be as in the proposition. Given a player $i \in N$, the problem of deciding whether there is a swing $S \subseteq N \setminus \{i\}$ for $i$ of a particular size $s$ is in NP. The associated counting problem (counting swings of size $s$ in a restricted game) is in #P.

*Proof.* It suffices to take the same witness as in the proposition and check if $|S| = s$ (which takes linear time) before running the procedure described in its proof. If this test fails, $S$ does not satisfy all of the required properties; otherwise, the algorithm in the proposition will yield the appropriate conclusion. □

The lemma shows that the problem of computing the Banzhaf index in these restricted games is in #P; combined with Property 2.6, the corollary shows that the computation of the Shapley-Shubik index is also in #P. Furthermore, deciding whether the Banzhaf or Shapley-Shubik indices are non-zero for a given player are NP problems.

In general, if the incompatibility graph does not have few maximal independent sets, these results are unlikely to hold. If they did, it would be implied that we can decide in polynomial time whether a property holds over the maximal independent sets in a graph, even when the amount of these is not bounded by any polynomial function of the size of the input instance.

REMARK 3.2. In contrast with Remark 3.1, Proposition 3.2 and Corollary 3.1 can be slightly altered to yield analogous results on communication games. This is because a minimal winning coalition of a communication game must induce a connected subgraph of the communication graph. Thus, it is the case that every minimal winning coalition of the communication game is a subset of some (not necessarily unique) connected component of $G$. From this point onward, replacing "maximal independent sets" in Proposition 3.2 by "connected components", we obtain a procedure to check whether $S$ is a swing for $i$ in the communication game with communication graph $G$.

A graph may have at most $n$ connected components, and there are polynomial time (in fact, linear time) algorithms to find them all via breadth or depth-first search (see [20], Chapters 22.2 and 22.3). In other words, in contrast to what we have seen for incompatibility games, the problems of computing the Banzhaf and Shapley-Shubik indices in communication games are both in #P (and the problems of deciding whether the indices are non-zero are in NP) regardless of the communication graph.

Within the discussion of the complexity of the algorithm in [4], so far we have seen that, in general, the calculations needed prior to the computation of the generating function introduced in Proposition 3.1 cannot be done efficiently. As for the computation of the generating function itself, this result states that the number of operations needed to do so is $O(nW^{\mu})$, where $\mu$ is the number of maximal independent sets of the incompatibility graph. Thus, if $\mu = O(1)$, that is, if the incompatibility graph has very few maximal independent sets, the algorithm counts swings in pseudopolynomial time.

All in all, the algorithm provides an upper bound on the complexity of the computation at issue for graphs that have very few maximal independent sets. The following results introduce

two classes of graphs with very few maximal independent sets for which the algorithm is optimal (or, at least, it cannot be strengthened to a polynomial time algorithm).

Definition 3.2. A *double star* is a graph $G = (V, E)$ such that $V$ can be partitioned in two disjoint independent sets $A = \{1, \ldots, n\}$, $B = \{n+1, \ldots, n'\}$ such that vertex 1 (respectively, vertex $n+1$) is adjacent to every vertex in $B$ (respectively, $A$). See Figure 2.

Note that, regardless of its total number of vertices, a double star has exactly three maximal independent sets. Indeed, following the notation introduced in Definition 3.2, sets $A$ and $B$ are in fact maximal independent sets. By construction, no independent set $S$ in a double star such that $S \cap A \neq \varnothing$ and $S \cap B \neq \varnothing$ can contain vertices 1 or $n+1$. It follows that $A^* \cup B^*$, where $A^* = A \setminus \{1\}$ and $B^* = B \setminus \{n+1\}$, is the third and final maximal independent set in a double star. This proves the following result.

Lemma 3.2. The class of double stars has very few maximal independent sets.

Lemma 3.3. Let $\Gamma = [q;\ w_1, \ldots, w_n]$ be a weighted voting game, and consider the $n'$-player voting game $\Gamma' = [q;\ w_1, \ldots, w_n, q, \ldots, q]$, where $n' > n$. In $\Gamma'$, the first $n$ players are those of $\Gamma$, and have the same weights as in that game; the remaining $n' - n$ players all have weight $q$. Let $G$ be a double star with $A = \{1, \ldots, n\}$, $B = \{n+1, \ldots, n'\}$ (see Figure 2). A coalition $S \subseteq (A \cup B) \setminus \{i\}$ is a swing for player $i \in A$ in the $G$-restriction of $\Gamma'$ if and only if it is so in the unrestricted game $\Gamma$.

*Proof.* Suppose $i \neq 1$. In such case, from our previous discussion on maximal independent sets in double stars, $i$ is contained in maximal independent sets $A$ and $A^* \cup B^*$. Thus, by the characterization of swings given in the proof of Proposition 3.2, $S \subseteq (A \cup B) \setminus \{i\}$ is a swing for player $i$ if and only if $w(S \cap M) \leqslant q-1\ \forall M \in \{A,\ B,\ A^* \cup B^*\}$ and either $q - w_i \leqslant w(S \cap A)$ or $q - w_i \leqslant w\left(S \cap (A^* \cup B^*)\right)$ (or possibly both, a priori). If $i = 1$, the only maximal independent set that contains player 1 is $A$. The same result thus yields that $S \subseteq (A \cup B) \setminus \{1\}$ is a swing for player 1 in the restricted game if and only if $q - w_1 \leqslant w(S \cap A) \leqslant q - 1$, $w(S \cap B) \leqslant q - 1$ and $w\left(S \cap (A^* \cup B^*)\right) \leqslant q - 1$.

However, since all players of $\Gamma'$ that are in $B$ have weight $q$, for any player $i \in A$ the appropriate set of inequalities may only hold if $S \cap B = \varnothing$, equivalently, $S \subseteq A \setminus \{i\}$. All in all, we obtain that $S \subseteq (A \cup B) \setminus \{i\}$ is a swing for player $i$ in the restricted game if and only if $S \subseteq A \setminus \{i\}$ and $q - w_i \leqslant w(S) \leqslant q - 1$. This is precisely what characterizes the swings for player $i$ in the unrestricted game $\Gamma$, and the result follows. □

Proposition 3.3. Counting swings in a double star restricted game is a weakly #P-complete problem.

*Proof.* The problem is in #P as a consequence of Lemma 3.2 and Proposition 3.2. Thus, we only need to show that the problem is weakly #P-hard.

To do so, recall that Property 2.7 establishes weak #P-completeness for counting swings in unrestricted weighted voting games. Let $\Gamma = [q;\ w_1, \ldots, w_n]$ be one such game, and consider

a double star graph $G$ with $A = \{1, \ldots, n\}$ and $B = \{n+1\}$. Let $\Gamma'$ be the $(n+1)$-player weighted voting game $\Gamma' = [q;\ w_1, \ldots, w_n, q]$, and consider the $G$-restriction of $\Gamma'$. Lemma 3.3 provides a one-to-one correspondence between swings for a player in the unrestricted game and swings for that same player in the restricted game.

Moreover, the restricted game can be constructed from the original game in polynomial time with respect to the latter's size. Indeed, the restricted game has $n+1 = O(n)$ players, and so both the list of weights and the incompatibility graph can be determined in linear time.

In other words, Lemma 3.3 provides a polynomial time reduction from counting swings in an unrestricted game to this same problem in a double star restricted game. As a consequence, the latter problem is weakly #P-hard. □

Proposition 3.4. Deciding whether there exists a swing in a double star restricted game is a weakly NP-complete problem.

*Proof.* By Lemma 3.3, in particular, there exists a swing for a player in the restricted game if and only if there exists one in the unrestricted game. Thus, it suffices to replace any appearance of #P and "counting swings" in the proof of Proposition 3.3 by NP and "deciding whether there exists a swing", respectively. □

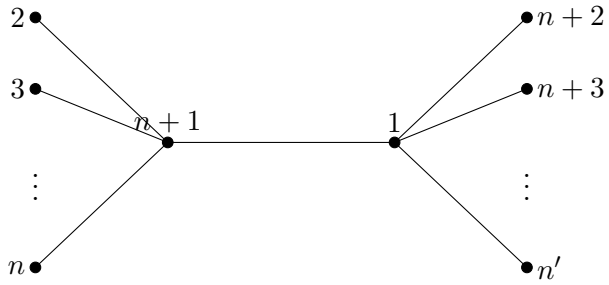



Figure 2: A double star, as defined in Definition 3.2.

