

Complexity of Unambiguous Problems in Σ_2^P

Matan Gilboa¹, Paul W. Goldberg¹, Elias Koutsoupias¹, Noam Nisan²

¹University of Oxford, UK

²Hebrew University of Jerusalem, Israel

Abstract

The complexity class Σ_2^P comprises problems based on polynomial-time checkable binary relations $\phi(x, y)$ in which we ask whether there exists x such that for all y , $\phi(x, y)$ holds. We let $\mathbf{US}_2^P$ denote the subclass of *unambiguous* problems in Σ_2^P , namely those whose yes-instances correspond with a *unique* choice of x . $\mathbf{US}_2^P$ is unlikely to have complete problems, but we identify various syntactic subclasses associated with general properties of ϕ that guarantee uniqueness. We use these to classify the complexity of problems arising in social choice and game theory, such as existence of (1) a dominating strategy in a game, (2) a Condorcet winner, (3) a strongly popular partition in hedonic games, and (4) a winner (source) in a tournament. We classify these problems, showing the first is Δ_2^P -complete, the second and third are complete for a class we term **PCW** (Polynomial Condorcet Winner), and the fourth for a class we term **PTW** (Polynomial Tournament Winner). We define another unambiguous class, **PMA** (Polynomial Majority Argument), seemingly incomparable to **PTW** and **PCW**. We show that with randomization, **PCW** and **PTW** coincide with Δ_2^P , and **PMA** is contained in Δ_2^P . Specifically, we prove: $\Delta_2^P \subseteq \mathbf{PCW} \subseteq \mathbf{PTW} \subseteq \mathbf{S}_2^P$, and $\mathbf{coNP} \subseteq \mathbf{PMA} \subseteq \mathbf{S}_2^P$ (and it is known that $\mathbf{S}_2^P \subseteq \mathbf{ZPP}^{\mathbf{NP}} \subseteq \Sigma_2^P \cap \Pi_2^P$). We demonstrate that unambiguity can substantially reduce computational complexity by considering ambiguous variants of our problems, and showing they are Σ_2^P -complete. Finally, we study the unambiguous problem of finding a weakly dominant strategy in a game, which seems not to lie in Σ_2^P .

Contents

1	Introduction	3
1.1	Related Literature	5
2	Preliminaries	6
3	The Class Polynomial Tournament Winner (PTW)	7
4	The Class Polynomial Condorcet Winner (PCW)	8
5	The Class Polynomial Majority Argument (PMA)	10
6	Unambiguous Problems Complete for Δ_2^P	11
7	Ambiguous Problem Variants	12
8	Beyond Σ_2^P: Weak Dominant Strategy and $U\exists\forall$-SAT	13
9	Discussion	14
	Appendix	15
A	Proofs of Section 3 (PTW)	15
B	Proofs of Section 4 (PCW)	19
C	Proofs of Section 5 (PMA)	26
D	Proofs of Section 6 (Δ_2^P)	32
E	Proofs of Section 7	35
F	Proofs of Section 8	37
G	Strong Popularity in ASHG is PCW-complete	38
G.1	Preliminaries	39
G.2	Setup of the Reduction	40
G.3	Condorcet string implies Strongly Popular Partition	43
G.4	Strongly Popular Partition implies Condorcet string	51

1 Introduction

Problems in **NP** are questions about the existence of solutions that can be efficiently checked, and problems in Σ_2^P ask about the existence of solutions that can be efficiently checked by an algorithm with an **NP** oracle. Such a decision problem is called *unambiguous* if it is guaranteed to have at most one solution. A starting-point of this paper is that in the case of Σ_2^P , unambiguous problems are more diverse than is the case for **NP**, and the classification of their computational complexity poses a new and interesting challenge. Examples include the existence of a dominating strategy in a game, a Condorcet winner in a voting scheme, a strongly popular partition in hedonic games, a winner (source) in a tournament, and many others. When trying to characterize the computational complexity of an unambiguous problem, this property may become an obstacle, since polynomial-time reductions—our main tool for such tasks—tend to be *parsimonious*, meaning they preserve the number of solutions between instances ([HB76, Val74, Sim77]). Hence, it is not clear how to reduce a problem that may admit multiple solutions to one that cannot, unless the former is significantly easier than the latter.

Unambiguous problems within **NP** (that is, those in **UP**) that are not known to belong to **P** are quite scarce. Notable examples include Prime Factorization, Parity Games, and Mean Payoff Games ([Jur98]). Interestingly, when allowing a second quantifier in the formulation of problems (moving “beyond **NP**”), unambiguity occurs more often. Our focus lies mainly in these problems, in particular on those contained in $U\Sigma_2^P$, the semantic¹ class containing all unambiguous problems in Σ_2^P .

Two of the most basic “beyond-**NP**” problems in $U\Sigma_2^P$ are UNIQUE-SAT (USAT) and TSP-UNIQUE-OPT. The former asks: “Given a Boolean formula ψ , does it admit a *unique* satisfying assignment?” Equivalently, it asks whether there exists an assignment $\mathbf{x}$ such that for all $\mathbf{y} \neq \mathbf{x}$ we have $\psi(\mathbf{x}) = 1$ and $\psi(\mathbf{y}) = 0$, illustrating why the problem lies in Σ_2^P but not necessarily in **NP**. [PY82] shows that USAT is contained in $\mathbf{D}^P$. Furthermore, several papers observe that it is **coNP**-hard. A proof for this can be found in [BG82], where they also show an oracle relative to which the problem is complete for $\mathbf{D}^P$, and another one relative to which it is not. The latter problem, TSP-UNIQUE-OPT, is the problem of determining whether a graph admits a unique optimal TSP tour. This can be expressed as the existence of a tour $\mathbf{x}$ that is shorter than any tour $\mathbf{y}$. It was the first problem shown to be Δ_2^P -complete, by Papadimitriou [Pap84].

These two problems share a unifying feature: Their unambiguity derives from the fact that solutions are defined as *winners* of a pairwise contest. In USAT, an assignment $\mathbf{x}$ “beats” another $\mathbf{y}$ if $\mathbf{x}$ satisfies the formula while $\mathbf{y}$ does not; in TSP-UNIQUE-OPT, one tour beats another if it is shorter. Thus, both problems ask whether there exists a winner who beats all other candidates according to some comparison rule, which induces a *tournament* among potential solutions. There can be at most one such winner because of the anti-symmetric nature of the “beat” relation ($\mathbf{y}$ cannot beat $\mathbf{x}$ if $\mathbf{x}$ beats $\mathbf{y}$).

As it turns out, many other problems can be formulated in a similar manner. One interesting example is the existence of a strictly dominant strategy in a game—one that yields a higher payoff than any other strategy of Player 1 against all opponent profiles. We show Δ_2^P -completeness of this problem, along with several other problems discussed in Section 6.

Note that the tournaments induced by USAT and TSP-UNIQUE-OPT are transitive, and hence

¹The distinction between syntactic complexity classes, which have complete problems, and semantic classes, which are unlikely to have complete problems, is discussed in [Pap94]. Syntactic classes are associated with machines having some recognizable structure, whereas semantic classes have problems solved by machines in which some (possibly undecidable) assumption is needed about how the machine behaves (for **UP**, the assumption is that a nondeterministic Turing machine has at most one accepting path for any input).

simpler and more structured than general tournaments. We further study several other tournament problems that seem to be harder than the above. One is based on the classical notion of Condorcet winners from social choice theory ([DC85]). Given a set of agents who each rank a set of possible outcomes, a Condorcet winner is an outcome that wins a majority vote against any other outcome, when the agents’ rankings are restricted to those two outcomes. The intransitive nature of majority comparisons (the *Condorcet paradox*, [DC85]) asserts that “winning cycles may be formed”, a property which seems to differentiate this problem from USAT and TSP-UNIQUE-OPT. When the number of voters is polynomial in the size of the input but the set of outcomes is exponential, we can model each voter as a Boolean circuit that takes a description of an outcome and computes a numerical evaluation of its merit. Then, comparing two outcomes is feasible but determining the existence of a Condorcet winner appears not to be.

One natural instance where this occurs is the existence of a *strongly popular partition* in hedonic games. In a hedonic game, an outcome is a partition of a set of agents into *coalitions*, and the agents have preferences over the partitions, an agent’s preferences depending on which other agents share their coalition. Introduced by [Gär75], a partition is called *strongly popular* if it wins a majority vote against any other partition. When agents’ preferences are additive, this problem is shown to be **coNP**-hard in [BB22], but its precise complexity is an open question that we resolve here.

To capture all problems of this tournament nature, we define the syntactic class **PTW** (Polynomial Tournament Winner), consisting of all problems that can be formulated as the question of existence of a winner in a tournament, whose edges can be queried in polynomial time by a Boolean circuit. The tournament is defined via edge queries to reflect the characteristic structure of these problems—exponentially large tournaments whose pairwise comparisons are efficiently computable.

Motivated by the Condorcet-winner problem, we further define a subclass of **PTW** that we call **PCW** (Polynomial Condorcet Winner). **PCW** captures tournament problems where comparison between vertices is obtained through a majority vote among a polynomial number of voters. We show that determining the existence of a strongly popular partition in an *additively separable hedonic game* (ASHG) is **PCW**-complete, resolving an open question by [BB22, BG25]. We further introduce a problem based on the notion of *intransitive dice*, which we show is also complete for **PCW**.

We have that $\mathbf{PTW} \subseteq \mathbf{US}_2^{\mathbf{P}}$: It is contained in $\mathbf{\Sigma}_2^{\mathbf{P}}$ because we ask whether there *exists* a vertex that beats *all* other vertices, and unambiguity follows from the impossibility of two vertices each defeating the other. We ask ourselves whether there exist other natural subclasses of $\mathbf{US}_2^{\mathbf{P}}$, namely other combinatorial reasons for unambiguity besides the anti-symmetric nature of a contest between two parties. Motivated by this question, we introduce another syntactic class that we call **PMA** (Polynomial Majority Argument). **PMA** captures unambiguous $\mathbf{\Sigma}_2^{\mathbf{P}}$ problems whose unambiguity is based on the principle that no two entities may simultaneously hold a majority of a shared commodity. More concretely, we syntactically construct a bipartite graph $G = (X, Y, E)$ whose number of edges is strictly less than $2|Y|$, and we ask whether there exists a vertex in X adjacent to all vertices in Y . Such a vertex would consume a majority of the edges, and thus this problem admits at most one solution.

The classes **PTW** and **PMA** seem to be quite robust, in that they capture at least all $\mathbf{US}_2^{\mathbf{P}}$ problems we have been able to come up with. To better understand the complexity of these problems, we relate **PTW** and **PMA** to other complexity classes. Are these unambiguous classes significantly easier than their immediate superclass $\mathbf{\Sigma}_2^{\mathbf{P}}$? Conversely, can we find any computational lower bounds for them? We answer these questions in the affirmative. Specifically, we prove that $\mathbf{\Delta}_2^{\mathbf{P}} \subseteq \mathbf{PCW} \subseteq \mathbf{PTW} \subseteq \mathbf{S}_2^{\mathbf{P}}$ and $\mathbf{coNP} \subseteq \mathbf{PMA} \subseteq \mathbf{S}_2^{\mathbf{P}}$.

The class $\mathbf{S}_2^{\mathbf{P}}$, introduced independently by [RS98] and [Can96], captures problems in which there is a polynomial-size witness for no-instances as well as yes-instances, and those witnesses are defined symmetrically (formally defined in Section 2). $\mathbf{S}_2^{\mathbf{P}}$ is shown in [Cai07] to be a subclass

of $\mathbf{ZPP}^{\mathbf{NP}}$. This means that, with randomization, we have that both $\mathbf{PTW}$ and $\mathbf{PMA}$ become subsets of $\Delta_2^{\mathbf{P}}$, while $\mathbf{PTW}$ is even equal to $\Delta_2^{\mathbf{P}}$ (as $\Delta_2^{\mathbf{P}}$ is also a subset of $\mathbf{PTW}$). These result are the first to show any non-trivial upper bound on broad classes of unambiguous problems. A summary of our results regarding the relation of our proposed complexity classes and existing ones is depicted in Figure 1.

Next, we address the question of how pivotal is the role of unambiguity in the computational complexity of problems. To give a partial answer to this question, we look into some of the unambiguous problems in this paper, and make small changes to eliminate the unambiguity property. We show that these variations turn out to be $\Sigma_2^{\mathbf{P}}$ -complete, significantly harder than their unambiguous counterparts. We exemplify this with problems we denote $\mathbf{CKT-UNIQUE-VALUE}$ and $\mathbf{2-CKT-PARETO}$, which are variations of $\mathbf{CKT-UNIQUE-OPT}$ and $\mathbf{CKT-CONSENSUS}$ (defined in Section 6), respectively. Another example of this phenomenon can be found in hedonic games: Although we show that strong popularity in additive hedonic games is $\mathbf{PCW}$ -complete, in [BG25] it is shown that weak popularity (where a partition needs only to weakly win a majority vote against any other, and thus a winner is not necessarily unique) is $\Sigma_2^{\mathbf{P}}$ -complete.

Finally, we consider the problem of determining the existence of a weakly-dominant strategy in a game, and show it is computationally equivalent to the two-quantified version of $\mathbf{USAT}$. These are the only two problems in this work that seem not to lie within $\Sigma_2^{\mathbf{P}}$. Specifically, we prove that they are positioned between $\Pi_2^{\mathbf{P}}$ and $\mathbf{D}_2^{\mathbf{P}}$, where $\mathbf{D}_2^{\mathbf{P}}$ (defined in [ACHI17]) is a generalization of the class $\mathbf{D}^{\mathbf{P}}$ (defined in [PY82]). These results support the view that the hardness of unambiguous problems does not typically reflect the full potential of their immediate upper-bounding class—in this case, $\Sigma_3^{\mathbf{P}}$.

The rest of this paper is organized as follows. Following a brief account of related literature, in Section 2 we introduce some relevant notation and definitions. Sections 3 to 5 introduce the classes $\mathbf{PTW}$, $\mathbf{PCW}$, and $\mathbf{PMA}$ respectively, discuss their relation with other complexity classes, and problems that are captured by them. In Section 6 we exhibit several $\Delta_2^{\mathbf{P}}$ -complete unambiguous problems. Section 7 considers ambiguous variants of unambiguous problems, and shows they are $\Sigma_2^{\mathbf{P}}$ -complete. All proofs are deferred to the appendix.

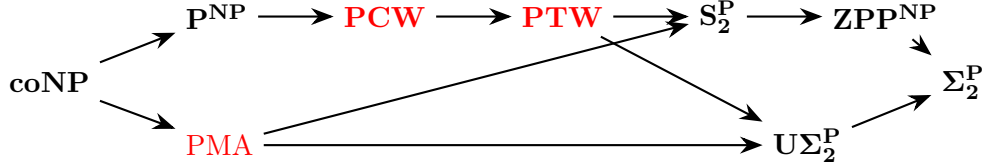


Figure 1: Our proposed complexity classes (in red) relative to known subclasses of $\Sigma_2^{\mathbf{P}}$.

1.1 Related Literature

We conclude this introduction by briefly highlighting several related studies, in addition to the works discussed throughout the paper. An interesting related line of work concerns the class $\mathbf{TF}\Sigma_2^{\mathbf{P}}$ of total function in $\Sigma_2^{\mathbf{P}}$. Unlike $\mathbf{U}\Sigma_2^{\mathbf{P}}$ where problems are guaranteed to have at most one solution, $\mathbf{TF}\Sigma_2^{\mathbf{P}}$ problems are guaranteed to have at least one solution. A prominent problem in $\mathbf{TF}\Sigma_2^{\mathbf{P}}$ which has gained much attention in recent years is the range avoidance problem, or $\mathbf{AVOID}$ ([KKMP21]). Interestingly, $\mathbf{AVOID}$ is shown in [CHR24, Li24] to be included in a single-valued, functional variant of $\mathbf{S}_2^{\mathbf{P}}$, resonating with our results that $\mathbf{PTW}$ and $\mathbf{PMA}$ are included in $\mathbf{S}_2^{\mathbf{P}}$. We refer to [Kor25] for a recent survey on range avoidance.

An interesting result by Stearns [Ste59] shows that, given a tournament T on n vertices, $\theta(\frac{n}{\log n})$ voters may be required to induce a majority graph which exactly coincides with T . A related result by Erdős [EM64] shows that $\theta(\frac{n}{\log n})$ voters also suffice to induce any tournament. The former result may hint towards a separation between the classes **PTW** and **PCW**.

In Section 4 we address computational aspects of intransitive dice. The literature typically approaches them from the perspectives of combinatorics, probability theory, and game theory. Intransitive dice first appeared in [Gar70], which credits Bradley Efron for a set of three dice, known as Efron’s dice, that form a winning cycle. In [Aki20, Sch17], it is shown that every tournament on n vertices can be induced by a set of dice using the “beat” relation. [Sch17] further show that for any $n, m \geq 3$, there exists a set of n dice, each with m faces, that form an n -cycle, a result later rediscovered by [CFL⁺23]. [CFL⁺23] study a random model where the dice’s faces are random variables. [HW19] consider a game where two agents simultaneously construct an n -faced die, and then compare them; they show it admits a unique pure Nash equilibrium. [CH20] study the probability of obtaining an intransitive tournament when randomly constructing four dice.

Numerous works have discussed unambiguous computation. The class **UP** was introduced in [Val76], and has since then been tied with foundational questions in cryptography. It is shown in [GS88] that one-way functions exist if and only if **P** $\neq$ **UP**. The famous Valiant-Vazirani theorem establishes that **Promise** – **UP** is as hard as **NP**, up to randomization ([VV85]). In [LR94], three seemingly different hierarchies of unambiguous classes were introduced and characterized using Boolean circuits of exponential size with suitable restrictions on gate behavior. [NR98] provides an alternative characterization of **UP** and introduces the class **UAP**, defined via local functions on the nodes of a Turing machine’s computation tree. While the latter two papers focus on the structural theory of unambiguous computation, our interest lies in concrete Σ_k^P problems with inherent uniqueness properties, resulting in a different unambiguous framework.