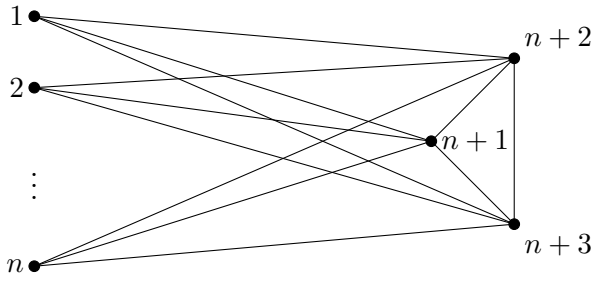



Figure 3: The complete $k$-partite graph used in Lemma 3.5 (for $k = 4$).

The other class of graphs that we will discuss is that of complete multipartite graphs. It follows from their definition that, for a fixed $k \geqslant 2$, any complete $k$-partite graph has precisely $k$ maximal independent sets, no matter how many vertices it has.

Lemma 3.4. Let $k$ be a fixed positive integer, $k \geqslant 2$. The class of complete $k$-partite graphs has very few maximal independent sets.

Lemma 3.5. Let $k$ be a fixed positive integer, $k \geqslant 2$. Let $\Gamma = [q;\ w_1, \ldots, w_n]$ be a weighted voting game, and consider the $(n+k-1)$-player weighted voting game $\Gamma' = [q;\ w_1, \ldots, w_n, q, \ldots, q]$. Let $G = (V, E)$ be the complete $k$-partite graph with maximal independent sets $V_1 = \{1, \ldots, n\}$, $V_2 = \{n+1\}, \ldots, V_k = \{n+k-1\}$ (see Figure 3). A coalition $S \subseteq V \setminus \{i\}$ is a swing for a player $i \in V_1$ in the $G$-restriction of $\Gamma'$ if and only if it is in the unrestricted game $\Gamma$.

*Proof.* The proof is similar to that of Lemma 3.3. By Proposition 3.2, $S \subseteq V \setminus \{i\}$ is a swing for $i \in V_1$ in the restricted game if and only if $q - w_i \leqslant w(S \cap V_1) \leqslant q - 1$ and $w(S \cap V_l) \leqslant q - 1$ for $2 \leqslant l \leqslant k$. Again, any coalition of players of $\Gamma'$ containing players outside of those in $V_1$ is winning; in particular, it is not a swing for any player. Thus, $S \subseteq V \setminus \{i\}$ is a swing for $i$ in the restricted game if and only if $S \subseteq V_1 \setminus \{i\}$ and $q - w_i \leqslant w(S) \leqslant q - 1$, that is, if $S$ is a swing for $i$ in the unrestricted game. □

Proposition 3.5. Let $k \geqslant 2$. Counting swings in a complete $k$-partite restricted game is weakly #P-complete.

*Proof.* Analogously to Proposition 3.3, we only need to show weak #P-hardness, since membership in #P follows from Lemma 3.4 and Proposition 3.2.

Let $\Gamma = [q;\ w_1, \ldots, w_n]$ be a weighted voting game, where $n \geqslant k$. Given such game, let $\Gamma'$ and $G$ be as in the statement of Lemma 3.5. This result establishes a one-to-one correspondence between swings for a player in $\Gamma$ to swings for that same player in the $G$-restriction of $\Gamma'$.

Moreover, by construction, the restricted game has $n + k - 1 = O(n)$ players, and so can be built in linear time. All in all, we have a polynomial time reduction from counting swings in an unrestricted game, a weakly #P-complete problem, to that same counting problem in a complete $k$-partite restricted game. In other words, it is shown that the latter problem is weakly #P-complete. □

Proposition 3.6. Deciding whether there exists a swing for a given player in a complete multipartite restricted game is weakly NP-complete.

*Proof.* Analogous to Proposition 3.4, using Lemma 3.5 and Proposition 3.5 instead of the results there referenced. □

Remark 3.3. The results we have provided for counting swings and the associated decision problem in the latter part of this section also apply to the corresponding problems involving the number of swings of a fixed size and the computation of the Banzhaf and Shapley-Shubik indices in restricted games. For the former, the same arguments used in Lemma 3.3 and Lemma 3.5 suffice to show weak #P-hardness (and NP-hardness) for the appropriate problems. For the Banzhaf index, recall from the discussion following Property 2.7 that it is polynomially related to counting swings, and so the two problems have the same complexity. Finally, for the Shapley-Shubik index, a separate reduction need be constructed, which we will not cover here. Details can be found in Lemma 3.1 of [9].

# 4. Hardness on graphs with few maximal independent sets

So far, we have shown that, if the incompatibility graph has very few maximal independent sets, the algorithm described in Proposition 3.1 computes the Banzhaf index in pseudopolynomial time. More specifically, the result states that the time complexity of the algorithm is $O(nW^{\mu})$, where $\mu$ denotes the number of maximal independent sets in the incompatibility graph. Note that, in fact, $nW^{\mu}$ is a polynomial in terms of $n$ and $W$ if and only if the incompatibility graph has very few maximal independent sets, i.e. $\mu = O(1)$. In particular, the algorithm runs in exponential time (with respect to the number of players) even when the incompatibility graph has few maximal independent sets.

However, by itself, this does not preclude the existence of a pseudopolynomial time algorithm to solve the problem in such cases, nor in general. With this in mind, we now proceed to study the complexity of the computation of the Banzhaf index in restricted games with few maximal independent sets. In what follows, we will mostly focus on co-planar restricted games. Later on, we will discuss the consequences our results have for the general case.

The choice of this specific class of incompatibility graphs is not arbitrary. In [42] it is shown not only that planar graphs have few maximal cliques, but that the number of such sets in an $n$-vertex planar graph is bounded by a linear function of $n$. Moreover, [39] provides a linear time algorithm to compute them. Due to Property 2.2, this directly implies that $n$-vertex co-planar graphs have a linear number of maximal independent sets with respect to $n$, and that these sets can be computed efficiently.

Furthermore, there are computational problems on graphs for which it is meaningful to restrict the inputs to planar graphs. In this context, there are problems that are hard in general, but polynomial time solvable when restricted to planar graphs; others remain hard for planar graphs.

Example 4.1. Let $G = (V, E)$ be a graph, $k \leqslant |E|$. Consider the problem of deciding whether $G$ contains a bipartite subgraph with $k$ or more edges, i.e. there is some $E' \subseteq E$ such that $|E'| \geqslant k$ and $G' = (V, E')$ is bipartite. This problem is NP-complete in general, but can be solved in polynomial time if $G$ is planar (for details, see the problem BIPARTITE SUBGRAPH in the final chapter of [24]).

Example 4.2. Let $G = (V, E)$ be a graph, $k \leqslant |V|$. Consider the problem of deciding whether $G$ contains a clique of size $k$ or larger. This problem can be solved in polynomial time if $G$ is a planar graph, but is NP-complete in general. Recall that, due to Property 2.5, a planar graph cannot contain cliques of size larger than four.

The problem of deciding whether $G$ contains an independent set of size $k$ or more is NP-complete even if $G$ is a planar graph (for details, see problems CLIQUE and INDEPENDENT SET in the final chapter of [24], respectively). Note that it follows from Property 2.2 that INDEPENDENT SET can be solved in polynomial time if $G$ is a co-planar graph, while CLIQUE remains NP-complete for this class.

Both examples above are of decision problems. Below, we introduce some counting problems that will be relevant to the subsequent results of this section.

EXAMPLE 4.3. Counting the number of perfect matchings in a graph $G = (V, E)$ is a strongly #P-complete problem. However, the computation can be performed in polynomial time if $G$ is planar. An algorithm to do so is described in Section 1.2. of [27].

In contrast, the problem of counting *all* matchings in $G$ is strongly #P-complete even if $G$ is planar.

EXAMPLE 4.4. The problem of counting the number of matchings in a 2,3-regular bipartite planar graph is strongly #P-complete (see [55] for details[16], and for a thorough study on the complexity of counting problems restricted to planar graphs).

This latter example will be used as the starting point to show that, even when the incompatibility graph has few maximal independent sets, computing the Banzhaf index in a restricted game is a strongly #P-complete problem. We will see that our arguments reveal slightly stronger results. In order to formalize a suitable reduction, we will first describe a polynomial time procedure to build a co-planar graph from a 2,3-regular bipartite planar graph. Then, we will assess how the number of swings for a player in a game restricted by the resulting incompatibility graph embodies the computation of the number of matchings in the original graph. To these ends, the following result from graph theory will be of great importance.

PROPERTY 4.1 ([44]). The line graph of a graph $G = (V, E)$ is planar if and only if all of the following conditions are satisfied:

(i) $G$ is planar.

(ii) No vertex of $G$ has degree larger than 4.

(iii) Any vertex of $G$ of degree 4 is a cut-vertex[17].

In particular, the line graph of any 2,3-regular bipartite planar graph is planar. Indeed, by definition, such a graph is planar and all of its vertices have degree 2 or 3; the third condition in Property 4.1 need not be used[18]. Note that given an $n$-vertex graph $G = (V, E)$, its line graph $L(G)$ has $|E| \leqslant \binom{n}{2} = O(n^2)$ vertices, so it can be constructed in polynomial time with respect to $n$. Moreover, recall from Property 2.3 that there is a one-to-one correspondence between

[16]In a purely technical note, the relevant reduction in the reference is not completely a Turing reduction, which is the notion we have used to define #P-completeness. Indeed, at a certain point, a *holographic reduction* is needed. It is beyond the scope of this work to define these; see [53, 14]. For our purposes, it is important that holographic reductions provide a correspondence between the results of instances of two different counting problems. As such, in a way, they are constant time computable, and so can be embedded within a Turing reduction.

[17]Given a connected graph $G = (V, E)$, a vertex $v \in V$ is said to be a *cut-vertex* (or *separating vertex*) of $G$ when $G \setminus v$ is not a connected graph.

[18]It is worth mentioning that, for graphs with no vertex with degree larger than 3, Sedláček states in [44] that "it is obvious that planarity of $G$ implies planarity of [its line graph, $L(G)$]". No proof for this statement is given thereafter. Not only that, but it seems as though such a proof has never been published. In Proposition A.1 we provide a constructive proof for Sedláček's obvious statement.

matchings in a graph and independent sets in its line graph. The following result establishes the connection between swings in a restricted game and cliques in its incompatibility graph (which, due to Property 2.2, correspond to independent sets in this graph's complement).

DEFINITION 4.1. Let $\Gamma = [q;\, w_1, \ldots, w_n]$ be a weighted voting game. If $w_j = 1\ \forall j \in N = \{1, \ldots, n\}$, we say $\Gamma$ is a ($n$-player) *majority* game (with quota $q$). If $\Gamma$ is a majority game with quota $q = \lceil \frac{n}{2} \rceil$, we say it is a ($n$-player) *simple* majority game ([26]).

LEMMA 4.1. Let $G = (V, E)$ be a graph, $n = |V|$ and $\Gamma$ be the $n$-player majority game with quota $q = 2$, and consider the $G$-restriction of $\Gamma$. For any $i \in V$, a coalition $S \subseteq V \setminus \{i\}$ is a swing for player $i$ in the restricted game if and only if its vertices form a (non-empty) clique in $G$ and $S \cup \{i\}$ is not a clique.

*Proof.* Let $\mathcal{M}$ be the collection of maximal independent sets in $G$. Let $i \in V$ be arbitrarily chosen, and denote the collection of maximal independent sets that contain $i$ by $\mathcal{M}_i$. Recall from the proof of Proposition 3.2, that a coalition $S \subseteq V \setminus \{i\}$ is a swing for $i$ in the restricted game if and only if there is some $M \in \mathcal{M}_i$ such that $q - w_i \leqslant w(S \cap M) \leqslant q - 1$, while $w(S \cap M') \leqslant q - 1$ for any maximal independent set $M' \neq M$.

Now, in the game $\Gamma$ at issue, $w_j = 1\ \forall j \in V$ and $q = 2$. Hence, $w(S) = \sum_{j \in S} w_j = |S|\ \forall S \subseteq V$. As a consequence, in the restricted game, $S \subseteq V \setminus \{i\}$ is a swing for $i$ if and only if $|S \cap M| \leqslant 1\ \forall M \in \mathcal{M}$ and equality holds for some $M \in \mathcal{M}_i$.

In other words, swings for $i$ are characterized as coalitions that do not contain two vertices of a same maximal independent set in $G$, while they contain a vertex that is not adjacent to $i$ in $G$; this is precisely what we wanted to show. □

COROLLARY 4.1. Let $H = (V, E)$ and $u \notin V$. Also let $\Gamma$ be a majority game with player set $V \cup \{u\}$ and quota $q = 2$, and consider the $(H \cup u)$-restriction of $\Gamma$. The number of swings for $u$ in the restricted game equals the number of (non-empty) cliques in $H$.

*Proof.* Since, by definition, $u$ is an isolated vertex in $H \cup u$, there is no non-empty coalition $S \subseteq V$ such that $S \cup \{u\}$ forms a clique in $H \cup u$. Thus, in this case Lemma 4.1 shows that $S \subseteq V$ is a swing for $u$ if and only if its vertices form a non-empty clique in $H$. □

PROPOSITION 4.1. Counting swings in a restricted game with few maximal independent sets is strongly #P-complete.

*Proof.* The problem is in #P due to Proposition 3.2; we only need to show #P-hardness in the strong sense. To do so, we reduce from the problem of counting all matchings in a 2,3-regular bipartite planar graph (see Example 4.4). Let $G = (V, E)$ be such a graph.

As previously argued, by Property 4.1, the line graph of $G$, $L(G)$, is planar. Hence, its complement, call it $H = (W, F)$, is co-planar. In particular, $H$ has few maximal independent sets. Moreover, since $L(G)$ can be constructed in polynomial time with respect to $n = |V|$, so can $H$.

Note that, given $u \notin W$, there are the same number of independent sets in $H \cup u$ than in $H$. Indeed, observe that no $S \subseteq W$ can be a maximal independent set in $H \cup u$: by definition, $u$ is an isolated vertex in this graph, so if $S \subseteq W$ is an independent set in $H$, then $S \cup \{u\}$ is an independent set in $H \cup u$.[19] Furthermore, if $S \subseteq W$ is not a maximal independent set in $H$, then $S \cup \{u\}$ cannot be a maximal independent set in $H \cup u$. Hence, the maximal independent sets in $H \cup u$ are all exactly those of the form $S \cup \{u\}$, where $S \subseteq W$ is a maximal independent set in $H$. For our purposes, this implies that $H \cup u$ also has few maximal independent sets.

All of this being said, recall from Property 2.3 that the number of matchings in $G$ is equal to the number of independent sets in $L(G)$. Moreover, by Property 2.2, this quantity coincides with the number of cliques in $H = \overline{L(G)}$. Finally, by Corollary 4.1, we can compute the latter value as the number of swings for player $u$ in a restricted game with incompatibility graph $H \cup u$. In summary, the computation of the number of matchings in $G$ polynomial time reduces to counting swings in a restricted game with few maximal independent sets, as desired. □

Corollary 4.2. Counting the independent sets in a perfect planar graph is strongly #P-complete.

*Proof.* This is an intermediate result that appears in the proof of the previous proposition. The problem is in #P, because we can efficiently check whether a given set of vertices of a graph is an independent set.