2 Preliminaries

Given a directed graph $G = (V, E)$ and $v_1, v_2 \in V$, we say v_1 beats v_2 if $(v_1, v_2) \in E$ while $(v_2, v_1) \notin E$ (namely, an outgoing edge is considered a win in a contest between the vertices). If both edges are present or neither of them are, we normally say there is a tie between the vertices. We say G is a *weak tournament* if for every $v_1, v_2 \in V$, at least one of the edges (v_1, v_2) or (v_2, v_1) is present in E . We say it is a *tournament* if exactly one of those edges is present in E for every pair of vertices.

We let Σ be a fixed, finite alphabet; without loss of generality $\Sigma = \{0, 1\}$. Σ^* is then the set of all finite strings over $\{0, 1\}$. We define $\mathbf{US}_k^P$ as follows.

Definition 2.1. Let $k \geq 1$. A language $L \subseteq \Sigma^*$ is in $\mathbf{US}_k^P$ if there is a polynomial-time Turing machine M and a polynomial p such that for all $w \in \Sigma^*$ we have:

- $w \in L \iff \exists x_1 \forall x_2 \exists x_3 \dots Q x_k M(w, x_1, \dots, x_k) = 1$, and
- there exists at most one x_1 such that $\forall x_2 \exists x_3 \dots Q x_k M(w, x_1, \dots, x_k) = 1$

where Q alternates between $\exists$ and $\forall$, and for all i we have $x_i \in \Sigma^*$ and $|x_i| \leq p(|w|)$.

Using the dual characterization of the polynomial hierarchy ([Sto76, Wra76]), it may be shown that an equivalent definition is $\mathbf{US}_k^P = \mathbf{UP}^{\Sigma_{k-1}^P}$ for all $k \geq 1$. We informally say a problem is *unambiguous* if some argument shows it can have at most one solution. Given problems A and B , $A \leq_P B$ means A is many-one reducible to B in polynomial time. $\mathbb{N}$ is the set of positive integers,

and $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. Given two bit strings $\mathbf{x}$ and $\mathbf{y}$ we write $\mathbf{xy}$ or $\mathbf{x} \parallel \mathbf{y}$ for their concatenation. We often interpret bit strings as binary numbers, and use operations like $\mathbf{x} < \mathbf{y}$.

Definition 2.2. ([RS98, Can96]). A language $L \subseteq \Sigma^*$ is in $\mathbf{S}_2^P$ if there is a polynomial-time predicate P such that for all $w \in \Sigma^*$ we have:

- $w \in L \implies \exists \mathbf{x} \forall \mathbf{y} P(w, \mathbf{x}, \mathbf{y}) = 1$, and
- $w \notin L \implies \exists \mathbf{y} \forall \mathbf{x} P(w, \mathbf{x}, \mathbf{y}) = 0$

where $|\mathbf{x}|$ and $|\mathbf{y}|$ are bounded by a polynomial function of $|w|$.

It is straightforward (by predicate composition) that $\mathbf{US}_2^P$ and $\mathbf{S}_2^P$ are closed under polynomial-time many-one reductions. The same holds for the classes $\mathbf{PTW}$, $\mathbf{PCW}$, and $\mathbf{PMA}$ defined in Sections 3 to 5. Hence, to show that a class $\mathbf{A}$ is contained in a class $\mathbf{C} \in \{\mathbf{US}_2^P, \mathbf{S}_2^P, \mathbf{PTW}, \mathbf{PCW}, \mathbf{PMA}\}$, it suffices to exhibit an $\mathbf{A}$ -complete language L such that $L \in \mathbf{C}$.

3 The Class Polynomial Tournament Winner (PTW)

In this section, we introduce a class denoted Polynomial Tournament Winner ($\mathbf{PTW}$). $\mathbf{PTW}$ is a subclass of $\mathbf{US}_2^P$ which aims to describe problems of a “tournament” nature, where we seek a winner that beats all other participants under a polynomial-time computable criterion. This class contains all unambiguous $\mathbf{S}_2^P$ problems discussed in this paper, apart from $\mathbf{EDGE-MAJORITY}$ and its variations, discussed in Section 5. We define $\mathbf{PTW}$ via a problem we call $\mathbf{WEAK-TOURNAMENT-SOURCE}$. $\mathbf{WEAK-TOURNAMENT-SOURCE}$ is the problem of determining the existence of a source in an exponentially large weak tournament. It is described through a Boolean circuit which compares pairs of vertices in polynomial time, and the question is whether there exists a vertex that beats all other vertices, which immediately places the problem in $\mathbf{S}_2^P$.

Problem 3.1:

$\mathbf{WEAK-TOURNAMENT-SOURCE}$

Input: Boolean circuit $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x} \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) = 0$?

Definition 3.1. The class Polynomial Tournament Winner ($\mathbf{PTW}$) is defined by

$$\mathbf{PTW} = \{\mathbf{L} \subseteq \Sigma^*: \mathbf{L} \leq_P \mathbf{WEAK-TOURNAMENT-SOURCE}\}.$$

$\mathcal{C}$ induces a weak tournament where, for all distinct $\mathbf{x}, \mathbf{y}$, the edge $(\mathbf{x}, \mathbf{y})$ is present if $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) = 0$, and both $(\mathbf{x}, \mathbf{y})$ and $(\mathbf{y}, \mathbf{x})$ are present if $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{y} \parallel \mathbf{x})$. Thus, a solution corresponds to a source in the induced tournament. Indeed, $\mathbf{WEAK-TOURNAMENT-SOURCE}$ is unambiguous, and so $\mathbf{PTW} \subseteq \mathbf{US}_2^P$. This is because given two vertices, either they tie against each other or one of them beats the other, and so either way they cannot both be sources.

An important result of this section is that $\mathbf{PTW} \subseteq \mathbf{S}_2^P$, which implies that $\mathbf{PTW} \subseteq \mathbf{ZPP}^{\mathbf{NP}}$, since $\mathbf{S}_2^P \subseteq \mathbf{ZPP}^{\mathbf{NP}}$ ([Cai07]). An interesting step along the way of our proof shows that finding a source in a weak tournament is computationally equivalent to finding one in a tournament ($\mathbf{TOURNAMENT-SOURCE}$), as shown in Theorem 3.2.

Problem 3.2:

$\mathbf{TOURNAMENT-SOURCE}$

Input: Boolean circuit $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x} (\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) = 0) \vee (\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) \wedge \mathbf{x} > \mathbf{y})$?

$\mathcal{C}$ induces a tournament where the edge (x, y) is present if and only if $(\mathcal{C}(x \parallel y) = 1 \wedge \mathcal{C}(y \parallel x) = 0) \vee (\mathcal{C}(x \parallel y) = \mathcal{C}(y \parallel x) \wedge x > y)$ (we use their numeric values as a tie breaker to ensure it describes a tournament rather than a weak one). Once again, a solution corresponds to a source. The following two theorems establish the main results of the section.

Theorem 3.2. *TOURNAMENT-SOURCE is $\mathbf{PTW}$ -complete.*

Theorem 3.3. $\mathbf{PTW} \subseteq \mathbf{S}_2^P$.

The containment of $\mathbf{PTW}$ in $\mathbf{S}_2^P$ provides a non-trivial upper bound for several problems, such as ASHG-STRONG-POPULARITY and GRAPH-DICE, discussed in Section 4. In Section 4 we define the subclass $\mathbf{PCW}$ of $\mathbf{PTW}$ for which these problems are complete. We further show there that $\Delta_2^P \subseteq \mathbf{PCW}$, thus also providing a lower bound for $\mathbf{PTW}$.

To illustrate the robustness of the definition of $\mathbf{PTW}$, we introduce a generalization of WEAK-TOURNAMENT-SOURCE which we call MULTI-WEAK-TOURNAMENT-SOURCE, and show that it is $\mathbf{PTW}$ -complete. It generalizes WEAK-TOURNAMENT-SOURCE by adding a third input string z for the circuit; ignoring this input recovers an instance of WEAK-TOURNAMENT-SOURCE. Intuitively, each z induces a weak tournament T_z , and we seek a vertex that serves as a source across all such tournaments.

Problem 3.3:

MULTI-WEAK-TOURNAMENT-SOURCE

Input: Boolean circuit $\mathcal{C}: \{0, 1\}^{3n} \rightarrow \{0, 1\}$.

Question: $\exists x \in \{0, 1\}^n \forall y, z \in \{0, 1\}^n$ with $y \neq x$ $\mathcal{C}(x \parallel y \parallel z) = 1 \wedge \mathcal{C}(y \parallel x \parallel z) = 0$?

Theorem 3.4. *MULTI-WEAK-TOURNAMENT-SOURCE is $\mathbf{PTW}$ -complete.*

4 The Class Polynomial Condorcet Winner ($\mathbf{PCW}$)

In this section, we introduce a subclass of $\mathbf{PTW}$, termed Polynomial Condorcet Winner ($\mathbf{PCW}$), inspired by the notion of a Condorcet winner ([DC85]). This class captures tournament problems in which the underlying tournament is induced by pairwise majority comparisons among candidates, determined by a polynomial-size set of voters. Each voter is represented by a Boolean circuit that succinctly encodes the voter's preferences. Given a bit string representing a candidate, each circuit outputs a numerical value, thereby inducing a weak total order over the preferences. The question is whether there exists a candidate who defeats every other candidate in a pairwise majority vote—namely, a Condorcet winner. To formalize this notion, we define the problem CKT-CONDORCET, and we define $\mathbf{PCW}$ as the class of problems polynomial-time reducible to it.

Problem 4.1:

CKT-CONDORCET

Input: A sequence of Boolean circuits $\mathcal{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$, each with n inputs and n outputs.

Question: $\exists x \in \{0, 1\}^n \forall y \in \{0, 1\}^n$ with $y \neq x$ $\sum_{i=1}^m (\text{sgn}(\mathcal{C}_i(x) - \mathcal{C}_i(y))) > 0$?

A solution is called a *Condorcet string*.

Definition 4.1. The class Polynomial Condorcet Winner ($\mathbf{PCW}$) is defined by

$$\mathbf{PCW} = \{\mathbf{L} \subseteq \Sigma^*: \mathbf{L} \leq_P \text{CKT-CONDORCET}\}.$$

We establish lower and upper computational bounds on $\mathbf{PCW}$, formalized in the following theorems.

Theorem 4.2. $\mathbf{PCW} \subseteq \mathbf{PTW}$.

Theorem 4.3. $\Delta_2^P \subseteq \mathbf{PCW}$.

Our bounds can be extended to the parameterized version of CKT-CONDORCET, where the number of input circuits is a fixed parameter k . For $k = 1$ and $k = 2$, the resulting problems are shown to be Δ^P -complete in Section 6 (when $k = 1$ we obtain the problem CKT-UNIQUE-OPT, defined in Section 6). For any $k \geq 3$, we do not know the problem's exact complexity, but given the results for $k = 1$ and $k = 2$ it is naturally Δ^P -hard and in $\mathbf{PCW}$.

Problem 4.2:

CKT-CONDORCET[k]

Parameter: A fixed, positive integer k .

Input: Sequence of k Boolean circuits $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_k \rangle$, each with n inputs and n outputs.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x} \ \sum_{i=1}^k (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_i(\mathbf{y}))) > 0$?

Theorem 4.4. Fix $k \geq 1$. We have that:

1. For all $k' > k$, we have that $\text{CKT-CONDORCET}[k] \leq_P \text{CKT-CONDORCET}[k']$.
2. $\text{CKT-CONDORCET}[k] \in \mathbf{PCW}$.
3. $\text{CKT-CONDORCET}[k]$ is Δ_2^P -hard.

Notice that in Papadimitriou's Δ_2^P -complete problem TSP-UNIQUE-OPT (and, indeed, in several other Δ_2^P -complete problems discussed in Section 6) there exists a total order over the possible solutions, and we search for a solution with a value lower than any other. In the Condorcet winner problem, however, we do not generally have this convenient structure, as solutions may beat one another in a cyclic manner. Therefore, $\mathbf{PCW}$ provides us with a tool to examine intransitive problems where solutions are not necessarily ordered², and thus do not easily fall into Δ_2^P .

We proceed to exhibit several natural problems that are complete for $\mathbf{PCW}$, beginning with ASHG-STRONG-POPULARITY. This problem asks whether an *additively separable hedonic game* (ASHG) admits a *strongly popular* partition. In ASHG, agents must be partitioned into coalitions, and each agent's utility is the sum of her valuations for the other agents in her coalition. A partition is strongly popular if it wins a majority vote against any other partition. Determining the existence of such a partition was shown to be $\mathbf{coNP}$ -hard in [BB22]. The question of its precise complexity was raised in [BB22, BG25]. We resolve this question by proving that the problem is $\mathbf{PCW}$ -complete. This proof is the most technically involved in our work, and we therefore devote a separate section for it (Section G), where a formal definition of the problem is also provided.

Theorem 4.5. ASHG-STRONG-POPULARITY is $\mathbf{PCW}$ -complete.

We note that, since $\mathbf{coNP} \subseteq \Delta_2^P$, Theorems 4.3 and 4.5 together subsume the result in [BB22] that ASHG-STRONG-POPULARITY is $\mathbf{coNP}$ -hard.

We proceed with another problem whose solutions are intransitive, which we term GRAPH-DICE, and its Boolean-circuit generalization CKT-DICE. The motivation for these problems originates in the well-studied mathematical concept of *intransitive dice* (see Section 1.1). Suppose we have a set of dice, each with m faces, and on each face is a number. Assume that when we roll a die there is a

²While intransitivity hints as to why problems are difficult to place in Δ_2^P , it does not immediately imply they are not in Δ_2^P . For instance, the solutions of CKT-CONDORCET[2] do not necessarily induce a total order, but we show it is Δ_2^P -complete (Theorem 6.1). To see why, consider two voters v_1 and v_2 and four candidates a, b, c , and d . Voter v_1 assigns the values (4, 4, 3, 5) and voter v_2 assigns (2, 1, 5, 3) to (a, b, c, d) , respectively. It may be verified that $a \succ b \approx c \approx d \succ a$ by majority vote.

probability of $\frac{1}{m}$ to roll each of its faces. We say die x *beats* die y if the probability³ that x rolls a higher number than y is greater than the probability that y rolls a higher number than x . Similarly to popularity, the dice game is also not transitive; that is, we may have a set of dice that beat each other in a cyclic manner.

Consider thus the problem of deciding whether there exists a die that beats all others in a given a set of dice. When the set is small, the answer to this question can be easily checked by iterating over all pairs of dice and performing simple probability calculations (we assume the number of faces on each die is small). However, when the set of dice is large this becomes an interesting computational problem. We introduce two formulations for this problem. In the more general formulation, CKT-DICE, we let n -bit strings represent dice labels, and their faces are generated by the outputs of a collection of Boolean circuits whose input is the die label. The second formulation, GRAPH-DICE, is more natural in that it is defined on a directed graph with weighted edges. Each partition π of the vertices induces a die, where each vertex v corresponds to a face; the value of that face is the weighted sum of outgoing edges from v to other vertices in the set v belongs to by π . We will show that, for both formulations, determining the existence of a winning die is **PCW**-complete.

Problem 4.3:

CKT-DICE

Input: Sequence of Boolean circuits $\mathcal{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$, each with n inputs and n' outputs.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x} \ \sum_{i=1}^m \sum_{j=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_j(\mathbf{y}))) > 0$?

We interpret $\mathcal{C}_i(x)$ as the value on the i 'th face of die $x \in \{0, 1\}^n$, as signed integers written in binary.

Problem 4.4:

GRAPH-DICE

Input: Directed graph $G = (V, w)$, where $|V| = n$ and w is an edge-weight function $w : V \times V \rightarrow \mathbb{R}$ on all edges, including self loops.

Question: $\exists \text{partition } \pi^* \ \forall \text{partition } \pi' \text{ with } \pi' \neq \pi^* \ \sum_{i=1}^n \sum_{j=1}^n (\text{sgn}(s_i(\pi^*) - s_j(\pi'))) > 0$?

Where $\pi(i)$ denotes the set to which vertex i belongs in the partition π , and $s_i(\pi) := \sum_{l \in \pi(i)} w_i(l)$.

Theorem 4.6. *GRAPH-DICE and CKT-DICE are PCW-complete.*

5 The Class Polynomial Majority Argument (PMA)

In this section we introduce another subclass of $\mathbf{US}_2^{\mathbf{P}}$, termed Polynomial Majority Argument (**PMA**). This class captures unambiguous problems in $\Sigma_2^{\mathbf{P}}$ whose unambiguity follows from a majority argument, intuitively be described as follows. Suppose we are given a directed bipartite graph $G = (X, Y, E)$ with edges only from X to Y , where the existence of edges can be efficiently queried by a Boolean circuit. Assume further that the total number of edges in the graph is strictly less than $2|Y|$ —a property that we will soon show can be syntactically enforced. We then ask whether there exists a vertex $x \in X$ such that for every $y \in Y$, $(x, y) \in E$. This problem is unambiguous since two vertices cannot both claim ownership for a majority of the edges. Formally:

Problem 5.1:

EDGE-MAJORITY

Input: Two Boolean circuits $\mathcal{C}: \{0, 1\}^{n+m} \rightarrow \{0, 1\}^{m+1}$ and $\mathcal{V}: \{0, 1\}^{m+1} \rightarrow \{0, 1\}^{n+m}$, and integer $k \geq 1$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^m \ \mathcal{V}(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) = \mathbf{x} \mathbf{y} \ \wedge \ \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \geq k$?

³In the literature often a probability greater than $\frac{1}{2}$ is required.

Definition 5.1. The class Polynomial Majority Argument (**PMA**) is defined by

$$\mathbf{PMA} = \{\mathbf{L} \subseteq \Sigma^* : \mathbf{L} \leq_{\mathbf{P}} \text{EDGE-MAJORITY}\}.$$

We interpret $\{l \in \{0, 1\}^{m+1} : l \geq k\}$ as the set of admissible edge-labels. The circuit $\mathcal{V}$ (for “verifier”) ensures that edges are correctly mapped in the reverse direction—from the label $\mathcal{C}(\mathbf{x} \parallel \mathbf{y})$ back to the vertex pair $(\mathbf{x}, \mathbf{y})$. Since there are only $2^{m+1} - k$ admissible labels, this enforces an upper bound on the number of possible edges in graph. Moreover, as evident from its syntactic form, EDGE-MAJORITY is clearly in $\Sigma_2^{\mathbf{P}}$.