For #P-hardness, let $G$ be a 2,3-regular bipartite planar graph. As was pointed out in the previous proof, $L(G)$ is a planar graph. By Property 2.4, $L(G)$ is also a perfect graph. The corollary follows from Property 2.3 and the fact that the line graph of $G$ can be constructed in polynomial time. □

This corollary strengthens some previously known graph theoretical results. Namely, #P-complete-ness for counting independent sets had already been established for both planar ([51]) and perfect[20] graphs. Corollary 4.2 shows that the problem remains intractable for graphs that are planar *and* perfect at the same time.

On the other hand, even though our proof for Proposition 4.1 suffices to deduce strong #P-completeness for the Banzhaf index in restricted games with few maximal independent sets, it does not prove such classification if the incompatibility graph is co-planar. This is due to the graph $H \cup u$ in the proof having an isolated vertex. In the complement graph, this vertex becomes universal; in general, we cannot ensure a graph with a universal vertex to be planar. As a consequence, $H \cup u$ may not be co-planar. The following lemma shows that, given an incompatibility graph $G$ with an isolated vertex $u$, there are relevant cases in which we can compute the Banzhaf index in the game restricted by $G$ via the Banzhaf index in the game restricted by $G \setminus u$.

[19] In other words, if a vertex is isolated in a graph, it must be contained in all of its maximal independent sets.

[20] In this case, #P-hardness follows from the #P-completeness of counting matchings of a bipartite graph ([27, 52]), combined with Property 2.3.

Definition 4.2. Let $k \geqslant 0$. We define the class of graphs $\mathcal{I}_k$ as that of graphs $G$ with at most $k$ isolated vertices.

Remark 4.1. For any $k$, $l \geqslant 0$, if $l > k$, the classes of graphs defined above satisfy $\mathcal{I}_k \subsetneq \mathcal{I}_l$.

Lemma 4.2. For $n \geqslant 1$, let $\Gamma_n$ be the $n$-player majority game with quota $q = 2$, $N = \{1, \ldots, n\}$. Let $k \in \{0, \ldots, n-1\}$. Counting swings in a $\mathcal{I}_{k+1}$-restriction of $\Gamma_n$ polynomial time Turing reduces to counting swings in a $\mathcal{I}_k$-restriction of $\Gamma_n$.

*Proof.* Let $G = (N, E) \in \mathcal{I}_{k+1}$. Let $v_G^I$ be the characteristic function of the $G$-restriction of $\Gamma_n$, so that, for $i \in N$, $\eta_i\left(v_G^I\right)$ is the number of swings for player $i$ in the corresponding restricted game. We seek to show that, for any $i \in N$, we can compute $\eta_i\left(v_G^I\right)$ in polynomial time, using an oracle to the number of swings in $\mathcal{I}_k$-restricted games. If $G$ has $k$ or less isolated vertices, then $G \in \mathcal{I}_k$, and the result is trivial. Thus, assume that $G$ has exactly $k+1$ isolated vertices, and, without loss of generality, suppose vertex $n$ is isolated.

Suppose $i \neq n$, and consider the graph $G \setminus n$. Note that the graph can be constructed in polynomial time, and $G \setminus n \in \mathcal{I}_k$. Now, recall from Lemma 4.1 that $\eta_i\left(v_G^I\right)$ equals the number of cliques $S \subseteq N \setminus \{i\}$ in $G$ such that $S \cup \{i\}$ is not a clique. Let $S$ satisfy this characterization. Since $n$ is an isolated vertex, if $n \in S$, then $S = \{n\}$, because $S$ must be a clique in $G$. If $n \notin S$, then $S$ is a clique in $G \setminus n$ such that $S \cup \{i\}$ is not a clique in this same graph; by definition, there are $\eta_i\left(v_{G\setminus n}^I\right)$ coalitions satisfying this description. Thus, $\eta_i\left(v_G^I\right) = \eta_i\left(v_{G\setminus n}^I\right) + 1$; as $G \setminus n$ can be computed in polynomial time, the right hand side of this expression can also be obtained in polynomial time with the given oracle.

For $i = n$, having assumed that $n$ is an isolated vertex of $G$, Corollary 4.1 shows that $\eta_n\left(v_G^I\right)$ is the number of cliques in $G \setminus n$. It will be useful to express this value as $\eta_n\left(v_G^I\right) = \sum_{l=1}^{n-1} \widetilde{\eta}_l$, where $\widetilde{\eta}_l$ is the number of cliques in $G$ that contain vertex $l$ but no vertex $l' < l$. For $l \in \{1, \ldots, n-1\}$, consider the graph $G_l = (N_l, E_l)$, where $N_l = \{l' \in N : l' \geqslant l\}$ and, given $u, w \in N_l$, $\{u, w\} \in E_l$ if and only if either $\{u, w\} \in E$, or $u = n$ and $w > l$. Again, observe that for all $l$ we have $G_l \in \mathcal{I}_k$; in fact, $G_l \in \mathcal{I}_2$, since $l$ and $n$ are the only potentially isolated vertices in the graph.

We claim $\widetilde{\eta}_l = \eta_n\left(v_{G_l}^I\right)$. Indeed, by Lemma 4.1, $\eta_n\left(v_{G_l}^I\right)$ is the number of cliques $S \subseteq N_l$ in $G_l$ such that $S \cup \{n\}$ is not a clique in this same graph. Note that all such $S$ contains $l$, since, by construction, all other vertices of $G_l$ are adjacent to $n$. Moreover, condition $S \subseteq N_l$ implies no vertex $l' < l$ of $G$ is contained in $S$. It follows that there is a one-to-one correspondence between swings for $n$ in $v_{G_l}^I$ and cliques in $G$ that contain $l$ but no vertex $l' < l$, which proves our claim.

Finally, observe that each of the graphs $G_l$ can be constructed in polynomial time in $n$, and there are $n-1$ of them. Hence, the computation of the values $\eta_n\left(v_{G_l}^I\right)$, $l \in \{1, \ldots, n-1\}$, can be done in polynomial time; in conclusion, with the given oracle, we can compute $\eta_n\left(v_G^I\right)$ in polynomial time. □

Proposition 4.2. Counting swings in a perfect co-planar restricted game is a strongly #P-complete problem.

*Proof.* Since the class of co-planar graphs has few maximal independent sets, it follows from Proposition 3.2 that the problem is in #P. Note that Proposition 4.1 shows strong #P-completeness for counting swings in restricted games where the incompatibility graph $G = (V, E)$ is of the form $G = H \cup u$, where $H = (W, F)$ is a perfect co-planar graph and $u \notin W$. Let $V = \{1, \ldots, n\}$ and assume, without loss of generality, that $u = n$. Lemma 4.2 shows that the number of swings for $u$ in such a restricted game can be computed in polynomial time with an oracle to the number of swings in the game restricted by the graphs $G_l$ therein described.

In order to prove the proposition, it will suffice to observe that, if $H$ is perfect and co-planar, then all of the $G_l$ are both perfect and co-planar as well. We start by showing that co-planarity is preserved. To this end, note that for any valid $l$, $G_l$ can be seen as a subgraph of $H$, call it $H_l$, to which a new vertex $u$ is added. In $G_l$, this new vertex is adjacent to all other vertices except for one. Now, due to Property 2.1, since $H_l$ is an induced subgraph of $H$, and $H$ is assumed to be co-planar, so is $H_l$. As such, $\overline{G_l}$ can be seen as $\overline{H_l}$ with an additional vertex that is adjacent to only one vertex, $l$. In other words, this new vertex is a pendant vertex in $\overline{G_l}$. Thus, we can extend any planar embedding of $\overline{H_l}$ to a planar embedding of $\overline{G_l}$, as desired.

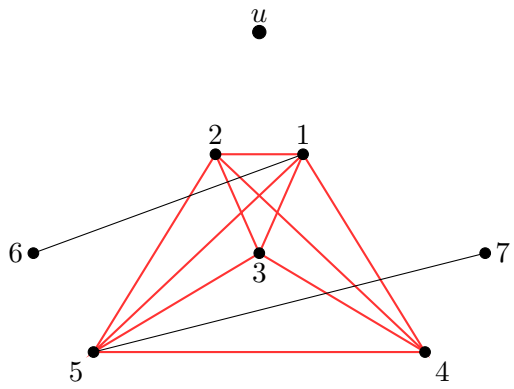
By Property 2.4, to see that $G_l$ is a perfect graph, it will suffice to show that $\overline{G_l}$ is a perfect graph. To do so, note that, by definition, every induced subgraph of a perfect graph is itself a perfect graph. In particular, $H_l$ is a perfect graph, and, once more by Property 2.4, so is $\overline{H_l}$. Finally, as argued in the previous paragraph, $\overline{G_l}$ is the result of adding a pendant vertex to $\overline{H_l}$, and so, applying Property 2.4 yet again, it follows that $\overline{G_l}$ is a perfect graph. □

In Figure 4 we illustrate how, as argued in the previous result, graphs $G_l$ preserve co-planarity of $H$. Consider the graph in Figure 4a. Note that vertex $u$ is isolated in $G$, and let $H = G \setminus u$; we have $G \in \mathcal{I}_1$ and $H \in \mathcal{I}_0$. Moreover, $H$ is not a planar graph, as the subgraph induced by vertices 1 to 5 (in red) is $K_5$. The complement of $H$ is planar, and so $H$ is a co-planar graph. However, the complement of $G$ is not planar, because vertex $u$ is universal in this graph. Indeed, Figure 4b represents $\overline{G}$, with the edges in red showing that it contains $K_{3,3}$ as a subgraph.

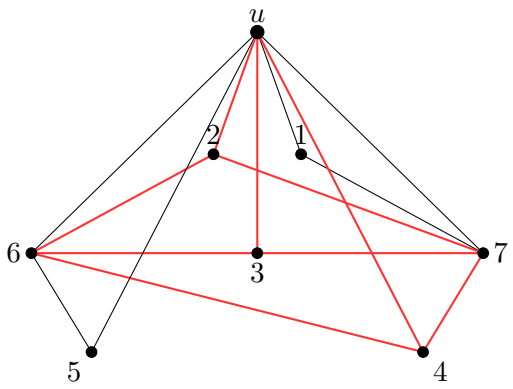
Now, we initiate the procedure described in Lemma 4.2 to compute the number of swings for $u$ in the $G$-restriction of $\Gamma_n$. Figure 4c shows graph $G_1 \in \mathcal{I}_0$ as defined in the aforementioned result; Figure 4d shows the complement of $G_1$. Note that, as pointed out in Proposition 4.2, this graph is the complement of $H_1 = G_1 \setminus u$ with $u$ added as a pendant vertex adjacent to vertex 1. In particular, since $H_1$ is a subgraph of $H$, which is co-planar, so is $H_1$.

Figure 4: An example to illustrate Proposition 4.2.

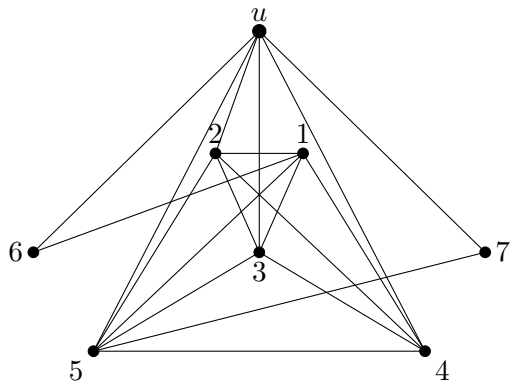
(a) Graphs $G$ and $H = G \setminus u$.



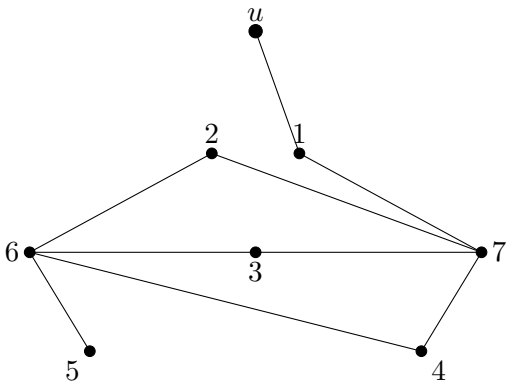

(b) The complement of graph $G$.




(c) Graphs $G_1$ and $H_1 = G_1 \setminus u$.




(d) The complement of $G_1$, a planar graph.




After having dealt with the complexity of power indices in co-planar restricted games, we provide some notes on the consequences our results have beyond this class. Observe that Proposition 4.1 implies that, even if the incompatibility graph has few maximal independent sets, there is no pseudopolynomial time algorithm to compute the Banzhaf index in restricted games. Indeed, such an algorithm should apply to co-planar restricted games, for which our findings rule out the existence of such algorithms (unless $\mathsf{FP} = \#\mathsf{P}$). However, there may still be particular subclasses of graphs with few maximal independent sets which admit a pseudopolynomial time algorithm for the problem at issue. In the following section we will suggest some candidate classes.

Finally for this section, we discuss the complexity of deciding whether $\eta_i(v) > 0$ in restricted games with few maximal independent sets. As discussed in Section 3, due to Proposition 3.2, the decision problem at issue is in $\mathsf{NP}$. Also recall that the problem is weakly $\mathsf{NP}$-complete for unrestricted games (Property 2.7), double star restricted games (Proposition 3.4) and complete multipartite restricted games (Proposition 3.6). The following result provides a correspondence between the decision problem for restricted and unrestricted games.

LEMMA 4.3. Let $\Gamma = [q;\ w_1, \ldots, w_n]$ be a weighted voting game, $N = \{1, \ldots, n\}$, $G = (N, E)$ be a graph. Let $\mathcal{M} = \{M_1, \ldots, M_\mu\}$ be the collection of maximal independent sets in $G$, and for each $l \in \{1, \ldots, \mu\}$, let $M_l = \{i_1, \ldots, i_{n_l}\}$ and consider the weighted voting game $\Gamma_l = \left[q;\ w_{i_1}, \ldots, w_{i_{n_l}}\right]$. Let $v_l$ be the characteristic function of $\Gamma_l$ and $v_G^I$ be the characteristic function of the $G$-restriction of $\Gamma$. For each player $i \in N$, $\eta_i\left(v_G^I\right) > 0$ if and only if $\eta_i(v_l) > 0$ for some $l$.

*Proof.* Suppose $\eta_i\left(v_G^I\right) > 0$, and let $S \subseteq N \setminus \{i\}$ be a swing for $i$ in the restricted game. As seen in the proof of Proposition 3.2, for any such $S$ there is some maximal independent set $M_l$ such that $i \in M_l$ and $q - w_i \leqslant w(S \cap M_l) \leqslant q - 1$. In particular, we can find some $l$ such that $S \cap M_l$ is a swing for $i$ in $\Gamma_l$, and so $\eta_i(v_l) > 0$.

We proceed similarly to prove the converse statement. Namely, suppose $l \in \{1, \ldots, \mu\}$ is such that $\eta_i(v_l) > 0$, and let $S \subseteq M_l \setminus \{i\}$ be a swing for $i$ in $\Gamma_l$. The proof ends by observing that $S$ is also a swing for $i$ in the restricted game. Indeed, since the quota in both $\Gamma_l$ and the original weighted voting game $\Gamma$ is $q$, we have that $w(S) \leqslant q - 1$, because $S$ is assumed to be a swing for $i$ in $\Gamma_l$. In particular, $w(S \cap M_{l'}) \leqslant w(S) \leqslant q - 1\ \forall l' \neq l$. Moreover, $q - w_i \leqslant w(S \cap M_l) = w(S) \leqslant q - 1$, and so $S$ satisfies the characterization for swings in the restricted game given in the proof for Proposition 3.2. □

Note that the correspondence Lemma 4.3 establishes is not one-to-one. This loosely follows from the characterization of swings in restricted games provided in Proposition 3.2. Namely, given a swing $S$ for $i$ in the restricted game, there may be more than one maximal independent set $M_l$ for which $S \cap M_l$ is a swing for $i$ in $\Gamma_l$, where $\Gamma_l$ is as introduced in the lemma. Conversely, given a swing $S \subseteq M_l \setminus \{i\}$ for $i$ in $\Gamma_l$, several swings $T \subseteq N \setminus \{i\}$ for $i$ in the restricted game may satisfy $T \cap M_l = S$.