Theorem 5.2. $\mathbf{PMA} \subseteq \mathbf{US}_2^{\mathbf{P}}$.

We next establish computational bound for **PMA**. Similarly to **PTW**, we show that **PMA** is a subset of $\mathbf{S}_2^{\mathbf{P}}$, implying that all unambiguous problems considered in this work are upper-bounded by $\mathbf{S}_2^{\mathbf{P}}$. As a lower bound, we prove that **PMA** contains **coNP**, providing evidence that this class indeed contains hard problems. In contrast to **PTW**, however, it remains unclear whether **PMA** contains $\Delta_2^{\mathbf{P}}$, or even **NP**, suggesting some interesting open questions.

Theorem 5.3. $\mathbf{PMA} \subseteq \mathbf{S}_2^{\mathbf{P}}$.

Theorem 5.4. $\mathbf{coNP} \subseteq \mathbf{PMA}$.

We also consider several natural variants of EDGE-MAJORITY. First, we define a parameterized variant, denoted EDGE-MAJORITY-BALANCED[k], in which the threshold of disqualified labels k is a fixed parameter, and the bipartite graph is balanced (i.e., $|X| = |Y|$). Surprisingly, for every fixed $k \geq 1$ EDGE-MAJORITY-BALANCED[k] remains **PMA**-complete. This result naturally extends to variants imposing only one of these restrictions (either balance or fixed k). Furthermore, if the input specifies an explicit set of disqualified labels instead of a threshold, the problem remains **PMA**-complete.

Problem 5.2:

EDGE-MAJORITY-BALANCED[k]

Parameter: A fixed integer $k \geq 1$.

Input: Two Boolean circuits $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}^{n+1}$ and $\mathcal{V}: \{0, 1\}^{n+1} \rightarrow \{0, 1\}^{2n}$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^n \ \mathcal{V}(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) = \mathbf{xy} \ \wedge \ \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \geq k$?

Problem 5.3:

EDGE-MAJORITY-SET

Input: Two Boolean circuits $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}^{n+1}$ and $\mathcal{V}: \{0, 1\}^{n+1} \rightarrow \{0, 1\}^{2n}$, and a set $\emptyset \neq S \subseteq \mathbb{N}_0$ given explicitly as a list of binary numbers.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^n \ \mathcal{V}(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) = \mathbf{xy} \ \wedge \ \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \notin S$?

Theorem 5.5. Let $k \geq 1$. EDGE-MAJORITY-BALANCED[k] is **PMA**-complete.

Theorem 5.6. EDGE-MAJORITY-SET is **PMA**-complete.

It seems non-trivial to establish whether either of the classes **PMA** or **PTW** is contained in the other. We conjecture that they are incomparable, as their unambiguity relies on fundamentally distinct combinatorial principles.

6 Unambiguous Problems Complete for $\Delta_2^{\mathbf{P}}$

In this section, we consider several unambiguous problems that are complete for $\Delta_2^{\mathbf{P}}$ ($= \mathbf{P}^{\mathbf{NP}}$), the class of problems that are Turing reducible to SAT. Since $\Delta_2^{\mathbf{P}} \subseteq \mathbf{PCW} \subseteq \mathbf{PTW}$ (see Theorems 4.2

and 4.3), these problems seem easier than other **PTW** problems discussed above. Specifically, we prove Δ_2^P -completeness of **CKT-CONDORCET**[2] (defined in Section 4), and the following three problems.

Problem 6.1:

STRONG-DOMINANT-STRATEGY

Input: Boolean circuit $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}^n$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{x}', \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{x} \neq \mathbf{x}'$ $\mathcal{C}(\mathbf{x}, \mathbf{y}) > \mathcal{C}(\mathbf{x}', \mathbf{y})$?

Problem 6.2:

CKT-CONSENSUS

Input: Sequence of Boolean circuits $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$, each with n inputs and n outputs.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{x}' \in \{0, 1\}^n$ with $\mathbf{x}' \neq \mathbf{x} \forall i \in \{1, \dots, m\}$ $\mathcal{C}_i(\mathbf{x}) > \mathcal{C}_i(\mathbf{x}')$?

Problem 6.3:

CKT-WINNER-THRESHOLD

Input: A Boolean circuit $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{x}', \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{x}' \neq \mathbf{x}$ $\mathcal{C}(\mathbf{x}, \mathbf{y}) = 1 \wedge (\mathbf{x}' > \mathbf{x} \implies \mathcal{C}(\mathbf{x}', \mathbf{y}) = 0)$?

In **STRONG-DOMINANT-STRATEGY**, we interpret $\mathcal{C}$ as formulating a game where $\mathbf{x}$ describes the strategy of Player 1, $\mathbf{y}$ describes the strategies of the other players, and $\mathcal{C}$'s output is the utility of Player 1. The question then is whether there exists a strongly dominant strategy for Player 1. We refer to Section 8 for results on finding a weak-dominant strategy in a game.

CKT-CONSENSUS asks for a string achieving optimal value in all given circuits. This can be seen as a circuit-based generalization of **TSP-UNIQUE-OPT**, where several sets of weights (each encoded by a different circuit) are given for the same graph and we seek a TSP tour (encoded by the input $\mathbf{x}$) that is uniquely optimal with respect to all sets of weights.

CKT-WINNER-THRESHOLD asks whether there is an $\mathbf{x}$ that is an all-winner (it satisfies ψ regardless of $\mathbf{y}$) but it is also a threshold, in that any string greater than $\mathbf{x}$ is an all-loser (it does not satisfy ψ regardless of $\mathbf{y}$). Notice that it differs from the other three problems in that the circuit's output is in $\{0, 1\}$, whereas the other problems' circuits output strings. Thus, hardness for **CKT-WINNER-THRESHOLD** stems only from the size of the input space of ψ , while in the other problems it also relates to the size of the output space of the circuits, yielding different-flavored reductions.

CKT-CONDORCET[2] is the Condorcet-winner existence problem with two voters. It is an interesting example of a problem that, despite having intransitive solutions (see Footnote 2) resembling the **PCW**-complete problems discussed in Section 4, is included in Δ_2^P .

The results of this section are summarized in the following theorem.

Theorem 6.1. *The problems **STRONG-DOMINANT-STRATEGY**, **CKT-CONSENSUS**, **CKT-WINNER-THRESHOLD**, and **CKT-CONDORCET**[2] are Δ_2^P -complete.*

7 Ambiguous Problem Variants

In the previous sections, we have seen many examples of unambiguous problems in Σ_2^P that are relatively easy compared with the full potential of Σ_2^P . In this section, we consider slight variations of previously discussed problems, which are ambiguous. We exhibit the idea that unambiguity seems to play an important role in diminishing the computational complexity of problems, by showing that their ambiguous variations yield Σ_2^P -complete problems. One such example can be found in [BG25], where it is shown that Weak-Popularity in additive hedonic games (a version of **ASHG-STRONG-POPULARITY** defined with a weak inequality) is Σ_2^P -complete, in contrast to our result of Theorem 4.5.

We provide two additional such results, showing Σ_2^P -completeness of the problems CKT-UNIQUE-VALUE and 2-CKT-PARETO, defined below. Indeed, it may be observed that CKT-UNIQUE-VALUE differs from CKT-UNIQUE-OPT only in that $>$ becomes $\neq$ (instead of asking about the existence of a string obtaining a value *greater* than any other string, we ask about obtaining a value *different* than any other string); and 2-CKT-PARETO differs from CKT-CONSENSUS (with 2 circuits) only in that the condition changes from $\wedge$ to $\vee$ (we seek a string which, against any other string, wins in either one of the circuits, rather than in both circuits). A solution for 2-CKT-PARETO is Pareto optimal in the sense that any string that challenges it would reduce the value of at least one circuit.

Problem 7.1:

CKT-UNIQUE-VALUE

Input: Boolean circuit $\mathcal{C}: \{0, 1\}^n \rightarrow \{0, 1\}^n$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x}$ $\mathcal{C}(\mathbf{x}) \neq \mathcal{C}(\mathbf{y})$?

Problem 7.2:

2-CKT-PARETO

Input: Two Boolean circuits $\mathcal{C}_1: \{0, 1\}^n \rightarrow \{0, 1\}^n$ and $\mathcal{C}_2: \{0, 1\}^n \rightarrow \{0, 1\}^n$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x}$ $\mathcal{C}_1(\mathbf{x}) > \mathcal{C}_1(\mathbf{y}) \vee \mathcal{C}_2(\mathbf{x}) > \mathcal{C}_2(\mathbf{y})$?

Theorem 7.1. CKT-UNIQUE-VALUE is Σ_2^P -complete.

Theorem 7.2. 2-CKT-PARETO is Σ_2^P -complete.

8 Beyond Σ_2^P : Weak Dominant Strategy and $U\exists\forall$ -SAT

In this section, we turn our attention to the problem of determining the existence of a weakly-dominant strategy in a game. There, we seek a strategy s_1^* for Player 1 such that for any other strategy s_1' , we have that s_1^* is weakly better than s_1' against any strategy profile of the other players, and is strictly better than s_1' against *some* strategy profile of the other players. We formalize this problem as follows.

Problem 8.1:

WEAK-DOMINANT-STRATEGY (WDOM-STRATEGY)

Input: Boolean circuit $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}^n$.

Question: $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{x}', \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{x} \neq \mathbf{x}' \exists \mathbf{y}' \in \{0, 1\}^n$ $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \geq \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}) \wedge \mathcal{C}(\mathbf{x} \parallel \mathbf{y}') > \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}')$?

As in STRONG-DOMINANT-STRATEGY, we interpret $\mathcal{C}$ as describing a game in which $\mathbf{x}$ specifies the strategy of Player 1, $\mathbf{y}$ specifies the joint strategies of the remaining players, and the circuit's output represents Player 1's utility. An inspection of the problem's definition shows it is unambiguous. However, unlike all previously discussed problems, it seems improbable that it is included in Σ_2^P . Indeed, as suggested by the quantifier structure in its definition, its complexity seems to lie somewhere between the second and third levels of the polynomial hierarchy.

Supporting this intuition, we first show that WDOM-STRATEGY is many-one equivalent to $U\exists\forall$ -SAT ("Is there a unique satisfying assignment in a two-quantified Boolean formula?"). We then show that $U\exists\forall$ -SAT is Π_2^P -hard and contained in $\mathbf{D}_2^P$, a class introduced in [ACHI17] that generalizes $\mathbf{D}^P$ ([PY82]). On the scale between the second and third levels of the polynomial hierarchy, our results seem to place both $U\exists\forall$ -SAT and WDOM-STRATEGY closer to the former.

This characterization is analogous to known results for USAT ("Is there a unique satisfying assignment in a one-quantified Boolean formula?"), which is **coNP**-hard and contained in $\mathbf{D}^P$ ([BG82, PY82]). We now proceed to define a generalization of $\mathbf{D}^P$.

Definition 8.1. ([ACHI17]). For any $k \in \mathbb{N}$, let $\mathbf{D}_k^P = \{\mathbf{L}_1 \cap \mathbf{L}_2 : \mathbf{L}_1 \in \Sigma_k^P \wedge \mathbf{L}_2 \in \Pi_k^P\}$.

We note that $\mathbf{D}^P = \mathbf{D}_1^P$. Furthermore, for all $k \in \mathbb{N}$, we have $\Sigma_k^P \cup \Pi_k^P \subseteq \mathbf{D}_k^P$, since for any language L we have $L = L \cap \Sigma^*$, and $\Sigma^* \in \Sigma_k^P \cap \Pi_k^P$. Additionally, we have that $\mathbf{D}_k^P \subseteq \mathbf{P}^{\Sigma_k^P}$, since two queries to a Σ_k^P oracle can be used to solve any problem in $\mathbf{D}_k^P$.

To see why $\text{WDOM-STRATEGY} \in \mathbf{D}_2^P$, one may observe—though this may not be immediately apparent—that WDOM-STRATEGY can alternatively be stated as follows: “Is there a strategy for Player 1 that weakly dominates any other (without requiring strict domination for some opponent profile), but there do not exist two such strategies?” This reformulation corresponds to the intersection of a Σ_2^P statement and a Π_2^P statement, thus placing the problem within $\mathbf{D}_2^P$.

Problem 8.2:

$U\exists\forall\text{-SAT}$

Input: Two sets $\mathcal{X} = \{x_1, \dots, x_n\}$ and $\mathcal{Y} = \{y_1, \dots, y_n\}$ of Boolean variables and a Boolean formula ψ over $\mathcal{X} \cup \mathcal{Y}$.

Question: $\exists! \mathbf{x}^* \in \{0, 1\}^n \forall \mathbf{y}' \in \{0, 1\}^n \psi(\mathbf{x}^* \parallel \mathbf{y}') = 1$? Where $\exists!$ denotes that there is exactly one such $\mathbf{x}^*$.

Theorem 8.2. $U\exists\forall\text{-SAT} \leq_P \text{WDOM-STRATEGY}$ and $\text{WDOM-STRATEGY} \leq_P U\exists\forall\text{-SAT}$.

Theorem 8.3. $U\exists\forall\text{-SAT} \in \mathbf{D}_2^P$ and $\text{WDOM-STRATEGY} \in \mathbf{D}_2^P$.

Theorem 8.4. $U\exists\forall\text{-SAT}$ and WDOM-STRATEGY are Π_2^P -hard.

9 Discussion

Our work raises many open questions, some of which we highlight here. We have seen that all unambiguous classes discussed above are upper bounded by $\mathbf{S}_2^P$ and lower bounded by either Δ_2^P or $\mathbf{coNP}$. It would be valuable to establish tighter complexity-theoretic bounds for those classes, potentially involving other subclasses of $\mathbf{S}_2^P$, such as $\mathbf{MA}$ and $\mathbf{BPP}$. Notably, $\mathbf{MA}$ and $\mathbf{BPP}$ are unlikely to contain $\mathbf{PTW}$, $\mathbf{PCW}$ or $\mathbf{PMA}$, since such inclusions would imply that $\mathbf{AM}$ (Arthur-Merlin) contains $\mathbf{coNP}$ (given $\mathbf{BPP} \subseteq \mathbf{MA} \subseteq \mathbf{AM}$, see [Bab85, RS98]), which in turn would collapse the polynomial hierarchy to its second level ([BHZ87]).

Another direction for future work is to identify natural problems complete for $\mathbf{PMA}$ and $\mathbf{PTW}$ —that is, problems not defined via unrestricted Boolean circuits.

Finally, it would be intriguing to find unambiguous problems incomparable with $\mathbf{PTW}$ and $\mathbf{PMA}$ due to different combinatorial criteria for uniqueness. Such problems may shed light on further natural subclasses of $\mathbf{US}_2^P$.

Acknowledgements

We would like to thank Martin Bullinger for insightful discussions and comments regarding the proof of Theorem 4.5. We would like to thank Amnon Ta-Shma and Noam Ta-Shma for communicating to us their result of Theorem B.14. Matan Gilboa was supported by the Engineering and Physical Sciences Research Council (EPSRC, grant EP/W524311/1). Paul Goldberg was supported by the EPSRC (grants EP/X040461/1 and EP/X038548/1), and by the ARIA project “*Aggregating Safety Preferences for AI Systems: A Social Choice Approach*.” Noam Nisan was partially supported by a grant from the Israeli Science Foundation (ISF number 505/23).

Appendix

In this appendix, we provide all missing proofs from the paper. Section A contains proofs corresponding to Section 3, Section B to Section 4, Section C to Section 5, Section D to Section 6, Section E to Section 7, and Section F to Section 8. Section G provides the proof of Theorem 4.5.

In case of nested proofs, we use $\triangle$ to mark the end of the inner one and $\square$ for the outer one. We denote by $in_G(x) := |\{v \in V : (v, x) \in E\}|$ the in-degree of vertex $x \in V$ in G ; analogously, $out_G(x) := |\{v \in V : (x, v) \in E\}|$. Given $T \subseteq V$, we denote by $G[T]$ the subgraph induced by T on G . For $n \in \mathbb{N}$, we use the notation $[n] := \{1, \dots, n\}$ and $[n]_0 := \{0, \dots, n-1\}$. We take $\log(n)$ to mean the base 2 logarithm of n .

Recall that given two bit strings $\mathbf{x}$ and $\mathbf{y}$ we write $\mathbf{xy}$ or $\mathbf{x} \parallel \mathbf{y}$ for their concatenation. We may combine both notations in the same expression, e.g., $\mathbf{xy} \parallel \mathbf{x'y'}$, when it is helpful to emphasize that $\mathbf{x}$ and $\mathbf{y}$ (respectively $\mathbf{x'}$ and $\mathbf{y'}$) form natural pairs, even though the entire expression represents a single concatenation. This notation also includes explicit bit strings or single bits, for instance $0 \parallel \mathbf{x}$ or $10^n \mathbf{x}$. We often interpret bit strings as binary numbers, and use operations like $\mathbf{x} < \mathbf{y}$.

A Proofs of Section 3 (PTW)

In this section, we provide all proofs missing from Section 3. We begin with Theorem 3.2.

Proof. (Theorem 3.2). We first argue that $\text{TOURNAMENT-SOURCE} \leq_P \text{WEAK-TOURNAMENT-SOURCE}$. Given an input $\mathcal{C}$ of TOURNAMENT-SOURCE , denote by $\mathcal{C}'$ the circuit that outputs 1 if

$$(\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) = 0) \vee (\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) \wedge \mathbf{x} >_{lex} \mathbf{y})$$

and 0 otherwise. Thus, the question of TOURNAMENT-SOURCE becomes:

$$\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^n \text{ with } \mathbf{y} \neq \mathbf{x} \ \mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = 1$$

A careful review of the definition shows that $\mathcal{C}'$ is anti-symmetric, in that $\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = 1 - \mathcal{C}'(\mathbf{y} \parallel \mathbf{x})$. Thus, this question is equivalent to

$$\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^n \text{ with } \mathbf{y} \neq \mathbf{x} \ \mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = 1 \wedge \mathcal{C}'(\mathbf{y} \parallel \mathbf{x}) = 0$$

which is exactly of the form of $\text{WEAK-TOURNAMENT-SOURCE}$, and therefore, $\mathcal{C}$ is a Yes-instance of TOURNAMENT-SOURCE if and only if $\mathcal{C}'$ is a Yes-instance of $\text{WEAK-TOURNAMENT-SOURCE}$.