In particular, we cannot compute the number of swings of a player in the restricted game solely through the number of swings in games $\Gamma_l$, which are each unrestricted weighted voting games with players within a maximal independent set in the incompatibility graph. We argue that this was to be expected. A restricted game with few maximal independent sets would yield a polynomial number of games $\Gamma_l$ in terms of the number of players, $n$. By Property 2.7, we can compute each of the values $\eta_i(v_l)$ in pseudopolynomial time. Thus, if we could "easily" use them to retrieve the number of swings in the restricted game, we would have a pseudopolynomial time algorithm to compute the latter value. We already discussed that, in light of the results of this section, no such procedure can exist.

However, Lemma 4.3 does imply the following result.

PROPOSITION 4.3. Let $\Gamma = [q;\ w_1, \ldots, w_n]$ be a weighted voting game, $N = \{1, \ldots, n\}$, $G = (N, E)$ be a graph with few maximal independent sets, and $v_G^I$ be the characteristic function of the $G$-restriction of $\Gamma$. For any player $i \in N$, we can decide whether $\eta_i\left(v_G^I\right) > 0$ in pseudopolynomial time.

*Proof.* It will suffice to construct a pseudopolynomial time procedure to solve the problem. Let $\mathcal{M} = \{M_1, \ldots, M_\mu\}$ be the collection of maximal independent sets in $G$. As discussed in Section 3, we can compute $\mathcal{M}$ in polynomial time with respect to $n$. Furthermore, we can also check in polynomial time whether $i \in M_l$ for a given $M_l \in \mathcal{M}$.

Now, for each $l \in \{1, \ldots, \mu\}$ such that $i \in M_l$, check whether $\eta_i(v_l) > 0$, where $v_l$ is the characteristic function of the corresponding game $\Gamma_l$ defined in Lemma 4.3. If $\eta_i(v_l) > 0$ indeed, stop the procedure; conclude that $\eta_i\left(v_G^I\right) > 0$. Otherwise, if all required values of $l$ have been checked, and $\eta_i(v_l) = 0$ in all cases, conclude that $\eta_i\left(v_G^I\right) = 0$ as well.

Lemma 4.3 ensures that this algorithm always returns the right answer. On the other hand, by Property 2.7, for each $l$ we can check whether $\eta_i(v_l) > 0$ in pseudopolynomial time. In the worst case, we have to perform as many such tests as there are maximal independent sets in $G$. We have assumed $G$ to have few of those, and so it follows that the described procedure runs in pseudopolynomial time. □

## 5. Further work and final remarks

The main result of this work is strong #P-completeness for the Banzhaf index in perfect co-planar restricted games. As pointed out in the previous section, this implies that, if $\mathsf{FP} \neq \#\mathsf{P}$, there is no pseudopolynomial time algorithm to compute this index in restricted games with few maximal independent sets, since these include co-planar restricted games.

Nonetheless, the existence of a pseudopolynomial time algorithm that applies to more limited cases is not ruled out. More specifically, for some class $\mathcal{C}$ of graphs with few maximal independent sets, there may be such algorithms to compute the Banzhaf index in $\mathcal{C}$ restricted games. In what follows, we introduce two classes of graphs whose structure makes them compelling candidates for this to be the case.

A graph $G = (V, E)$ is said to be *chordal* if and only if its maximal cliques can be organized in a *clique tree* ([48]). A clique tree of $G$ is a tree graph $T(G)$ whose vertices represent the maximal cliques in $G$, and such that any two adjacent vertices of $T(G)$ represent maximal cliques that share some vertex. It is also required that, for each $v \in V$, the maximal cliques in $G$ that contain $v$ form a subtree of the clique tree. There are efficient algorithms to decide whether a graph is chordal, and construct its clique tree should this be the case ([12, 48]). Moreover, in [42], it is shown that chordal graphs have few maximal cliques.

All in all, this implies that co-chordal graphs have few maximal independent sets, and, more importantly, these can be organized in an "independent set tree". Trees can be explored efficiently via depth or breadth-first procedure. We may be able to exploit this structure to efficiently compute power indices in co-chordal restricted games. In this sense, the results in [36] are encouraging. It is shown there that several #P-hard graph-related counting problems are solvable in polynomial time on chordal graphs. In particular, the independent sets of a chordal graph can be counted in linear time. However, it is also relevant to point out that chordal graphs are a subclass of perfect graphs ([12]); our results imply that counting swings is strongly #P-complete in perfect restricted games.

An interesting, albeit "small" ([12]), subclass of chordal graphs is that of *threshold* graphs. A graph $G = (V, E)$ is a threshold graph if and only if there exists an ordering $v_1, \ldots, v_n$ of its vertices and a partition of $V \setminus \{v_1\}$ into disjoint sets $P$ and $Q$ such that $v_i \in P$ if and only if it is adjacent to every $v_j \in V$ with $j < i$, and $v_i \in Q$ if and only if it is adjacent to no such vertex ([18]). It is not too difficult to see that this class has few maximal independent sets. A deeper look into their structure makes it plausible that we could build a polynomial time algorithm to compute the Banzhaf index at least in threshold restricted[21] majority games.

On the other hand, instead of limiting the class of incompatibility graphs, we could focus on particular classes of weighted voting games. Recall that computing the Banzhaf index in unrestricted weighted voting games is already known to be weakly #P-complete. However, there are subclasses of weighted voting games in which power indices can be computed in polynomial

[21] Due to its forbidden subgraph characterization in [18], the class of threshold graphs is closed under complementation, and so it is equal to the class of co-threshold graphs.

time. A detailed analysis on this issue can be found in [17]. It may be worth studying the complexity of power indices in the restrictions of the subclasses in this reference.

Our work could also be expanded by studying the complexity of power indices if the incompatibility graph has "many" maximal independent sets. As pointed out in Section 3, it is unlikely that the problem remains in #P. The main reference for counting classes beyond #P is [25], which may prove useful to classify the problem at issue. We could also study the complexity of the Shapley-Shubik index in restricted games. Although we focused on the Banzhaf index, we argued that most of our results also hold for the Shapley-Shubik index, with only slight modifications on the proofs. However, even though we suspect strong #P-completeness for the latter index in co-planar restricted games, we would need a separate reduction to prove this result.

Now, to conclude, we summarize our other results. Recall that our study started as an analysis of the algorithm in [4]. In this regard, our first observation was that its requirement to compute the maximal independent sets in a graph made this algorithm inefficient. In spite of this, we argued that the algorithm runs in pseudopolynomial time if the incompatibility graph has very few maximal independent sets. Since we showed this problem to be weakly #P-complete, there is no polynomial time algorithm to solve it; the problem remains hard even if the incompatibility graph is a double star or a complete multipartite graph.

All in all, the studied algorithm is optimal in these cases. Moreover, in light of our results, there is no meaningfully faster way to compute the Banzhaf index in restricted games with few maximal independent sets either.

Finally, we also studied the complexity of deciding whether the Banzhaf index is non-zero in a restricted game with few maximal independent sets. In contrast to the exact computation of the index, we showed NP-completeness for this decision problem to be in the weak sense. We also sketched a pseudopolynomial time procedure to solve it in Proposition 4.3.

## References


[1] Akkoyonlu, E. A. (1973): "The enumeration of maximal cliques of large graphs". *Society for Industrial and Applied Mathematics Journal on Computing*, 2: pp. 1–6.