We now wish to show $\text{WEAK-TOURNAMENT-SOURCE} \leq_P \text{TOURNAMENT-SOURCE}$. We begin with an intuitive sketch of the reduction. Consider the graph G induced by $\mathcal{C}$, where n -bit strings are vertices, and a directed edge $(\mathbf{x}, \mathbf{y})$ exists if and only if $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 1 \vee \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) = 0$. This is a weak tournament, as some pairs of vertices may have both edge between them (if $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{y} \parallel \mathbf{x})$). We wish to resolve this issue by removing edges where needed, thus forming a tournament G' . We G' to admit a solution if and only if G does. Thus, we cannot simply remove edges arbitrarily, as we might accidentally create a winner. Specifically, we want to make sure that any vertex that tied with another vertex in G remains a loser in G' . To do so, for each pair of vertices $\{\mathbf{x}, \mathbf{y}\}$ in G we add a vertex $\mathbf{v}_{\mathbf{x}, \mathbf{y}}$. If $\mathbf{x}$ and $\mathbf{y}$ tied in G , we will make sure to create a cycle between $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{v}_{\mathbf{x}, \mathbf{y}}$, thus eliminating ties while ensuring none of them is a winner G' . If in G there was only one edge between $\mathbf{x}$ and $\mathbf{y}$, say $(\mathbf{y}, \mathbf{x})$, then we add the edge $(\mathbf{x}, \mathbf{v}_{\mathbf{x}, \mathbf{y}})$ to ensure $\mathbf{v}_{\mathbf{x}, \mathbf{y}}$ is not preventing $\mathbf{x}$ from winning, if $\mathbf{x}$ happens to be a solution in G . Lastly, all added vertices by default lose to any of the original vertices of G that are not associated with them.

We proceed to the formal construction. First, we introduce some definitions. Throughout the construction, for bit strings $\mathbf{x}$ and $\mathbf{y}$, we write $\mathbf{x} < \mathbf{y}$ for $\mathbf{x} \prec_{lex} \mathbf{y}$ (namely, we interpret them as binary numbers and compare them accordingly). Let $\mathbf{x} < \mathbf{y}$. We say the string $\mathbf{x}0^n$ is an *original vertex* which *corresponds* to $\mathbf{x}$, and the string $\mathbf{xy}$ is a *structure vertex* which *corresponds* to the unordered pair $\{\mathbf{x}, \mathbf{y}\}$; we also say $\mathbf{xy}$ *corresponds* to $\mathbf{x}$ and to $\mathbf{y}$ (individually). Original and structure vertices are *valid* strings, while the rest are *invalid*. That is, invalid strings are those of the form $\mathbf{yx}$ (where $\mathbf{x} < \mathbf{y}$), if $\mathbf{x} \neq 0^n$. In particular, we emphasize that if $\mathbf{x} = 0^n$, then the string $\mathbf{y}0^n$ corresponds to $\mathbf{y}$, the string 0^{2n} corresponds to $\mathbf{x}$, and the string $0^n\mathbf{y}$ corresponds to the pair $\{\mathbf{x}, \mathbf{y}\}$.

Now, given a circuit $\mathcal{C}$ with $2n$ inputs for WEAK-TOURNAMENT-SOURCE, we construct a circuit $\mathcal{C}'$ with $4n$ inputs. Given $\mathbf{x}, \mathbf{y}, \mathbf{w}, \mathbf{z} \in \{0, 1\}^n$, we define $\mathcal{C}'(\mathbf{xy} \parallel \mathbf{wz})$ as follows.

First, we describe its behavior on the trivial cases, where non-corresponding or non-valid vertices are given as input.

If $\mathbf{xy} = \mathbf{wz}$, output 0 (recall that we do not care about the behavior of the circuit on identical strings, as the formulation of the question only concerns non-identical strings).

If both of the strings $\mathbf{xy}$ and $\mathbf{wz}$ are invalid, output 1 if and only if $\mathbf{xy} > \mathbf{wz}$.

If $\mathbf{xy}$ is valid and $\mathbf{wz}$ is invalid, output 1, and if the converse holds output 0.

If both of the strings $\mathbf{xy}$ and $\mathbf{wz}$ are structure vertices, output 1 if and only if $\mathbf{xy} > \mathbf{wz}$.

If $\mathbf{xy}$ is an original vertex and $\mathbf{wz}$ is a structure vertex not corresponding to $\{\mathbf{x}, \mathbf{y}\}$, output 1.

If $\mathbf{wz}$ is an original vertex and $\mathbf{xy}$ is a structure vertex not corresponding to $\{\mathbf{w}, \mathbf{z}\}$, output 0.

We now move on to the more interesting cases.

If both $\mathbf{xy}$ and $\mathbf{wz}$ are original vertices (i.e., $\mathbf{y} = \mathbf{z} = 0^n$), calculate $c_1 := \mathcal{C}(\mathbf{x} \parallel \mathbf{w})$ and $c_2 := \mathcal{C}(\mathbf{w} \parallel \mathbf{x})$. If $c_1 \neq c_2$, output c_1 . If $c_1 = c_2$, output 1 if and only if $\mathbf{x} > \mathbf{w}$. Namely, we break the tie somewhat arbitrarily using lexicographic ordering (we avoid unintentionally creating a winner in the following cases).

Suppose $\mathbf{xy}$ is an original vertex and $\mathbf{wz}$ is a structure vertex corresponding to $\mathbf{x}$. Then by definition $\mathbf{z} > \mathbf{w}$, and either $\mathbf{x} = \mathbf{z}$ or $\mathbf{x} = \mathbf{w}$. First, assume $\mathbf{x} = \mathbf{z}$. Calculate $c_1 := \mathcal{C}(\mathbf{x} \parallel \mathbf{w})$ and $c_2 := \mathcal{C}(\mathbf{w} \parallel \mathbf{x})$. If $c_1 > c_2$, output 1. If $c_1 \leq c_2$, output 0 (the case $c_1 = c_2$ is included here because we have $\mathbf{x} > \mathbf{w}$). Thus, we have that $\mathbf{xy}$ (i.e. $\mathbf{x}0^n$) wins against $\mathbf{w}0^n$, so we maintain that neither of the strings is a winner). Now, assume $\mathbf{x} = \mathbf{w}$. Then, calculate $c_1 := \mathcal{C}(\mathbf{x} \parallel \mathbf{z})$ and $c_2 := \mathcal{C}(\mathbf{z} \parallel \mathbf{x})$. Symmetrically to the previous case, if $c_1 \geq c_2$, output 1. If $c_1 < c_2$, output 0 (in this case the tie breaker is in favor of $\mathbf{x}0^n$, since $\mathbf{x}0^n$ already loses to $\mathbf{z}0^n$).

The case where $\mathbf{wz}$ is an original vertex and $\mathbf{xy}$ is a structure vertex corresponding to $\mathbf{w}$ is defined anti-symmetrically (one may define the output by $\mathcal{C}'(\mathbf{xy} \parallel \mathbf{wz}) = 1 - \mathcal{C}'(\mathbf{wz} \parallel \mathbf{xy})$, since $\mathcal{C}'(\mathbf{wz} \parallel \mathbf{xy})$ is well-defined above).

We argue that there exists a WEAK-TOURNAMENT-SOURCE solution in $\mathcal{C}$ if and only if there exists a TOURNAMENT-SOURCE solution in $\mathcal{C}'$. Suppose there exists a WEAK-TOURNAMENT-SOURCE solution $\mathbf{x}^*$ in $\mathcal{C}$, and consider $\mathbf{x}^*0^n$. It is immediate that, in $\mathcal{C}'$, $\mathbf{x}^*0^n$ beats any invalid vertex or non-corresponding structure vertex. Additionally, since $\mathbf{x}^*$ is a winner in $\mathcal{C}$, $\mathbf{x}^*0^n$ must beat any original vertex in $\mathcal{C}'$. It then follows that for any corresponding structure vertex $\mathbf{v}_{\mathbf{x}^*, \mathbf{y}}$ for some $\mathbf{y}$, vertex $\mathbf{x}^*$ beats $\mathbf{y}$, and thus by design $\mathbf{v}_{\mathbf{x}^*, \mathbf{y}}$ loses to $\mathbf{x}^*$. Similarly, for any corresponding structure vertex $\mathbf{v}_{\mathbf{y}, \mathbf{x}^*}$ for some $\mathbf{y}$, vertex $\mathbf{x}^*$ beats $\mathbf{y}$, and thus by design $\mathbf{v}_{\mathbf{y}, \mathbf{x}^*}$ loses to $\mathbf{x}^*$.

In the other direction, if $\mathbf{x}^* = \mathbf{x}_1\mathbf{x}_2$ is a solution to TOURNAMENT-SOURCE in $\mathcal{C}'$, where $\mathbf{x}_1, \mathbf{x}_2 \in \{0, 1\}^n$, then first we must have that $\mathbf{x}_2 = 0^n$, as otherwise $\mathbf{x}^*$ is either invalid or a structure vertex, in which case it loses to any (non-corresponding) original vertex. Secondly, we must have that $\mathbf{x}_1$ is a solution to WEAK-TOURNAMENT-SOURCE in $\mathcal{C}$, since if it loses to some $\mathbf{x}'$ in $\mathcal{C}$ then $\mathbf{x}^*$ loses to $\mathbf{x}'0^n$ in $\mathcal{C}'$, a contradiction; and if $\mathbf{x}_1$ ties with some $\mathbf{x}'$ in $\mathcal{C}$, then by design $\mathbf{x}^*$ must lose either to $\mathbf{x}'0^n$ or to $\mathbf{v}_{\mathbf{x}_1, \mathbf{x}'}$ (depending on whether $\mathbf{x}^* > \mathbf{x}'$ or not). Again, we have a contradiction to $\mathbf{x}^*$ being a solution to $\mathcal{C}'$. $\square$

To prove Theorem 3.3, we begin by establishing the following lemma, which (formulated slightly differently) is attributed by [RMHH04] (Theorem 4) to Erdős. It asserts that if there is no winner in a tournament then we can find a short certificate for it, namely a set S of vertices of logarithmic size such that all vertices lose to some $s \in S$.

Lemma A.1. (*Erdős*) *Let $G = (V, E)$ be a tournament. If there is no source in G , then there exists a set of vertices $S \subseteq V$ with $|S| \leq \log(|V|) + 1$ such that for all $v \in V$ (including $v \in S$) we have $(s, v) \in E$ for some $s \in S$.*

Proof. First, we prove the following proposition:

Proposition A.2. *Let $G = (V, E)$ be a tournament with $|V| \geq 2$. Then there exists a vertex v^* with $\text{out}_G(v^*) \geq \frac{|V|-1}{2}$, namely:*

$$|\{v \in V : (v^*, v) \in E\}| \geq \frac{|V|-1}{2}$$

Since G is a tournament, there are exactly $\binom{|V|}{2} = \frac{|V|(|V|-1)}{2}$ directed edges in the graph. Thus, the average out-degree of the vertices is $\frac{|V|(|V|-1)}{2} / |V| = \frac{|V|-1}{2}$, implying there must exist a vertex with out-degree at least $\frac{|V|-1}{2}$. Therefore, Theorem A.2 holds.

Now, let us consider the following algorithm.

Algorithm 1 Construction of logarithmic winning set

- 1: **Input:** Tournament $G = (V, E)$.
 - 2: **Output:** Set $S \subseteq V$ that beats any vertex.
 - 3: $S \leftarrow \emptyset$
 - 4: $T \leftarrow V$
 - 5: Pick $x \in \text{argmax}\{\text{out}_{G[T]}(x) : x \in T\}$
 - 6: $S \leftarrow S \cup \{x\}$
 - 7: $T \leftarrow \{v \in V : \forall s \in S \text{ with } s \neq v \text{ we have } (v, s) \in E\}$ // vertices who do not lose to anyone in S .
 - 8: If $|T| \geq 2$, go back to Algorithm 1. If $|T| = 1$ with $T = \{t\}$ for some vertex $t \in V$, pick $x \in V \setminus S$ such that $(x, t) \in E$, set $S \leftarrow S \cup \{x\}$, and return S .
-

We wish to show the output set S of the algorithm satisfies the conditions of the lemma. In each iteration T is defined as the set of uncovered vertices in V , namely those that still beat all $s \in S$. Since the algorithm only stops when $T = \emptyset$, we have that upon termination, S indeed covers the entire set V , as all vertices in V lose to some $s \in S$.

It remains to show that the algorithm terminates and that $|S|$ is sufficiently small. Denote by T_i the state of the set T from Algorithm 1 in iteration i . By Theorem A.2 (applied on $G[T]$), we have that $|T_{i+1}| \leq \lceil \frac{|T_i|}{2} \rceil$ for all iterations i but the last iteration (when $|T| = 1$). Furthermore, Algorithm 1 is well defined, since there is no source in G , and therefore there must exist a vertex x that covers the final remaining vertex t . The first observation implies that the number of iterations before we reach $|T| = 1$ is bounded by $\log(|V|)$, while the second observation establishes that the algorithm indeed terminates. Since in each iteration $|S|$ increases by one, we conclude that at the end of the run $|S| \leq \log(|V|) + 1$. $\square$

We can now prove Theorem 3.3.

Proof. (Theorem 3.3). To prove this, we show that TOURNAMENT-SOURCE $\in \mathbf{S}_2^P$. Let $\mathcal{C}$ be an instance of TOURNAMENT-SOURCE. It is useful to rephrase the problem of this instance as follows. For all $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$ define the circuit $\mathcal{C}'$ by:

$$\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{y} \neq \mathbf{x} \wedge ((\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) = 0) \vee (\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{y} \parallel \mathbf{x}) \wedge \mathbf{x} >_{lex} \mathbf{y})) \\ 0 & \text{otherwise} \end{cases}$$

Thus, the question becomes:

$$\exists \mathbf{x} \forall \mathbf{y} \neq \mathbf{x} \mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = 1$$

It follows from the definitions that the answer to this question is Yes if and only if $\mathcal{C}$ is a Yes-instance of TOURNAMENT-SOURCE. This representation is convenient as $\mathcal{C}'$ induces a tournament on $\{0, 1\}^n$, since for all $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$ with $\mathbf{x} \neq \mathbf{y}$ we have $(\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = 1 \wedge \mathcal{C}'(\mathbf{y} \parallel \mathbf{x}) = 0) \vee (\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = 0 \wedge \mathcal{C}'(\mathbf{y} \parallel \mathbf{x}) = 1)$.

Given a set of strings $S \subseteq \{0, 1\}^n$ with $|S| \leq n + 1$, and a string $v \in \{0, 1\}^n$, let $P(\mathcal{C}', S, v)$ be the predicate that outputs 1 if and only if $\mathcal{C}'(s \parallel v) = 0$ for all $s \in S$ (that is, the vertex v loses to no vertex in S). Since $|S| \leq n + 1$, P can be computed in polynomial time.

Now, if there exists a source v^* in the graph induced by $\mathcal{C}'$, then for all $S \subseteq \{0, 1\}^n$ we have $P(\mathcal{C}', S, v^*) = 1$, since v^* loses to no vertex (if v^* itself is in S we still have $P(\mathcal{C}', S, v^*) = 1$, since $\mathcal{C}'(v^* \parallel v^*) = 0$).

On the other hand, if there does not exist such a source, then by Theorem A.1 there exists a set $S \subseteq \{0, 1\}^n$ with $|S| \leq n + 1$ such that for all $v \in \{0, 1\}^n$ we have that v loses to some $s \in S$, namely $P(\mathcal{C}', S, v) = 0$.

Thus, by definition we have that TOURNAMENT-SOURCE $\in \mathbf{S}_2^P$. $\square$

Proof. (Theorem 3.4). It is immediate that WEAK-TOURNAMENT-SOURCE is polynomial-time reducible to MULTI-WEAK-TOURNAMENT-SOURCE, by ignoring the last n input bits.

We now show MULTI-WEAK-TOURNAMENT-SOURCE $\leq_P$ WEAK-TOURNAMENT-SOURCE. Given an instance $\mathcal{C}$ of MULTI-WEAK-TOURNAMENT-SOURCE with $3n$ input bits, we construct a circuit $\mathcal{C}'$ with $4n$ input bits. On inputs $\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, \mathbf{z} \in \{0, 1\}^n$ we define:

$$\mathcal{C}'(\mathbf{x}_1 \parallel \mathbf{x}_2 \parallel \mathbf{y} \parallel \mathbf{z}) = \begin{cases} 1 & \text{if } \mathbf{x}_2 = 0^n \wedge (\mathbf{x}_1 \neq \mathbf{y} \implies (\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y} \parallel \mathbf{z}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}_1 \parallel \mathbf{z}) = 0)) \\ 0 & \text{otherwise} \end{cases}$$

We prove that there exists a solution in the original instance if and only if there exists one in the reduced instance. Assume that there exists $\mathbf{x}^* \in \{0, 1\}^n$ such that $\forall \mathbf{y}, \mathbf{z} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x}^*$ we have that $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y} \parallel \mathbf{z}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}^* \parallel \mathbf{z}) = 0$. Consider the string $\mathbf{x}^*0^n$. Let $\mathbf{y}, \mathbf{z} \in \{0, 1\}^n$ such that $\mathbf{yz} \neq \mathbf{x}^*0^n$. If $\mathbf{y} = \mathbf{x}^*$ then we must have $\mathbf{z} \neq 0^n$. Thus, by definition of $\mathcal{C}'$ we have that $\mathcal{C}'(\mathbf{x}^*0^n \parallel \mathbf{yz}) = 1$ (since $\mathbf{x}^* = \mathbf{y}$), while $\mathcal{C}'(\mathbf{yz} \parallel \mathbf{x}^*0^n) = 0$ (since $\mathbf{z} \neq 0^n$). If $\mathbf{y} \neq \mathbf{x}^*$, then the choice of $\mathbf{x}^*$ ensures that $\mathcal{C}'(\mathbf{x}^*0^n \parallel \mathbf{yz}) = 1 \wedge \mathcal{C}'(\mathbf{yz} \parallel \mathbf{x}^*0^n) = 0$. Hence, $\mathbf{x}^*0^n$ is a solution for WEAK-TOURNAMENT-SOURCE on $\mathcal{C}'$.