[2] Algaba, E., Bilbao, J. M., Fernández, J. R., Jiménez, A., Jiménez, N. and López, J. J. (2002): "Generating functions for computing the Myerson value". *Annals of Operations Research*, 109: pp. 143–158.

[3] Alonso-Meijide, J. M., Álvarez-Mozos, M. and Fiestras-Janeiro, M. G. (2009): "The Banzhaf value when some players are incompatible". *Homo Oeconomicus*, 26: pp. 403–415.

[4] Alonso-Meijide, J. M., Casas-Méndez, B. and Fiestras-Janeiro, M. G. (2015): "Computing Banzhaf-Coleman and Shapley-Shubik power indices with incompatible players". *Applied Mathematics and Computation*, 252: pp. 377–387.

[5] Alonso-Meijide, J. M. and Fiestras-Janeiro, M. G. (2006): "The Banzhaf value and communication situations". *Naval Research Logistics*, 53: pp. 198–203.

[6] Banzhaf, III, J. F. (1965): "Weighted voting doesn't work: a mathematical analysis". *Rutges Law Review*, 19: pp. 317–343.

[7] Banzhaf, III, J. F. (1968): "One man, 3.312 votes: a mathematical analysis of the Electoral College". *Villanova Law Review*, 13: pp. 304–346.

[8] Behr, R., Sivaraman, V. and Zaslavsky, T. (2018): "Mock threshold graphs". *Discrete Mathematics*, 341: pp. 2159–2178.

[9] Benati, S., Rizzi, R. and Tovey, C. (2015): "The complexity of power indexes with graph restricted coalitions". *Mathematical Social Sciences*, 76: pp. 53–63.

[10] Bergantiños, G., Carreras, F. and García-Jurado, I. (1993): "Cooperation when some players are incompatible". *Zeitschrift für Operations Research - Methods and Models of Operations Research*, 38: pp. 187–201.

[11] Bilbao, J. M., Fernández, J. R., Jiménez, A. and López, J. J. (2000): "Generating functions for computing power indices efficiently". *Top*, 8: pp. 191–213.

[12] Brandstädt, A., Le, V. B. and Spinrad, J. P. (1999): *Graph classes: a survey*. 4th Edition. Society for Industrial and Applied Mathematics, Philadelphia.

[13] Bron, C. and Kerbosch, J. (1973): "Algorithm 457: finding all cliques of an undirected graph". *Communications of the Association for Computing Machinery*, 16: pp. 575–577.

[14] Cai, J. Y. and Choudhary, V. (2007): "Valiant's Holant theorem and matchgate tensors". *Theoretical Computer Science*, 384: pp. 22–32.

[15] Carreras, F. (1991): "Restriction of simple games". *Mathematical Social Sciences*, 21: pp. 245–260.

[16] Carreras, F. and Owen, G. (1995): "Valor coalicional y estrategias parlamentarias". *Revista Española de Investigaciones Sociológicas*, 71/72: pp. 157–176.

[17] Chakravarty, N., Goel, M. and Sastry, T. (2000): "Easy weighted majority games". *Mathematical Social Sciences*, 40: pp. 227–235.

[18] Chvátal, V. and Hammer, P. L. (1977): "Aggregation of inequalities in integer programming". *Annals of Discrete Mathematics*, 1: pp. 145–162.

[19] Cook, S. A. (1971): "The complexity of theorem-proving procedures". In: *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*. Association for Computing Machinery, New York, pp. 151–158.

[20] Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein, C. (2009): *Introduction to algorithms*. 3rd Edition. The MIT Press, Cambridge.

[21] Courcelle, B., Makowsky, J. A. and Rotics, U. (2000): "Linear time solvable optimization problems on graphs of bounded clique-width". *Theory of Computing Systems*, 33: pp. 125–150.

[22] Deng, X. and Papadimitriou, C. H. (1994): "On the complexity of cooperative solution concepts". *Mathematics of Operations Research*, 19: pp. 257–266.

[23] Fáry, I. (1948): "On straight line representation of planar graphs". *Acta Universitatis Szegediensis. Sectio Scientiarum Mathematicarum*, 11: pp. 229–233.

[24] Garey, M. R. and Johnson, D. S. (1979): *Computers and intractability: a guide to the theory of* NP*-completeness*. 1st Edition. W.H. Freeman and Co., New York.

[25] Hemaspaandra, L. A. and Vollmer, H. (1995): "The Satanic Notations: counting classes beyond #P and other definitional adventures". *Special Interest Group on Algorithms and Computation Theory News*, 26: pp. 2–13.

[26] Izquierdo i Aznar, J. M., Marín Solano, J., Martínez de Albéniz Salas, F. J., Núñez Oliva, M. and Ybern Carballo, N. (1999): *Jocs cooperatius i aplicacions econòmiques*. Ed. by C. R. i Pallarola. Edicions Universitat de Barcelona, Barcelona.

[27] Jerrum, M. (2003): *Counting, sampling and integrating: algorithms and complexity*. Ed. by M. Struwe. 1st Edition. Birkhäuser Verlag, Basel.

[28] Karp, R. (1972): "Reducibility among combinatorial problems". In: *Complexity of Computer Computations*. Ed. by R. E. Miller and J. W. Thatcher. Plenum Press, New York, pp. 85–103.

[29] Lovász, L. (1972): "Normal hypergraphs and the perfect graph conjecture". *Discrete Mathematics*, 2: pp. 253–267.

[30] Lovász, L. and Plummer, M. D. (1986): *Matching Theory*. Ed. by P. L. Hammer. 1st Edition. Vol. 29. Annals of Discrete Mathematics, Amsterdam.

[31] Matsui, T. and Matsui, Y. (2000): "A survey of algorithms for calculating power indices of weighted majority games". *Journal of the Operations Research Society of Japan*, 43: pp. 71–86.

[32] Matsui, T. and Matsui, Y. (2001): "NP-completeness for calculating power indices of weighted majority games". *Theoretical Computer Science*, 263: pp. 305–310.

[33] Moon, J. W. and Moser, L. (1965): "On cliques in graphs". *Israel Journal of Mathematics*, 3: pp. 23–28.

[34] Myerson, R. B. (1977): "Graphs and cooperation in games". *Mathematics of Operations Research*, 2: pp. 225–229.

[35] Neto, A. F. and Fonseca, C. R. (2019): "An approach via generating functions to compute power indices of multiple weighted voting games with incompatible players". *Annals of Operations Research*, 279: pp. 221–249.

[36] Okamoto, Y., Uno, T. and Uehara, R. (2008): "Counting the number of independent sets in chordal graphs". *Journal of Discrete Algorithms*, 6: pp. 229–242.

[37] Owen, G. (1978): "Characterization of the Banzhaf-Coleman index". *Society for Industrial and Applied Mathematics Journal on Applied Mathematics*, 35: pp. 315–327.

[38] Owen, G. (1986): "Values of graph-restricted games". *Society for Industrial and Applied Mathematics Journal on Algebraic and Discrete Methods*, 7: pp. 210–220.

[39] Papadimitriou, C. H. and Yannakakis, M. (1981): "The clique problem for planar graphs". *Information Processing Letters*, 13: pp. 131–133.

[40] Papadimitriou, C. H. (1994): *Computational complexity*. Addison-Wesley Publishing Company, Reading.

[41] Prasad, K. and Kelly, J. S. (1990): "NP-completeness of some problems concerning voting games". *International Journal of Game Theory*, 19: pp. 1–9.

[42] Prisner, E. (1995): "Graphs with few cliques". In: *Graph Theory, Combinatorics, and Algorithms: Proceedings of the 7th Quadrennial International Conference on the Theory and Applications of Graphs*. Ed. by Y. Alavi and A. Schwenk. Wiley, New York, pp. 945–956.

[43] Rosgen, B. and Stewart, L. (2005): "Complexity results on graphs with few cliques". *Discrete Mathematics and Theoretical Computer Science*, 9: pp. 258–266.