In the opposite direction, assume there exist $\mathbf{x}_1, \mathbf{x}_2 \in \{0, 1\}^n$ such that $\forall \mathbf{y}, \mathbf{z} \in \{0, 1\}^n$ with $\mathbf{yz} \neq \mathbf{x}_1\mathbf{x}_2$ we have $\mathcal{C}'(\mathbf{x}_1\mathbf{x}_2 \parallel \mathbf{yz}) = 1 \wedge \mathcal{C}'(\mathbf{yz} \parallel \mathbf{x}_1\mathbf{x}_2) = 0$. Let $\mathbf{y}, \mathbf{z} \in \{0, 1\}^n$ with $\mathbf{y} \neq \mathbf{x}_1$. In particular, this implies $\mathbf{yz} \neq \mathbf{x}_1\mathbf{x}_2$, and therefore $\mathcal{C}'(\mathbf{x}_1\mathbf{x}_2 \parallel \mathbf{yz}) = 1$. Thus, by definition of $\mathcal{C}'$, and since $\mathbf{x}_1 \neq \mathbf{y}$, we have that $\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y} \parallel \mathbf{z}) = 1 \wedge \mathcal{C}(\mathbf{y} \parallel \mathbf{x}_1 \parallel \mathbf{z}) = 0$. Hence, $\mathbf{x}_1$ is a solution for MULTI-WEAK-TOURNAMENT-SOURCE on $\mathcal{C}$. $\square$

B Proofs of Section 4 (PCW)

In this section, we provide all proofs missing from Section 4, except for the proof of Theorem 4.5, which can be found in Section G.

Throughout, given an instance $\langle \mathcal{C}_1, \dots, \mathcal{C}_k \rangle$ of CKT-CONDORCET or CKT-CONDORCET[k], we say a bit string $\mathbf{x}$ is *more popular* than $\mathbf{y}$ if $\mathbf{x}$ beats $\mathbf{y}$ in a majority vote among the circuits, namely if $\sum_{i=1}^k (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_i(\mathbf{y}))) > 0$.

Proof. (Theorem 4.2). CKT-CONDORCET can be reduced to WEAK-TOURNAMENT-SOURCE by constructing a circuit which, given two strings, outputs 1 if the first is more popular than the second in the original instance, and 0 otherwise. A string $x \in \{0, 1\}^n$ is thus a Condorcet winner in the original instance if and only if it is a source in the reduced one. $\square$

Proof. (Theorem 4.4). Item 1 holds since we can pad an instance of CKT-CONDORCET[k] with $k' - k$ degenerate circuits that always output 0, thus not affecting the result of a majority vote of any pair of strings. Item 2 holds trivially since an instance of CKT-CONDORCET[k] is by definition an instance of CKT-CONDORCET. Item 3 holds for $k = 1$ as this yields exactly the problem CKT-UNIQUE-OPT, which by Theorem D.2 is Δ^P -complete; for any $k \geq 2$, the result then follows from Item 1. $\square$

Proof. (Theorem 4.3). The proof follows from Items 2 and 3 of Theorem 4.4. $\square$

We now turn our attention to proving Theorem 4.6. We begin with establishing **PCW**-hardness of the considered problems, followed by containment in **PCW**.

Lemma B.1. $\text{GRAPH-DICE} \leq_P \text{CKT-DICE}$.

Proof. Given an edge-weighted graph we can construct a circuit for each vertex which gets an encoding of a partition as input and outputs the sum of outgoing edge-weights of its vertex in the given partition. The calculations are then identical in both games, and thus there is a winner in the former if and only if there is one in the latter. $\square$

Lemma B.2. GRAPH-DICE is **PCW**-hard.

Proof. We reduce from ASHG-STRONG-POPULARITY (defined in Section G.1) to GRAPH-DICE. The idea of the reduction is to modify the weights such that the ranges of values that different vertices may obtain are pairwise disjoint. Thus, comparisons between different vertices becomes irrelevant since one vertex dominates the other regardless of the input dice, and so a strongly popular partition in the original instance is equivalent to a winning die in the reduced instance. Formally, given an instance $\langle N, v \rangle$ of ASHG-STRONG-POPULARITY, let

$$U = \sum_{\substack{i, j \in [n] \\ v_i(j) > 0}} v_i(j), \text{ and } L = \sum_{\substack{i, j \in [n] \\ v_i(j) < 0}} |v_i(j)| \quad (1)$$

be the sums of all positive and absolute values of all negative values that agents assign to each other in the given hedonic game, respectively. Define the directed graph $G = (V, w)$ with $V = N$ and for all $i, j \in N$, let:

$$w_i(j) = \begin{cases} v_i(j) & \text{if } i \neq j \\ i \cdot (U + L + 1) & \text{if } i = j \end{cases}$$

Notice that weights correspond to valuations of the original instance, except for self-loops which are modified proportionally to the vertex number (whereas in ASHG agents assign value 0 to themselves). Therefore, with s as in the definition of GRAPH-DICE, for any partition π we have that $s_i(\pi) \in [i \cdot (U + L + 1) - L, i \cdot (U + L + 1) + U]$, which can also be written as

$$s_i(\pi) \in [i \cdot (U + L + 1) - L, (i + 1) \cdot (U + L + 1) - L - 1].$$

Thus, given two partitions π_1 and π_2 we have $s_i(\pi_1) \leq (i + 1) \cdot (U + L + 1) - L - 1 < (i + 1) \cdot (U + L + 1) - L \leq s_{i+1}(\pi_2)$. Inductively, for vertices $i < j$ we have that

$$s_i(\pi_1) < s_j(\pi_2). \quad (2)$$

Furthermore, notice that

$$s_i(\pi_1) > s_i(\pi_2) \iff u_i(\pi_1) > u_i(\pi_2) \quad (3)$$

(namely, the preference order of each individual agent over the partitions is preserved), since $s_i(\pi_1) = u_i(\pi_1) + i \cdot (U + L + 1)$ and $s_i(\pi_2) = u_i(\pi_2) + i \cdot (U + L + 1)$. Thus, for any two partitions π_1 and π_2 we have:

$$\begin{aligned} & \sum_{i=1}^n \sum_{j=1}^n (\text{sgn}(s_i(\pi_1) - s_j(\pi_2))) = \\ & \sum_{i=1}^n \left(\sum_{j=1}^{i-1} (\text{sgn}(s_i(\pi_1) - s_j(\pi_2))) + \sum_{j=i+1}^n (\text{sgn}(s_i(\pi_1) - s_j(\pi_2))) + (\text{sgn}(s_i(\pi_1) - s_i(\pi_2))) \right) = \\ & \sum_{i=1}^n \left((i-1) - (n-i) + (\text{sgn}(s_i(\pi_1) - s_i(\pi_2))) \right) = \sum_{i=1}^n (2i - n - 1) + \sum_{i=1}^n (\text{sgn}(u_i(\pi_1) - u_i(\pi_2))) = \\ & \sum_{i=1}^n (\text{sgn}(u_i(\pi_1) - u_i(\pi_2))) \end{aligned}$$

where the second equality stems from Equation (2), the third from Equation (3), and the fourth is due to:

$$\sum_{i=1}^n (2i - n - 1) = 2 \sum_{i=1}^n i - \sum_{i=1}^n n - \sum_{i=1}^n 1 = n(n+1) - n^2 - n = 0.$$

Therefore, we have $\sum_{i=1}^n \sum_{j=1}^n (\text{sgn}(s_i(\pi_1) - s_j(\pi_2))) > 0$ if and only if $\sum_{i=1}^n (\text{sgn}(u_i(\pi_1) - u_i(\pi_2))) > 0$, and so π_1 is a solution for GRAPH-DICE on G if and only if it is a solution for ASHG-STRONG-POPULARITY on $\langle N, v \rangle$. $\square$

Lemma B.3. *CKT-DICE is PCW-hard.*

Proof. This follows immediately from Theorems B.1 and B.2. $\square$

So far, we have seen that both formulations of the dice problem are **PCW**-hard. We now wish to reduce from CKT-DICE to CKT-CONDORCET to show it is in **PCW**. We begin with an intuitive explanation. Consider the following randomized reduction (which we will de-randomize). Given an instance $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$ of CKT-DICE, we create a larger collection of $M > m$ circuits $\mathfrak{C}' = \langle \mathcal{C}'_1, \dots, \mathcal{C}'_M \rangle$, such that for $i \in [M]$ we set $\mathcal{C}'_i(\mathbf{x}) = \mathcal{C}_j(\mathbf{x})$ where j is chosen from $[m]$ uniformly at random. Let $\mathbf{x}$ and $\mathbf{y}$ be two dice, and consider two faces $l, k \in [m]$. For how many values of $i \in [M]$ do we have that $\mathcal{C}'_i(\mathbf{x}) = \mathcal{C}_l(\mathbf{x})$, and $\mathcal{C}'_i(\mathbf{y}) = \mathcal{C}_k(\mathbf{y})$? If M is large enough, with high probability we

have that the number of such values will be almost identical for all (l, k) pairs, and therefore we have that if die $\mathbf{x}$ beats die $\mathbf{y}$ in $\mathfrak{C}$, then with high probability $\mathbf{x}$ is more popular than $\mathbf{y}$ in $\mathfrak{C}'$.

However, we encounter a problem with this idea: If two dice tie with each other, the randomized process will most likely result in one of them being more popular than the other in the reduced Condorcet instance, and thus unintentional Condorcet winners may be created. To cope with this issue, we introduce a *strict* version of CKT-DICE, namely STRICT-CKT-DICE, where we break ties using lexicographic order of the dice labels. We show that the original version can be reduced to the strict one, and then proceed with formalizing a de-randomized version of the construction described above.

The de-randomization process relies on a result from coding theory by Ta-Shma and Ta-Shma (personal communication), who show that there exists a code with a pseudo-randomness property that ensures the desired functionality of the reduction, as described above. That is, instead of setting $\mathcal{C}'_i(\mathbf{x}) = \mathcal{C}_j(\mathbf{x})$ with j chosen randomly, we choose j deterministically according to this code. The notation in Theorem 4.6 differs from the rest of the section in order to correspond with coding theory literature, as appears in Theorems B.12 and B.13 and Theorem B.14.

Problem B.1:

STRICT-CKT-DICE

Input: Sequence of Boolean circuits $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$, with $\mathcal{C}_i: \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$ for all $i \in [m]$.

Question: Does the following hold:

$$\exists \mathbf{x} \in \{0, 1\}^n \ \forall \mathbf{y} \in \{0, 1\}^n \text{ with } \mathbf{y} \neq \mathbf{x} \\ \left(\sum_{i=1}^m \sum_{j=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_j(\mathbf{y}))) > 0 \right) \vee \left(\sum_{i=1}^m \sum_{j=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_j(\mathbf{y}))) = 0 \wedge \mathbf{x} >_{lex} \mathbf{y} \right)$$

Lemma B.4. $CKT-DICE \leq_P STRICT-CKT-DICE$.

Proof. Suppose we are given an instance $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$ of CKT-DICE with $\mathcal{C}_i: \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$ for all $i \in [m]$. We construct an instance $\mathfrak{C}' = \langle \mathcal{C}'_1, \dots, \mathcal{C}'_{4m+3} \rangle$ of STRICT-CKT-DICE, where $\mathcal{C}_i: \{0, 1\}^{2n+2} \rightarrow \{0, 1\}^{n'+1}$ for all $i \in [4m+3]$. To describe the circuits' behavior, we introduce some terminology. We say a vertex (string) $\mathbf{z} \in \{0, 1\}^{2n+2}$ is an *original* vertex that *corresponds* to vertex $\mathbf{x} \in \{0, 1\}^n$ if $\mathbf{z} = 0^{n+2}\mathbf{x}$. We say $\mathbf{z}$ is an *edge-vertex* that *corresponds* to the vertices $\mathbf{x}$ and $\mathbf{y}$, and to the edge $(\mathbf{x}, \mathbf{y})$, if $\mathbf{z} = jk\mathbf{x}\mathbf{y}$, where $j, k \in \{0, 1\}$, and $(j, k) \neq (0, 0)$ (that is, $\mathbf{z}$ starts with 01, 10, or 11). Note that there are three edge-vertices for any edge. We say $\mathbf{z}$ is *valid* if it is an original vertex or an edge-vertex, and *invalid* otherwise.

We now describe the behavior of the circuits in $\mathfrak{C}'$. Let $\mathbf{z} \in \{0, 1\}^{2n+2}$. If $\mathbf{z}$ is invalid, we let $\mathcal{C}'_i(\mathbf{z}) = 0$ for all $i \in [4m+3]$. If $\mathbf{z}$ is valid, we distinguish between the first $4m$ circuits and the last three. The first $4m$ circuits are intuitively meant to multiply the original circuits, to enhance any advantage a vertex had over another vertex in the original game. Thus, let $i \in [m]$. If $\mathbf{z}$ is an original vertex corresponding to $\mathbf{x} \in \{0, 1\}^n$, we set $\mathcal{C}'_{4i-j}(\mathbf{z}) := \mathcal{C}_i(\mathbf{x})$ for all $j \in \{0, 1, 2, 3\}$. If $\mathbf{z}$ is an edge-vertex corresponding to $(\mathbf{x}, \mathbf{y}) \in \{0, 1\}^n \times \{0, 1\}^n$, let $DL_{\mathbf{x}, \mathbf{y}} \in \{\mathbf{x}, \mathbf{y}\}$ be the Dice-loser between $\mathbf{x}$ and $\mathbf{y}$ (if there is a tie, we choose $\mathbf{x}$), that is

$$DL_{\mathbf{x}, \mathbf{y}} = \begin{cases} \mathbf{x} & \text{if } \sum_{i=1}^m \sum_{j=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_j(\mathbf{y}))) \geq 0 \\ \mathbf{y} & \text{otherwise} \end{cases}$$

Then, we set $\mathcal{C}'_{4i-j}(\mathbf{z}) = \mathcal{C}_i(DL_{\mathbf{x}, \mathbf{y}})$ for all $j \in \{0, 1, 2, 3\}$.

We proceed with describing the last three circuits, $\mathcal{C}'_{4m+1}$, $\mathcal{C}'_{4m+2}$, and $\mathcal{C}'_{4m+3}$. If $\mathbf{z}$ is an original

vertex, we set $C'_{4m+i}(\mathbf{z}) = 2^{n'}$ for each $i \in [3]$. If $\mathbf{z}$ is an edge-vertex corresponding to $(\mathbf{x}, \mathbf{y}) \in \{0, 1\}^{n \times \{0, 1\}^n}$, we distinguish further according to the prefix of $\mathbf{z}$. If $\mathbf{z}$ begins with 01, we set $C'_{4m+1}(\mathbf{z}) = 2^{n'} + 2$, $C'_{4m+2}(\mathbf{z}) = 2^{n'} + 4$, and $C'_{4m+3}(\mathbf{z}) = 2^{n'} + 9$. If $\mathbf{z}$ begins with 10, we set $C'_{4m+1}(\mathbf{z}) = 2^{n'} + 1$, $C'_{4m+2}(\mathbf{z}) = 2^{n'} + 6$, and $C'_{4m+3}(\mathbf{z}) = 2^{n'} + 8$. If $\mathbf{z}$ begins with 11, we set $C'_{4m+1}(\mathbf{z}) = 2^{n'} + 3$, $C'_{4m+2}(\mathbf{z}) = 2^{n'} + 5$, and $C'_{4m+3}(\mathbf{z}) = 2^{n'} + 7$.

The intuition is that three edge-vertices v_1 , v_2 , and v_3 corresponding to the same edge form a “winning cycle” in the dice game, since the values they obtain from circuits C'_{4m+1} , C'_{4m+2} , and C'_{4m+3} simulate an intransitive dice instance. This ensures v_1 , v_2 , and v_3 cannot be a solution for STRICT-CKT-DICE on $\mathcal{C}'$; thus, we avoid creating an unintentional winner in the reduction.

Formally, we observe that, on valid vertices, circuits $C'_1, \dots, C'_{4m}$ output values upper bounded by $2^{n'} - 1$, while circuits C'_{4m+1} , C'_{4m+2} , and C'_{4m+3} output values lower bounded by $2^{n'}$. Thus, for all valid vertices $\mathbf{z}, \mathbf{z}' \in \{0, 1\}^{2n+2}$ we have

$$\begin{aligned} & \sum_{i=1}^{4m+3} \sum_{i'=1}^{4m+3} (\text{sgn}(C'_i(\mathbf{z}) - C'_{i'}(\mathbf{z}')) = \\ & \sum_{i=1}^{4m} \sum_{i'=1}^{4m} (\text{sgn}(C'_i(\mathbf{z}) - C'_{i'}(\mathbf{z}')) + \sum_{i=4m+1}^{4m+3} \sum_{i'=4m+1}^{4m+3} (\text{sgn}(C'_i(\mathbf{z}) - C'_{i'}(\mathbf{z}')) \end{aligned} \quad (4)$$

as all other terms are cancel-out (e.g., $\text{sgn}(C'_{4m+1}(\mathbf{z}) - C'_m(\mathbf{z}')) + \text{sgn}(C'_m(\mathbf{z}) - C'_{4m+1}(\mathbf{z}')) = 1 - 1 = 0$). Furthermore, if $\mathbf{z}$ and $\mathbf{z}'$ are original vertices corresponding to $\mathbf{x}$ and $\mathbf{x}'$ respectively, we have:

$$\begin{aligned} & \sum_{i=1}^{4m} \sum_{i'=1}^{4m} (\text{sgn}(C'_i(\mathbf{z}) - C'_{i'}(\mathbf{z}')) = \sum_{i=1}^m \sum_{j=0}^3 \sum_{i'=1}^m \sum_{j'=0}^3 (\text{sgn}(C'_{4i-j}(\mathbf{z}) - C'_{4i'-j'}(\mathbf{z}')) = \\ & \sum_{i=1}^m \sum_{j=0}^3 \sum_{i'=1}^m \sum_{j'=0}^3 (\text{sgn}(C_i(\mathbf{x}) - C_{i'}(\mathbf{x}')) = 16 \cdot \sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(C_i(\mathbf{x}) - C_{i'}(\mathbf{x}'))). \end{aligned} \quad (5)$$

If $\mathbf{z}$ is an original vertex corresponding to $\mathbf{x}$ and $\mathbf{z}_{\mathbf{x}', \mathbf{x}''}$ is an edge-vertex, we have:

$$\begin{aligned} & \sum_{i=1}^{4m} \sum_{i'=1}^{4m} (\text{sgn}(C'_i(\mathbf{z}) - C'_{i'}(\mathbf{z}_{\mathbf{x}', \mathbf{x}''}))) = \sum_{i=1}^m \sum_{j=0}^3 \sum_{i'=1}^m \sum_{j'=0}^3 (\text{sgn}(C'_{4i-j}(\mathbf{z}) - C'_{4i'-j'}(\mathbf{z}_{\mathbf{x}', \mathbf{x}''}))) = \\ & 16 \cdot \sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(C_i(\mathbf{x}) - C_{i'}(DL_{\mathbf{x}', \mathbf{x}''}))) \end{aligned} \quad (6)$$

We now establish several claims.

Proposition B.5. *Let $\mathbf{z}, \mathbf{z}' \in \{0, 1\}^{2n+2}$, where $\mathbf{z}$ is a valid vertex and $\mathbf{z}'$ is invalid. Then $\mathbf{z}$ beats $\mathbf{z}'$ on $\mathcal{C}'$.*

Proof. We have that $C'_i(\mathbf{z}') = 0$ for all $i \in [4m+3]$, while $C'_i(\mathbf{z}) \geq 0$ for all $i \in [4m+3]$ and $C'_{4m+1}(\mathbf{z}) \geq 2^{n'} > 0$. Hence, the number of pairs $(i, i') \in [4m+3]$ such that $C'_i(\mathbf{z}) > C'_{i'}(\mathbf{z}')$ is at least $4m+3$, while the number of pairs $(i, i') \in [4m+3]$ such that $C'_i(\mathbf{z}) < C'_{i'}(\mathbf{z}')$ is 0. Therefore $\mathbf{z}$ beats $\mathbf{z}'$. $\triangle$

Proposition B.6. *Let $\mathbf{x}, \mathbf{x}' \in \{0, 1\}^n$, and denote by $\mathbf{z} = 0^{n+2}\mathbf{x}$ and $\mathbf{z}' = 0^{n+2}\mathbf{x}'$ the corresponding original vertices. If $\mathbf{x}$ beats $\mathbf{x}'$ on $\mathcal{C}$, then $\mathbf{z}$ beats $\mathbf{z}'$ on $\mathcal{C}'$.*

Proof. Since $\mathbf{x}$ beats $\mathbf{x}'$ on $\mathfrak{C}$, we have $\sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_{i'}(\mathbf{x}')) \geq 1$. Thus, by Equations (4) and (5) we have that $\sum_{i=1}^{4m+3} \sum_{i'=1}^{4m+3} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_{i'}(\mathbf{z}')) \geq 16 - 9 = 7 > 0$, and therefore $\mathbf{z}$ beats $\mathbf{z}'$ on $\mathfrak{C}'$. $\triangle$

Proposition B.7. *Let $\mathbf{x}, \mathbf{x}' \in \{0, 1\}^n$, and denote their corresponding original vertices $\mathbf{z} = 0^{n+2}\mathbf{x}$ and $\mathbf{z}' = 0^{n+2}\mathbf{x}'$. Let $\mathbf{z}_{\mathbf{x}, \mathbf{x}'} \in \{0, 1\}^{2n+2}$ be an edge-vertex corresponding to the edge $(\mathbf{x}, \mathbf{x}')$. Then:*

1. *If $\mathbf{x}$ ties with $\mathbf{x}'$ on $\mathfrak{C}$, then $\mathbf{z}_{\mathbf{x}, \mathbf{x}'}$ beats $\mathbf{z}$ and $\mathbf{z}'$ on $\mathfrak{C}'$.*
2. *If $\mathbf{x}$ beats $\mathbf{x}'$ on $\mathfrak{C}$, then $\mathbf{z}$ beats $\mathbf{z}_{\mathbf{x}, \mathbf{x}'}$ on $\mathfrak{C}'$.*

Proof. We have:

$$\forall i, j \in [3] \quad \mathcal{C}'_{4m+i}(\mathbf{z}_{\mathbf{x}, \mathbf{x}'}) > \mathcal{C}'_{4m+j}(\mathbf{z}) \quad (7)$$

For the first part, assume $\mathbf{x}$ ties with $\mathbf{x}'$ on $\mathfrak{C}$. Then $\sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_{i'}(\mathbf{x}')) = 0$. Hence, by Equation (6) we have $\sum_{i=1}^{4m} \sum_{i'=1}^{4m} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_{i'}(\mathbf{z}_{\mathbf{x}, \mathbf{x}'})) = 0$. Hence, by Equations (4) and (7), we have that $\sum_{i=1}^{4m+3} \sum_{i'=1}^{4m+3} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_{i'}(\mathbf{z}')) = 0 - 9 < 0$ and therefore $\mathbf{z}_{\mathbf{x}, \mathbf{x}'}$ beats $\mathbf{z}$ (and $\mathbf{z}'$, similarly) on $\mathfrak{C}'$.

For the second part, assume $\mathbf{x}$ beats $\mathbf{x}'$ on $\mathfrak{C}$. Then $\sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_{i'}(\mathbf{x}')) \geq 1$ and $DL_{\mathbf{x}, \mathbf{x}'} = \mathbf{x}'$. Thus, by Equation (6) we have that $\sum_{i=1}^{4m} \sum_{j=1}^{4m} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_j(\mathbf{z}_{\mathbf{x}, \mathbf{x}'})) \geq 16$, and by Equation (4) we have:

$$\sum_{i=1}^{4m+3} \sum_{j=1}^{4m+3} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_j(\mathbf{z}_{\mathbf{x}, \mathbf{x}'})) \geq 16 - 9 = 7 > 0$$

and therefore $\mathbf{z}$ beats $\mathbf{z}_{\mathbf{x}, \mathbf{x}'}$. $\triangle$

Proposition B.8. *Let $\mathbf{x} \in \{0, 1\}^n$, and denote the corresponding original vertex $\mathbf{z} = 0^{n+2}\mathbf{x}$. Let $\mathbf{z}_{\mathbf{x}', \mathbf{x}''} \in \{0, 1\}^{2n+2}$ be an edge-vertex corresponding to the edge $(\mathbf{x}', \mathbf{x}'')$, where $\mathbf{x} \notin \{\mathbf{x}', \mathbf{x}''\}$. If $\mathbf{x}$ beats $\mathbf{x}'$ and $\mathbf{x}''$ on $\mathfrak{C}$, then $\mathbf{z}$ beats $\mathbf{z}_{\mathbf{x}', \mathbf{x}''}$ on $\mathfrak{C}'$.*

Proof. We have that $\mathbf{x}$ beats $DL_{\mathbf{x}', \mathbf{x}''}$ and thus by Equation (6) we have that $\sum_{i=1}^{4m} \sum_{j=1}^{4m} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_j(\mathbf{z}_{\mathbf{x}', \mathbf{x}''})) \geq 16$. Hence, by Equation (4) we have $\sum_{i=1}^{4m+3} \sum_{j=1}^{4m+3} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_j(\mathbf{z}_{\mathbf{x}', \mathbf{x}''})) \geq 16 - 9 = 7 > 0$, and therefore $\mathbf{z}$ beats $\mathbf{z}_{\mathbf{x}', \mathbf{x}''}$. $\triangle$

Proposition B.9. *Let $\mathbf{z}, \mathbf{z}', \mathbf{z}'' \in \{0, 1\}^{2n+2}$ be three edge-vertices corresponding to the same edge, where $\mathbf{z}$ begins with 01, $\mathbf{z}'$ with 10, and $\mathbf{z}''$ with 11. Then, on $\mathfrak{C}'$, $\mathbf{z}$ beats $\mathbf{z}'$, $\mathbf{z}'$ beats $\mathbf{z}''$, and $\mathbf{z}''$ beats $\mathbf{z}$.*

Proof. $\mathbf{z}, \mathbf{z}'$ and $\mathbf{z}''$ all obtain identical values in all circuits $\mathcal{C}'_i$ for $i \in [4m]$. Hence, by Equation (4) we have that

$$\sum_{i=1}^{4m+3} \sum_{j=1}^{4m+3} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_j(\mathbf{z}')) = \sum_{i=4m+1}^{4m+3} \sum_{j=4m+1}^{4m+3} (\text{sgn}(\mathcal{C}'_i(\mathbf{z}) - \mathcal{C}'_j(\mathbf{z}')) \quad (8)$$

By definition we have:

- $\mathcal{C}'_{4m+1}(\mathbf{z}) = 2$, $\mathcal{C}'_{4m+2}(\mathbf{z}) = 4$, and $\mathcal{C}'_{4m+3}(\mathbf{z}) = 9$.
- $\mathcal{C}'_{4m+1}(\mathbf{z}) = 1$, $\mathcal{C}'_{4m+2}(\mathbf{z}) = 6$, and $\mathcal{C}'_{4m+3}(\mathbf{z}) = 8$.
- $\mathcal{C}'_{4m+1}(\mathbf{z}) = 3$, $\mathcal{C}'_{4m+2}(\mathbf{z}) = 5$, and $\mathcal{C}'_{4m+3}(\mathbf{z}) = 7$.

Hence, by Equation (8), it may be verified that on $\mathfrak{C}'$, $\mathbf{z}$ beats $\mathbf{z}'$, $\mathbf{z}'$ beats $\mathbf{z}''$, and $\mathbf{z}''$ beats $\mathbf{z}$. $\triangle$

Now, suppose $\mathbf{x}^* \in \{0, 1\}^n$ is a solution for CKT-DICE on $\mathfrak{C}$, and let $\mathbf{z}^* = 0^{n+2}\mathbf{x}$ ($\mathbf{z}^*$ corresponds to $\mathbf{x}^*$). By Theorem B.5 $\mathbf{z}^*$ beats all invalid vertices, by Theorem B.6 it beats all other original vertices, by Theorem B.7 it beats all edge-vertices corresponding to it, and by Theorem B.8 it beats all edge-vertices not corresponding to it. Hence, $\mathbf{z}^*$ is a solution for STRICT-CKT-DICE on $\mathfrak{C}'$.

Conversely, assume there exists $\mathbf{z}^* \in \{0, 1\}^{2n+2}$ that beats all other $\mathbf{z} \in \{0, 1\}^{2n+2}$ in $\mathfrak{C}'$. By Theorems B.5 and B.9 we have that $\mathbf{z}^*$ must be an original vertex. Thus, let $\mathbf{x}^* \in \{0, 1\}^n$ such that $\mathbf{z}^* = 0^{n+2}\mathbf{x}^*$. Let $\mathbf{x}' \in \{0, 1\}^n$, and denote by $\mathbf{z}' = 0^{n+2}\mathbf{x}'$ its corresponding original vertex. If $\mathbf{x}'$ beats $\mathbf{x}^*$ on $\mathfrak{C}$, then by Theorem B.6 we have that $\mathbf{z}'$ beats $\mathbf{z}^*$ on $\mathfrak{C}'$, a contradiction. If $\mathbf{x}'$ ties with $\mathbf{x}^*$, then by Theorem B.7 the edge-vertices corresponding to the edge $(\mathbf{x}', \mathbf{x}^*)$ beat $\mathbf{z}^*$, a contradiction. Hence, $\mathbf{x}^*$ beats $\mathbf{x}'$, and therefore $\mathbf{x}^*$ is a solution for STRICT-CKT-DICE on $\mathfrak{C}$. $\square$

Lemma B.10. *Given an instance $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$ of STRICT-CKT-DICE, with $\mathcal{C}_i: \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$ for all $i \in [m]$, one can construct in polynomial time an instance $\mathfrak{C}' = \langle \mathcal{C}'_1, \dots, \mathcal{C}'_{2m+1} \rangle$ of STRICT-CKT-DICE, with $\mathcal{C}'_i: \{0, 1\}^n \rightarrow \{0, 1\}^{n'+1}$ for all $i \in [2m+1]$, such that:*

1. $\mathfrak{C}'$ is a Yes-instance if and only if $\mathfrak{C}$ is a Yes-instance, and
2. for each $\mathbf{x}', \mathbf{y}' \in \{0, 1\}^n$, $\mathbf{x}'$ and $\mathbf{y}'$ do not tie on $\mathfrak{C}'$.

Proof. As in the proof of Theorem B.4, circuits $\mathcal{C}'_1, \dots, \mathcal{C}'_{2m}$ are intuitively meant to multiply the original circuits, to enhance any advantage a vertex had over another vertex in the original game. The last circuit is meant to break ties lexicographically. Thus, let $i \in [m]$ and $\mathbf{x} \in \{0, 1\}^n$. We set $\mathcal{C}'_{2i}(\mathbf{x}) = \mathcal{C}'_{2i-1}(\mathbf{x}) = \mathcal{C}_i(\mathbf{x})$, and $\mathcal{C}'_{2m+1}(\mathbf{x}) = 2^{n'} + \mathbf{x}$.

We observe that circuits $\mathcal{C}'_1, \dots, \mathcal{C}'_{2m}$ output values upper bounded by $2^{n'} - 1$, while circuit $\mathcal{C}'_{2m+1}$ outputs values lower bounded by $2^{n'}$. Thus, for all vertices $\mathbf{x}, \mathbf{x}' \in \{0, 1\}^n$ we have:

$$\sum_{i=1}^{2m+1} \sum_{i'=1}^{2m+1} (\text{sgn}(\mathcal{C}'_i(\mathbf{x}) - \mathcal{C}'_{i'}(\mathbf{x}')))) = \sum_{i=1}^{2m} \sum_{i'=1}^{2m} (\text{sgn}(\mathcal{C}'_i(\mathbf{x}) - \mathcal{C}'_{i'}(\mathbf{x}')))) + \text{sgn}(\mathcal{C}'_{2m+1}(\mathbf{x}) - \mathcal{C}'_{2m+1}(\mathbf{x}')) \quad (9)$$

Furthermore, we have

$$\begin{aligned} \sum_{i=1}^{2m} \sum_{i'=1}^{2m} (\text{sgn}(\mathcal{C}'_i(\mathbf{x}) - \mathcal{C}'_{i'}(\mathbf{x}')))) &= \sum_{i=1}^m \sum_{j=0}^1 \sum_{i'=1}^m \sum_{j'=0}^1 (\text{sgn}(\mathcal{C}'_{2i-j}(\mathbf{x}) - \mathcal{C}'_{2i'-j'}(\mathbf{x}')))) = \\ &= \sum_{i=1}^m \sum_{j=0}^1 \sum_{i'=1}^m \sum_{j'=0}^1 (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_{i'}(\mathbf{x}')))) = 4 \sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}) - \mathcal{C}_{i'}(\mathbf{x}')))) \end{aligned} \quad (10)$$

Additionally, by definition we have

$$\text{sgn}(\mathcal{C}'_{2m+1}(\mathbf{x}) - \mathcal{C}'_{2m+1}(\mathbf{x}')) = \text{sgn}(\mathbf{x} - \mathbf{x}') \quad (11)$$

We require the following proposition to complete the proof of the lemma.

Proposition B.11. *Let $\mathbf{x}, \mathbf{x}' \in \{0, 1\}^n$ with $\mathbf{x} > \mathbf{x}'$. If one of them beats the other in $\mathfrak{C}$, then it beats it in $\mathfrak{C}'$ as well. If they tie, then $\mathbf{x}$ beats $\mathbf{x}'$ on $\mathfrak{C}$.*

Proof. For the first part, assume without loss of generality that $\mathbf{x}'$ beats $\mathbf{x}$ on $\mathfrak{C}$. Then $\sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}') - \mathcal{C}_{i'}(\mathbf{x}))) \geq 1$, and therefore by Equations (9) and (10) we have that $\sum_{i=1}^{4m+3} \sum_{i'=1}^{4m+3} (\text{sgn}(\mathcal{C}'_i(\mathbf{x}') - \mathcal{C}'_{i'}(\mathbf{x}))) \geq 4 - 1 = 3 > 0$. Hence, we have that $\mathbf{x}'$ beats $\mathbf{x}$ on $\mathfrak{C}'$.

If $\mathbf{x}$ and $\mathbf{x}'$ tie on $\mathfrak{C}$, then $\sum_{i=1}^m \sum_{i'=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}') - \mathcal{C}_{i'}(\mathbf{x}))) = 0$. Hence, by Equations (9) to (11) we have that $\mathbf{x}^*$ strictly beats $\mathbf{x}$ on $\mathfrak{C}'$. $\triangle$

From Theorem B.11 and by definition of STRICT-CKT-DICE, it is immediate that $\mathbf{x}^*$ is a solution of STRICT-CKT-DICE on $\mathfrak{C}$ if and only if it is a solution for STRICT-CKT-DICE on $\mathfrak{C}'$. Additionally, by Theorem B.11, no two string can tie on $\mathfrak{C}'$, since if one of them beats the other on $\mathfrak{C}$ then it does so in $\mathfrak{C}'$ as well, and if they tie on $\mathfrak{C}$ then the numeric winner beats the other on $\mathfrak{C}'$. $\square$

Definition B.12. Let $q, n \in \mathbb{N}$. A set S of words, each in $[q]^n$, is called an ϵ -pairwise code if for every $u, w \in S$ and every $a, b \in [q]$ we have that $|\mathbb{P}_{i \sim [n]}(u_i = a \wedge w_i = b) - \frac{1}{q^2}| \leq \epsilon$.

Definition B.13. A set S of words, each in $[q]^n$ is called *poly-constructible* if there is an algorithm that, when given an index $t \in [|S|]$ outputs the t 'th word in S in time that is polynomial in the length of the input ($\log(t)$) and the length of the output ($n \log(q)$).

Theorem B.14. (*Amnon Ta-Shma and Noam Ta-Shma, personal communication*). For every $q, T \in \mathbb{N}$ and $\epsilon > 0$, there exists $n = \text{poly}(q, \epsilon^{-1}, \log(T))$ and a poly-constructible ϵ -pairwise code S over $[q]^n$ with $|S| = T$.