[44] Sedláček, J. (1964): "Some properties of interchange graphs". In: *Theory of Graphs and its Applications*. Publishing House of the Czechoslovak Academy of Sciences, Prague, pp. 145–150.

[45] Shapley, L. S. (1953): "A Value for n-Person Games". In: *Contributions to the Theory of Games*. Ed. by H. Kuhn and A. Tucker. Vol. 2. Princeton University Press, Princeton, pp. 307–317.

[46] Shapley, L. S. and Shubik, M. (1954): "A method for evaluating the distribution of power in a committee system". *American Political Science Review*, 48: pp. 787–792.

[47] Skibski, O., Michalak, T. P., Sakurai, Y. and Yokoo, M. (2015): "A pseudo-polynomial algorithm for computing power indices in graph-restricted weighted voting games". In: *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pp. 631–637.

[48] Spinrad, J. P. (2003): *Efficient graph representations*. American Mathematical Society, Providence.

[49] Tomita, E., Akutsu, T. and Matsunaga, T. (2011): "Efficient algorithms for finding maximum and maximal cliques and their applications". In: *Biomedical Engineering Trends in Electronics, Communications and Software*. Ed. by A. N. Laskovski. InTech, London, pp. 625–640.

[50] Tsukiyama, S., Ide, M., Ariyoshi, H. and Shirakawa, I. (1977): "A new algorithm for generating all the maximal independent sets". *Society for Industrial and Applied Mathematics Journal on Computing*, 6: pp. 505–516.

[51] Vadhan, S. P. (2001): "The complexity of counting in sparse, regular and planar graphs". *Society for Industrial and Applied Mathematics Journal on Computing*, 31: pp. 398–427.

[52] Valiant, L. G. (1979): "The complexity of enumeration and reliability problems". *Society for Industrial and Applied Mathematics Journal on Computing*, 8: pp. 410–421.

[53] Valiant, L. G. (2007): "Holographic algorithms". *Society for Industrial and Applied Mathematics Journal on Computing*, 37: pp. 1565–1594.

[54] Wagner, K. (1937): "Uber eine Eigenschaft der ebenen Komplexe". *Mathematische Annalen*, 114: pp. 570–590.

[55] Xia, M., Zhang, P. and Zhao, W. (2007): "Computational complexity of counting problems on 3-regular planar graphs". *Theoretical Computer Science*, 384: pp. 111–125.

# A. Line graphs of planar graphs

The following result is a particular case of Property 4.1, the proof of which does not appear in the original reference. The constructive proof below[22] is illustrated with an example in Figure 5.

Proposition A.1. If $G = (V, E)$ is a planar graph and it has no vertex of degree 4 or larger, then its line graph $L(G)$ is planar.

*Proof.* Let $\mathcal{G}$ be a planar embedding of $G$, that is, a representation of graph $G$ in the plane so that two edges intersect may only intersect each other at their endpoints. Due to Fáry's theorem ([23]), we can assume that all edges are drawn as straight lines (Figure 5a). Let $\delta$ be

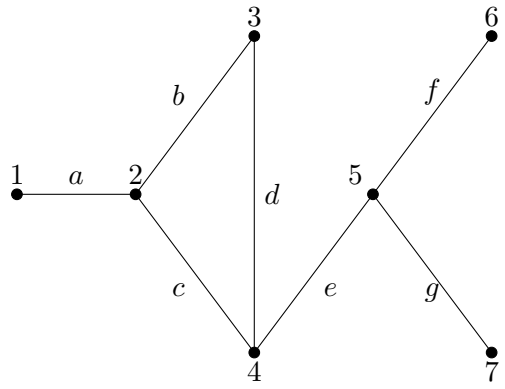
[22] Adapted from the sketch in `https://math.stackexchange.com/questions/4506448/if-deltag%e2%89%a43-then-planarity-of-g-implies-planarity-of-the-line-graph-lg` (retrieved November 3rd 2022)

the minimum distance between any two vertices of $G$ as drawn in $\mathcal{G}$, i.e. $\delta = \min\{||x - y|| : x, y \in V\}$. For each vertex $u \in V$, draw a circle $S_u$ of radius $\frac{\delta}{4}$ centered at $u$. For each edge $e \in E$ incident to $u$, create a new vertex $e_u$ where $S_u$ intersects $e$ and maintain the arcs of the circles as edges of a new graph $G^*$. Note that $\mathcal{G}$ induces a planar embedding of this new graph, call it $\mathcal{G}^*$ (Figure 5b).

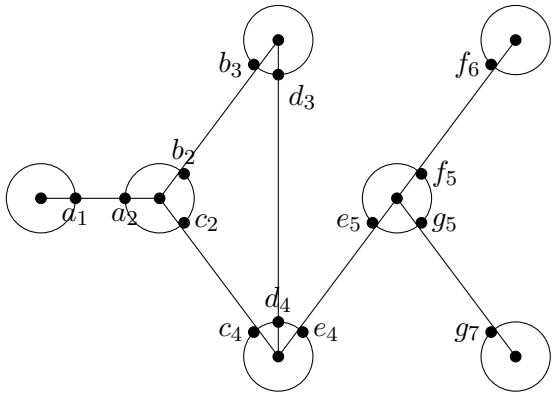
We claim that the line graph of $G$ is a minor of $G^*$. By the discussion preceding Property 2.5, this is enough to show that $L(G)$ itself is planar. Observe that $G^*$ has the same vertices as $G$ as well as two vertices per edge $e = \{u, v\} \in E$, $e_u$ and $e_v$. Moreover, by construction, each $e_u$ is adjacent to $e_v$ and $e'_u$ for all other edges $e' \in E$ incident to $u$, as well as the original vertex $u$ itself. Therefore, removing all vertices $u \in V$ from $\mathcal{G}^*$ (Figure 5c) and contracting each edge of the form $\{e_u, e_v\}$ while identifying its endpoints to a new vertex $\overline{e}$ (Figure 5d) results in a planar embedding of a graph with one vertex per edge of $G$, adjacent to one another if and only if they share an endpoint as edges of $G$. Removing any loops or multiple edges that may have appeared during the process (Figure 5e) will thus provide a planar embedding of the line graph of $G$. □

Figure 5: An example of the procedure described in Proposition A.1 to build the line graph of a planar graph with no vertex of degree larger than 3.

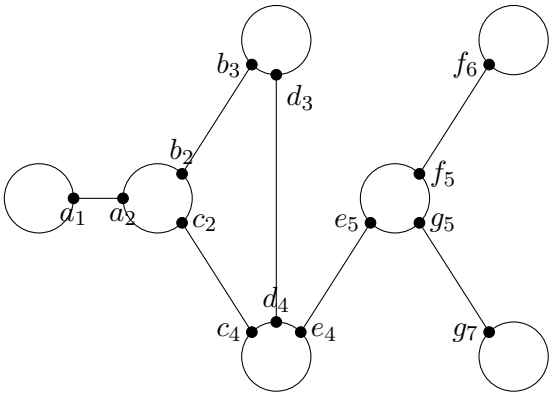
(a) Original planar embedding $\mathcal{G}$.



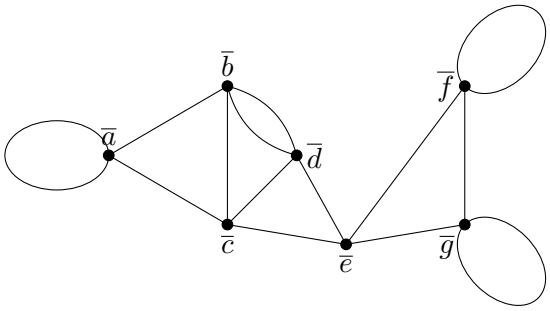

(b) Embedding $\mathcal{G}^*$.




(c) $\mathcal{G}^*$ after original vertices and their incident edges have been removed.




(d) Figure 5c after contracting edges $\{e_u, e_v\}$ for each edge $e = \{u, v\}$.




(e) Planar embedding of the line graph of $G$.