Proof. (**Theorem 4.6**). By Theorems B.2 and B.3 we have that CKT-DICE and GRAPH-DICE are **PCW**-hard, and by Theorems B.1 and B.4 we have that they both reduce to STRICT-CKT-DICE. Hence, it suffices to prove that STRICT-CKT-DICE is in **PCW**. We construct a polynomial time reduction from STRICT-CKT-DICE to CKT-CONDORCET.

Let $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$ be an instance of STRICT-CKT-DICE, with $\mathcal{C}_i: \{0, 1\}^n \rightarrow \{0, 1\}^{n'}$ for all $i \in [m]$. By Theorem B.10, we may assume without loss of generality that no two dice tie on $\mathfrak{C}$. Let $\epsilon < \frac{1}{2m^4}$. Then, by Theorem B.14, there exists $m' = \text{poly}(m, \epsilon^{-1}, \log(2^n)) = \text{poly}(m, n)$ and a poly-constructible ϵ -pairwise code S over $[m]^{m'}$ with $|S| = 2^n$. For $\mathbf{x} \in \{0, 1\}^n$, let $S(\mathbf{x})$ denote the $\mathbf{x}$ 'th word in S (where $\mathbf{x}$ is interpreted as a binary number), and for each $i \in [m']$ let $S(\mathbf{x})_i$ denote the i 'th element of $S(\mathbf{x})$ (recall that $S(\mathbf{x}) \in [m]^{m'}$, and thus $S(\mathbf{x})_i \in [m]$). We define the instance $\mathfrak{C}' = \langle \mathcal{C}'_1, \dots, \mathcal{C}'_{m'} \rangle$ by $\mathcal{C}'_i(\mathbf{x}) = \mathcal{C}_{S(\mathbf{x})_i}(\mathbf{x})$ for any $i \in [m']$ and $\mathbf{x} \in \{0, 1\}^n$. We can efficiently construct $\mathfrak{C}'$ as S is poly-constructible.

Let $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$. For any $i \in [m']$, let

$$D_{\mathbf{x}, \mathbf{y}, i} = \text{sgn}(\mathcal{C}'_i(\mathbf{x}) - \mathcal{C}'_i(\mathbf{y})) = \text{sgn}(\mathcal{C}_{S(\mathbf{x})_i}(\mathbf{x}) - \mathcal{C}_{S(\mathbf{y})_i}(\mathbf{y}))$$

For $a, b \in [m]$, define the event

$$W_{\mathbf{x}, \mathbf{y}}^{a, b} := \mathcal{C}_a(\mathbf{x}) > \mathcal{C}_b(\mathbf{y})$$

and let $A = |\{(a, b) \in [m] \times [m]: W_{\mathbf{x}, \mathbf{y}}^{a, b} \text{ holds}\}|$ and $B = |\{(a, b) \in [m] \times [m]: W_{\mathbf{y}, \mathbf{x}}^{a, b} \text{ holds}\}|$. Suppose $\mathbf{x}$ beats $\mathbf{y}$ in the Dice game on $\mathfrak{C}$. Then we have that

$$A - B \geq 1. \tag{12}$$

Furthermore, since $A, B \subseteq [m] \times [m]$ we have

$$A + B \leq 2m^2. \tag{13}$$

Let

$$F_{\mathbf{x}, \mathbf{y}}^S = \sum_{i=1}^{m'} \text{sgn}(\mathcal{C}'_i(\mathbf{x}) - \mathcal{C}'_i(\mathbf{y})) = \sum_{i=1}^{m'} D_{\mathbf{x}, \mathbf{y}, i}$$

Notice that $\mathbf{x}$ is more popular than $\mathbf{y}$ in $\mathfrak{C}'$ if and only if $F_{\mathbf{x},\mathbf{y}}^S > 0$. Thus, we want to show that $F_{\mathbf{x},\mathbf{y}}^S > 0$. First, we calculate the expectation of $D_{\mathbf{x},\mathbf{y},i}$, where i is drawn uniformly at random from $[m']$. We have:

$$\mathbb{E}_{i \sim [m']} [D_{\mathbf{x},\mathbf{y},i}] = 1 \cdot \mathbb{P}_{i \sim [m']}(\mathcal{C}'_i(\mathbf{x}) > \mathcal{C}'_i(\mathbf{y})) - 1 \cdot \mathbb{P}_{i \sim [m']}(\mathcal{C}'_i(\mathbf{y}) > \mathcal{C}'_i(\mathbf{x})) + 0 \cdot \mathbb{P}_{i \sim [m']}(\mathcal{C}'_i(\mathbf{x}) = \mathcal{C}'_i(\mathbf{y})) \quad (14)$$

We therefore calculate the above probabilities:

$$\begin{aligned} \mathbb{P}_{i \sim [m']}(\mathcal{C}'_i(\mathbf{x}) > \mathcal{C}'_i(\mathbf{y})) &= \mathbb{P}_{i \sim [m']}(\mathcal{C}_{S(\mathbf{x})_i}(\mathbf{x}) > \mathcal{C}_{S(\mathbf{y})_i}(\mathbf{y})) = \sum_{\substack{(a,b) \in [m] \times [m] \\ \text{s.t. } W_{\mathbf{x},\mathbf{y}}^{a,b}}} \mathbb{P}_{i \sim [m]}(S(\mathbf{x})_i = a \wedge S(\mathbf{y})_i = b) \geq \\ &\sum_{\substack{(a,b) \in [m] \times [m] \\ \text{s.t. } W_{\mathbf{x},\mathbf{y}}^{a,b}}} \left(\frac{1}{m^2} - \epsilon\right) = A \cdot \left(\frac{1}{m^2} - \epsilon\right) \end{aligned}$$

where the inequality follows from the definition of an ϵ -pairwise code. Similarly, we have:

$$\mathbb{P}_{i \sim [m']}(\mathcal{C}'_i(\mathbf{y}) > \mathcal{C}'_i(\mathbf{x})) \leq B \cdot \left(\frac{1}{m^2} + \epsilon\right)$$

Thus, by Equation (14) we have:

$$\mathbb{E}_{i \sim [m']} [D_{\mathbf{x},\mathbf{y},i}] \geq A \cdot \left(\frac{1}{m^2} - \epsilon\right) - B \cdot \left(\frac{1}{m^2} + \epsilon\right) = \frac{A-B}{m^2} - (A+B)\epsilon \geq_{(*)} \frac{1}{m^2} - 2m^2\epsilon >_{(**)} \frac{1}{m^2} - \frac{2m^2}{2m^4} = 0$$

where inequality $(*)$ follows from Equations (12) and (13), and inequality $(**)$ follows from the definition of ϵ . Now, if $\mathfrak{C}$ is a Yes-instance of STRICT-CKT-DICE then there exists a die $\mathbf{x}^* \in \{0,1\}^n$ that beats any other die $\mathbf{y} \in \{0,1\}^n$. Hence, we have $F_{\mathbf{x}^*,\mathbf{y}}^S > 0$ for all $\mathbf{y} \in \{0,1\}^n$, and thus $\mathbf{x}^*$ is more popular than any $\mathbf{y}$ in STRICT-CKT-DICE on $\mathfrak{C}'$, namely $\mathfrak{C}'$ is a Yes-instance of STRICT-CKT-DICE. Conversely, assume $\mathfrak{C}$ is a No-instance of STRICT-CKT-DICE, and let $\mathbf{x}' \in \{0,1\}^n$ be a die. Then there must exist some $\mathbf{y}' \in \{0,1\}^n$ that beats $\mathbf{x}'$ in STRICT-CKT-DICE defined on $\mathfrak{C}$. Thus, we have $F_{\mathbf{x}',\mathbf{y}'}^S < 0$, and therefore $\mathbf{x}'$ is not a strongly popular string in the instance $\mathfrak{C}'$ of STRICT-CKT-DICE. Hence, $\mathfrak{C}'$ is a No-instance of STRICT-CKT-DICE. $\square$

C Proofs of Section 5 (PMA)

In this section, we provide all proofs missing from Section 5, beginning with Theorem 5.5. We first show **PMA**-completeness of an intermediate problem EDGE-MAJORITY-BALANCED, where the induced bipartite graph is balanced but the threshold k remains part of the input. We then show that parameterizing the number of disqualified edges k does not affect the complexity, beginning with the case $k = 1$. We use these simpler variants to show the remaining results of the section regarding computational bounds on **PMA**.

Problem C.1:

EDGE-MAJORITY-BALANCED

Input: Two Boolean circuits $\mathcal{C}: \{0,1\}^{2n} \rightarrow \{0,1\}^{n+1}$ and $\mathcal{V}: \{0,1\}^{n+1} \rightarrow \{0,1\}^{2n}$, and integer $k \geq 1$.

Question: $\exists \mathbf{x} \in \{0,1\}^n \forall \mathbf{y} \in \{0,1\}^n \mathcal{V}(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) = \mathbf{x}\mathbf{y} \wedge \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \geq k$?

Lemma C.1. *EDGE-MAJORITY-BALANCED is **PMA**-complete.*

Proof. It is clear that EDGE-MAJORITY-BALANCED $\in$ **PMA** as EDGE-MAJORITY-BALANCED is a special case of EDGE-MAJORITY. To show hardness, we reduce from EDGE-MAJORITY. We begin with a proof sketch. Consider the bipartite graph $G = (X, Y, E)$ induced by a given EDGE-MAJORITY instance, with edges only from X to Y . If $|X| = |Y|$ then this is an EDGE-MAJORITY-BALANCED instance and we are done. If $|Y| > |X|$ then we can add “dummy” vertices to X with out-degree 0, to balance the sizes of both parts. Thus, the number of edges in the graph does not increase and therefore the bound on the number of edges is still satisfied, so we obtain a valid EDGE-MAJORITY-BALANCED instance, and winners are preserved. If $|Y| < |X|$, we add vertices to Y , but this time we must make sure that if a winner $\mathbf{x}^*$ exists in G then the edges from $\mathbf{x}^*$ to the new Y -vertices are present. To do so, we proceed inductively, adding 1 bit to the representation of the vertices of Y in each step (and in total $n - m$ steps). We think of each vertex $\mathbf{y} \in Y$ as “splitting” into two vertices $\mathbf{y}_1 = 0\mathbf{y}$ and $\mathbf{y}_2 = 1\mathbf{y}$, and similarly any edge label l is split into $0l$ and $1l$. We then make sure that if $\mathbf{xy}$ was an edge with label l , then $\mathbf{xy}_1$ and $\mathbf{xy}_2$ are edges with labels $0l$ and $1l$ respectively. It is not hard to show that solutions are preserved by this reduction.

Formally, let $n \neq m$, and suppose we are given an instance $\langle \mathcal{C}, \mathcal{V}, k \rangle$ of EDGE-MAJORITY, where $\mathcal{C}$ has $n + m$ inputs and $m + 1$ outputs, $\mathcal{V}$ has $m + 1$ inputs and $n + m$ outputs, and $k \geq 1$. If $n = m$ then this is also a EDGE-MAJORITY-BALANCED instance and we are done. We consider the cases $n < m$ and $m < n$.

First, assume $n < m$. Define circuits $\mathcal{C}': \{0, 1\}^{2m} \rightarrow \{0, 1\}^{m+1}$ and $\mathcal{V}': \{0, 1\}^{m+1} \rightarrow \{0, 1\}^{2m}$ as follows. For $\mathbf{x}, \mathbf{y} \in \{0, 1\}^m$, where $\mathbf{x} = \mathbf{x}_1\mathbf{x}_2$ with $\mathbf{x}_1 \in \{0, 1\}^{m-n}$ and $\mathbf{x}_2 \in \{0, 1\}^n$, we set:

$$\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = \begin{cases} \mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y}) & \text{if } \mathbf{x}_1 = 0^{m-n} \\ 0^{m+1} & \text{otherwise} \end{cases} \quad (15)$$

For $\mathbf{z} \in \{0, 1\}^{m+1}$, we set

$$\mathcal{V}'(\mathbf{z}) = 0^{m-n}\mathcal{V}(\mathbf{z}). \quad (16)$$

Consider the instance $\langle \mathcal{C}', \mathcal{V}', k \rangle$ of EDGE-MAJORITY-BALANCED. Let $\mathbf{x}^* \in \{0, 1\}^n$ be a solution for EDGE-MAJORITY on $\langle \mathcal{C}, \mathcal{V}, k \rangle$, and let $\mathbf{y} \in \{0, 1\}^m$. We have $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$ and $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k$. Consider $0^{m-n}\mathbf{x}^*$. We have $\mathcal{C}'(0^{m-n}\mathbf{x}^* \parallel \mathbf{y}) = \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k$. Furthermore, we have $\mathcal{V}'(\mathcal{C}'(0^{m-n}\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}'(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = 0^{m-n}\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = 0^{m-n}\mathbf{x}^*\mathbf{y}$. Hence, $0^{m-n}\mathbf{x}^*$ is a solution for EDGE-MAJORITY-BALANCED on $\langle \mathcal{C}', \mathcal{V}', k \rangle$.

Conversely, assume $0^{n-m}\mathbf{x}^*$ is a solution for EDGE-MAJORITY-BALANCED on $\langle \mathcal{C}', \mathcal{V}', k \rangle$ (note that any solution must have a prefix of 0^{n-m} , as otherwise $\mathcal{C}'$ maps it to 0^{m+1} regardless of $\mathbf{y}$, and $0^{m+1} < k$). Let $\mathbf{y} \in \{0, 1\}^m$. We have $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) = \mathcal{C}'(0^{n-m}\mathbf{x}^* \parallel \mathbf{y}) \geq k$. Furthermore, we have $0^{n-m}\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}'(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}'(\mathcal{C}'(0^{n-m}\mathbf{x}^* \parallel \mathbf{y})) = 0^{n-m}\mathbf{x}^*\mathbf{y}$, implying that $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$. Therefore, $\mathbf{x}^*$ is a solution for EDGE-MAJORITY on $\langle \mathcal{C}, \mathcal{V}, k \rangle$.

For the second case, assume $m < n$, and let $m' = m + 1$. We construct an instance $\langle \mathcal{C}', \mathcal{V}', k \rangle$ where $\mathcal{C}'$ has $n + m'$ inputs and $m' + 1$ outputs, and $\mathcal{V}'$ has $m' + 1$ inputs and $n + m'$ outputs, such that solutions are preserved. Applying this method inductively $n - m$ times yields an instance where $n = m'$, namely a EDGE-MAJORITY-BALANCED instance. We define $\langle \mathcal{C}', \mathcal{V}', k \rangle$ as follows. For $\mathbf{x} \in \{0, 1\}^n$ and $\mathbf{y}' \in \{0, 1\}^{m'}$, where $\mathbf{y}' = \mathbf{y}\mathbf{y}$ with $\mathbf{y} \in \{0, 1\}$ and $\mathbf{y} \in \{0, 1\}^m$, we set

$$\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}') = \mathbf{y}\mathcal{C}(\mathbf{x} \parallel \mathbf{y}). \quad (17)$$

Let $\mathbf{z}' \in \{0, 1\}^{m'+1}$, where $\mathbf{z}' = \mathbf{z}\mathbf{z}$ with $\mathbf{z} \in \{0, 1\}$ and $\mathbf{z} \in \{0, 1\}^m$. Denote $\mathbf{xy} = \mathcal{V}(\mathbf{z})$, where $\mathbf{x} \in \{0, 1\}^n$ and $\mathbf{y} \in \{0, 1\}^m$. We then set

$$\mathcal{V}'(\mathbf{z}') = \mathbf{x}\mathbf{z}\mathbf{y}. \quad (18)$$

Let $\mathbf{x}^* \in \{0, 1\}^n$ be a solution for EDGE-MAJORITY on $\langle \mathcal{C}, \mathcal{V}, k \rangle$, and let $\mathbf{y}' \in \{0, 1\}^{m'}$, where $\mathbf{y}' = \mathbf{y}\mathbf{y}$ with $\mathbf{y} \in \{0, 1\}$ and $\mathbf{y} \in \{0, 1\}^m$. We have $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$ and $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k$. Thus, we have that

$$\mathcal{V}'(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}')) = \mathcal{V}'(\mathbf{y}\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}\mathbf{y} = \mathbf{x}^*\mathbf{y}'$$

and also $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}') = \mathbf{y}\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq 0\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k$. Hence, $\mathbf{x}^*$ is a solution for EDGE-MAJORITY on $\langle \mathcal{C}', \mathcal{V}', k \rangle$.

Conversely, assume $\mathbf{x}^*$ is a solution for EDGE-MAJORITY on $\langle \mathcal{C}', \mathcal{V}', k \rangle$. Let $\mathbf{y} \in \{0, 1\}^m$. We have $0\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) = \mathcal{C}'(\mathbf{x}^*, 0\mathbf{y}) \geq k$, which implies that $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k$. Furthermore, we have $\mathcal{V}'(\mathcal{C}'(\mathbf{x}^*, 0\mathbf{y})) = \mathbf{x}^*0\mathbf{y}$, which, by definition of $\mathcal{V}'$, implies that $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$. Thus, $\mathbf{x}^*$ is a solution for EDGE-MAJORITY on $\langle \mathcal{C}, \mathcal{V}, k \rangle$. $\square$

Lemma C.2. *EDGE-MAJORITY-BALANCED[1] is PMA-complete.*

Proof. Clearly EDGE-MAJORITY-BALANCED[1] is in **PMA**, as an instance $\langle \mathcal{C}, \mathcal{V} \rangle$ can be reduced trivially to an instance $\langle \mathcal{C}, \mathcal{V}, 1 \rangle$ of EDGE-MAJORITY-BALANCED. For hardness, we show EDGE-MAJORITY-BALANCED[1] is PMA-hard. Let $\langle \mathcal{C}, \mathcal{V}, k \rangle$ be an instance of EDGE-MAJORITY-BALANCED with $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}^{n+1}$, $\mathcal{V}: \{0, 1\}^{n+1} \rightarrow \{0, 1\}^{2n}$, and $k \geq 1$. Construct the instance $\langle \mathcal{C}', \mathcal{V} \rangle$ of EDGE-MAJORITY-BALANCED[1], where $\mathcal{C}': \{0, 1\}^{2n} \rightarrow \{0, 1\}^{n+1}$ is defined as follows. For any $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$, let

$$\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = \begin{cases} \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) & \text{if } \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \geq k \\ 0^{n+1} & \text{if } \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) < k \end{cases}$$

Now, let $\mathbf{x}^*$ be a solution for EDGE-MAJORITY-BALANCED on $\langle \mathcal{C}, \mathcal{V}, k \rangle$. Then for all $\mathbf{y} \in \{0, 1\}^n$ we have $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k$ and $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$. Therefore, we have $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}) = \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k \geq 1$ and $\mathcal{V}(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$. Thus, $\mathbf{x}^*$ is a solution for EDGE-MAJORITY-BALANCED[1] on $\langle \mathcal{C}', \mathcal{V} \rangle$.

Conversely, let $\mathbf{x}^*$ be a solution for EDGE-MAJORITY-BALANCED[1] on $\langle \mathcal{C}', \mathcal{V} \rangle$. Then for all $\mathbf{y} \in \{0, 1\}^n$ we have that $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}) \geq 1$ and $\mathcal{V}'(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$. Let $\mathbf{y} \in \{0, 1\}^n$. If $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) < k$ then $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}) = 0^{n+1}$, a contradiction to the above. Thus $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq k$. Hence, by definition of $\mathcal{C}'$ we have that $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^*\mathbf{y}$. We conclude that $\mathbf{x}^*$ is a solution for EDGE-MAJORITY-BALANCED on $\langle \mathcal{C}, \mathcal{V}, k \rangle$. $\square$

We can now prove Theorem 5.5.

Proof. (Theorem 5.5). Since the case $k = 1$ was settled in Theorem C.2, let $k \geq 2$. Clearly, **PMA** contains EDGE-MAJORITY-BALANCED[k], as an instance $\langle \mathcal{C}, \mathcal{V} \rangle$ can be reduced trivially to an instance $\langle \mathcal{C}, \mathcal{V}, k \rangle$ of EDGE-MAJORITY-BALANCED. To show **PMA**-hardness, we prove that $\text{EDGE-MAJORITY-BALANCED}[1] \leq_P \text{EDGE-MAJORITY-BALANCED}[k]$, which suffices by Theorem C.2. Let $I = \langle \mathcal{C}, \mathcal{V} \rangle$ with $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}^{n+1}$ and $\mathcal{V}: \{0, 1\}^{n+1} \rightarrow \{0, 1\}^{2n}$. Without loss of generality assume $k \leq 2^n$ (otherwise we can determine whether it is a Yes- or No-instance in constant time, and construct a fixed Yes- or No-instance of EDGE-MAJORITY-BALANCED accordingly). Furthermore, assume that k is a power of two (otherwise, let k' be the smallest power of two greater than k ; the proof holds for k' , and EDGE-MAJORITY-BALANCED[k'] can be reduced to EDGE-MAJORITY-BALANCED[k] analogously to the proof of Theorem C.2).

To provide intuition, we begin with a proof sketch for the case $k = 2$. We define the instance $I' = \langle \mathcal{C}', \mathcal{V}' \rangle$ where $\mathcal{C}': \{0, 1\}^{2(n+1)} \rightarrow \{0, 1\}^{n+2}$ and $\mathcal{V}': \{0, 1\}^{n+2} \rightarrow \{0, 1\}^{2(n+1)}$. Since $\mathcal{C}'$ has one additional output bit compared with $\mathcal{C}$, its output space is twice as large, namely there are twice as many edge labels. For a label l in I we think of $0l$ and $1l$ as its copies in I' . Similarly, we can think of each edge as being duplicated: If $e := \mathbf{x}\mathbf{y} \in \{0, 1\}^{2n}$ is an edge in I (with $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$), then in

I' we consider $e' := 0x0y \in \{0,1\}^{2(n+1)}$ and $e'' := 0x1y \in \{0,1\}^{2(n+1)}$ as edges corresponding to e (we ignore the edge if x is prefixed with 1 rather than 0, but we allow both prefixes for y , and so we have two copies of e and not four).

We wish to define I' so that e is valid if and only if e' and e'' are. Thus, consider a label $l \in \{0,1\}^{n+1}$ of some edge e in I , and its copies $0l$ and $1l$ in I' . When $l \geq 2$ (interpreted as a binary number), we have that $1l \geq 2$ and $0l \geq 2$ (again, we interpret $0l$ and $1l$ as binary numbers, obtained by prefixing l with the bits 0 and 1 respectively). Therefore, both copies are within the acceptable range of labels of EDGE-MAJORITY-BALANCED[2], and we can set $\mathcal{C}'(e') = 0l$ and $\mathcal{C}'(e'') = 1l$ (and define $\mathcal{V}'$ accordingly).

When $l = 0^{n+1}$, we do not need to copy the label, as the edge e is invalid in I , so it suffices to set $\mathcal{C}'(e') = \mathcal{C}'(e'') = 0^{n+2}$ to ensure e' and e'' are invalid in I' . Thus, the label $1l = 10^{n+1}$ is currently available for future use.

However, when $l = 1$ (or in the general case, when $1 \leq l < k$), the label is valid in I , while not all of its copies are within the acceptable range in I' : We have that $0l < 2$ (prefixing with 0 the binary string representing the integer l still yields the same integer), while $1l$ is valid (prefixing l with 1 yields the integer $2^{n+1} + 1 > 2$). Therefore, we must find a valid label to replace $0l$. To do so, we exploit the fact that the copy $1 \parallel 0^{n+1}$ (representing the integer $2^n \geq 2$) of the label 0^{n+1} is available to replace $0l$.

It is not difficult to generalize this idea to any $k \geq 2$ value. Instead of adding one bit to the label space, we add $\log(k)$ bits, thus creating k copies of each edge and each label. As before, the copies of an invalid edge of I can all be mapped by $\mathcal{C}'$ to $0^{n+\log(k)}$ to guarantee they remain invalid. Hence, we have $k - 1$ copies of 0^{n+1} (all of which encode integers greater or equal to $2^{n+1} \geq k$) which can be used to replace the smallest copy of each label $l \in \{0,1\}^{n+1}$ with $1 \geq l < k$ (the smallest copy of l is always $0^{\log(k)}l$, which encodes an integer equal to l , and is thus invalid in I').

We proceed with the formal construction of $I' = \langle \mathcal{C}', \mathcal{V}' \rangle$. Let $k' = \log(k)$ (recall that k is a power of two). We define two circuits $\mathcal{C}': \{0,1\}^{2(n+k')} \rightarrow \{0,1\}^{n+k'+1}$ and $\mathcal{V}': \{0,1\}^{n+k'+1} \rightarrow \{0,1\}^{2(n+k')}$. Let $x', y' \in \{0,1\}^{n+k'}$. Denote $x' = x'_1 x'_2$ with $x'_1 \in \{0,1\}^{k'}$ and $x'_2 \in \{0,1\}^n$, and $y' = y'_1 y'_2$ with $y'_1 \in \{0,1\}^{k'}$ and $y'_2 \in \{0,1\}^n$. To define the behavior of the circuits, we make a case distinction based on the values of x'_1 , y'_1 , and $\mathcal{C}(x'_2 \parallel y'_2)$.

First, if $(x'_1 \neq 0^{k'}) \vee (\mathcal{C}(x'_2 \parallel y'_2) = 0^{n+1}) \vee (\mathcal{V}(\mathcal{C}(x'_2 \parallel y'_2)) \neq x'_2 y'_2)$, we let $\mathcal{C}'(x' \parallel y') = 0^{n+k'+1}$. The first condition ensures that only strings of the form $0^{k'} x'_2$ can be candidates for winners, while the second and third ensure that copies of invalid edges remain invalid in the reduced instance.

Now, assuming $(x'_1 = 0^{k'}) \wedge (\mathcal{C}(x'_2 \parallel y'_2) \neq 0^{n+1}) \wedge (\mathcal{V}(\mathcal{C}(x'_2 \parallel y'_2)) = x'_2 y'_2)$, let $l := \mathcal{C}(x'_2 \parallel y'_2)$ and $i := y'_1$. We define $\mathcal{C}'$ formally by case analysis. For the first case, assume $l \geq k$. Then we let $\mathcal{C}'(x' \parallel y') = i \parallel l$ (where now we interpret i and l as bit strings and concatenate them, obtaining the i -th copy of label l).

For the second case, assume $1 \leq l < k$. Then, if $i \neq 0^{k'}$ we let $\mathcal{C}'(x' \parallel y') = i \parallel l$ as before. If $i = 0^{k'}$, let l' be the k' -bit representation of the binary number l (k' bits suffice, since $l < k = 2^{k'}$). Then we set $\mathcal{C}'(x' \parallel y') = l' \parallel 0^{n+1}$. Namely, it is the l' -th copy of 0^{n+1} (notice how l' now defines the copy index instead of the label, since the 0-th copy of the label l is invalid, and so we need to refer to the unused copies of 0^{n+1}).

The circuit $\mathcal{V}'$ will be defined as follows. On input $i \parallel l$, where $i \in \{0,1\}^{k'}$ and $l \in \{0,1\}^{n+1}$, consider the following cases. If $l \neq 0^{n+1}$ (i.e., $i \parallel l$ is not a copy of 0^{n+1}), let $xy := \mathcal{V}(l)$ with $x, y \in \{0,1\}^n$. We then define $\mathcal{V}'(i \parallel l) = 0^{k'} x i y$. If $l = 0^{n+1}$ (i.e., $i \parallel l$ is a copy of 0^{n+1}), let i' be the $(n+1)$ -bit representation of the integer i , and let $xy := \mathcal{V}(i')$ where $x, y \in \{0,1\}^n$. Then, define $\mathcal{V}'(i \parallel l) = 0^{k'} x 0^{k'} y$ (this is the smallest copy of the edge xy).

We proceed with proving correctness of the reduction. Suppose $x^* \in \{0,1\}^n$ is a solution for the instance I of EDGE-MAJORITY-BALANCED[1]. Then for all $y \in \{0,1\}^n$ we have $\mathcal{C}(x^* \parallel y) > 0$ and

$\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^* \mathbf{y}$. We wish to show $0^{k'} \mathbf{x}^* \in \{0, 1\}^{n+k'}$ is a solution for I' . Let $\mathbf{y}'_1 \parallel \mathbf{y}'_2 \in \{0, 1\}^{n+k'}$, where $\mathbf{y}'_1 \in \{0, 1\}^{k'}$ and $\mathbf{y}'_2 \in \{0, 1\}^n$, and denote $l := \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}'_2) > 0$. We make the following case distinction.

If $l \geq k$, then by definition we have $\mathcal{C}'(0^{k'} \mathbf{x}^* \mathbf{y}'_1 \mathbf{y}'_2) = \mathbf{y}'_1 \parallel l \geq k$, and furthermore $\mathcal{V}'(\mathcal{C}'(0^{k'} \mathbf{x}^* \mathbf{y}'_1 \mathbf{y}'_2)) = \mathcal{V}'(\mathbf{y}'_1 \parallel l) = 0^{k'} \mathbf{x}^* \mathbf{y}'_1 \mathbf{y}'_2$, and thus $0^{k'} \mathbf{x}^*$ is a solution for EDGE-MAJORITY-BALANCED[k] on $\langle \mathcal{C}', \mathcal{V}' \rangle$.

If $1 \leq l < k$ and $\mathbf{y}'_1 \neq 0^{k'}$, then we have $\mathcal{C}'(0^{k'} \mathbf{x}^* \parallel \mathbf{y}'_1 \mathbf{y}'_2) = \mathbf{y}'_1 \parallel l \geq 2^{n+1} \geq k$ (the prefix $\mathbf{y}'_1$ ensures the binary interpretation of this string is at least 2^{n+1} , and we assumed $k \leq 2^n$). Furthermore, we have that $\mathcal{V}'(\mathcal{C}'(0^{k'} \mathbf{x}^* \parallel \mathbf{y}'_1 \mathbf{y}'_2)) = \mathcal{V}'(\mathbf{y}'_1 \parallel l) = 0^{k'} \mathbf{x}^* \mathbf{y}'_1 \mathbf{y}'_2$.

If $1 \leq l < k$ and $\mathbf{y}'_1 = 0^{k'}$, let l' be the k' -bit representation of the binary number l . Then by definition we have $\mathcal{C}'(0^{k'} \mathbf{x}^* \parallel \mathbf{y}'_1 \mathbf{y}'_2) = l' \parallel 0^{n+1} \geq 2^{n+1} \geq k$ (since $l' > 0$, the prefix l' ensures the binary interpretation of this string is at least 2^{n+1}). Furthermore, we have that $\mathcal{V}'(\mathcal{C}'(0^{k'} \mathbf{x}^* \parallel \mathbf{y}'_1 \mathbf{y}'_2)) = \mathcal{V}'(l' \parallel 0^{n+1}) = 0^{k'} \mathbf{x}^* 0^{k'} \mathbf{y}'_2 = 0^{k'} \mathbf{x}^* \mathbf{y}'_1 \mathbf{y}'_2$.

Hence, we have that $0^{k'} \mathbf{x}^*$ is a solution for EDGE-MAJORITY-BALANCED[k] on $\langle \mathcal{C}', \mathcal{V}' \rangle$.

Conversely, suppose $0^{k'} \mathbf{x}^* \in \{0, 1\}^{n+k'}$ is a solution for EDGE-MAJORITY-BALANCED[k] on $\langle \mathcal{C}', \mathcal{V}' \rangle$ (a solution must have a $0^{k'}$ prefix as otherwise $\mathcal{C}'$ maps it to $0^{n+k'+1}$ with any $\mathbf{y}' \in \{0, 1\}^{n+k'}$). Let $\mathbf{y} \in \{0, 1\}^n$, and denote $l := \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})$. If $l = 0^{n+1}$, then clearly $\mathcal{C}'(0^{k'} \mathbf{x}^* \parallel \mathbf{y}' \mathbf{y}) = 0^{n+k'+1}$ for any $\mathbf{y}' \in \{0, 1\}^{k'}$, a contradiction to $0^{k'} \mathbf{x}^*$ being a solution for $\langle \mathcal{C}', \mathcal{V}' \rangle$. Hence, $l \neq 0^{n+1}$.

Consider the string $0^{k'} \mathbf{y}$. We make the following case distinction.

If $l \geq k$ then we have that

$$0^{k'} \mathbf{x}^* 0^{k'} \mathbf{y} = \mathcal{V}'(\mathcal{C}'(0^{k'} \mathbf{x}^*, 0^{k'} \mathbf{y})) = \mathcal{V}'(0^{k'} \parallel l) \quad (19)$$

(the former equality is due to the definition of a solution, and the latter by definition of $\mathcal{C}'$). On the other hand, denoting $\mathbf{x}_l \mathbf{y}_l = \mathcal{V}(l)$ with $\mathbf{x}_l, \mathbf{y}_l \in \{0, 1\}^n$, by definition of $\mathcal{V}'$ we have that

$$\mathcal{V}'(0^{k'} \parallel l) = 0^{k'} \mathbf{x}_l 0^{k'} \mathbf{y}_l \quad (20)$$

By Equations (19) and (20) we have that $\mathbf{x}_l = \mathbf{x}^*$ and $\mathbf{y}_l = \mathbf{y}$. Hence, $\mathbf{x}^* \mathbf{y} = \mathbf{x}_l \mathbf{y}_l = \mathcal{V}(l) = \mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}))$.

If $l < k$, let l' be the k' -bit representation of the binary number l . By similar reasoning to Equation (19), we have that

$$0^{k'} \mathbf{x}^* 0^{k'} \mathbf{y} = \mathcal{V}'(\mathcal{C}'(0^{k'} \mathbf{x}^*, 0^{k'} \mathbf{y})) = \mathcal{V}'(l' \parallel 0^{n+1}) \quad (21)$$

On the other hand, denoting $\mathbf{x}_l \mathbf{y}_l = \mathcal{V}(l)$ with $\mathbf{x}_l, \mathbf{y}_l \in \{0, 1\}^n$, by definition we have that

$$\mathcal{V}'(l' \parallel 0^{n+1}) = 0^{k'} \mathbf{x}_l 0^{k'} \mathbf{y}_l \quad (22)$$

By Equations (21) and (22) we have that $\mathbf{x}_l = \mathbf{x}^*$ and $\mathbf{y}_l = \mathbf{y}$. Hence, $\mathbf{x}^* \mathbf{y} = \mathbf{x}_l \mathbf{y}_l = \mathcal{V}(l) = \mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}))$.

Therefore, $\mathbf{x}^*$ is a solution for $\langle \mathcal{C}, \mathcal{V} \rangle$. Therefore, the reduction preserves satisfiability, completing the proof of **PMA**-hardness. $\square$

Proof. (Theorem 5.6). Showing **PMA**-hardness of EDGE-MAJORITY-SET is straightforward, as it can be reduced from EDGE-MAJORITY-BALANCED[1] trivially by specifying the set $\{0\}$ of invalid edge labels, while keeping the Boolean circuit untouched. For inclusion in **PMA**, we now show that EDGE-MAJORITY-SET $\leq_P$ EDGE-MAJORITY-BALANCED. We begin by introducing the following notation. For a set $T \subseteq \mathbb{N}_0$ and an element $t \in T$, denote by $idx_T(t) \in [|T|]_0$ the index of t in the list $ordered(T)$, where $ordered(T)$ is the sequence of elements of T sorted in increasing order.

Conversely, for $i \in [|T|]_0$ denote by $T(i) \in T$ the i -th element of $\text{ordered}(T)$. Notice that both $\text{idx}_T(t)$ and $T(i)$ can be computed in time polynomial in $|T|$ (and in $\log(t')$ for all $t' \in T$).

Now, let $\langle \mathcal{C}, \mathcal{V}, S \rangle$ be an instance of EDGE-MAJORITY-SET, where $\mathcal{C}$ has $2n$ inputs and $n+1$ outputs, and $\mathcal{V}$ the opposite. Denote $k := |S|$, $K = [k]_0$, $S' := S \setminus K$, and $K' := K \setminus S$. Notice that $|S| = |K|$, and thus $|S'| = |K'|$. We construct the instance $\langle \mathcal{C}', \mathcal{V}', k \rangle$ of EDGE-MAJORITY-BALANCED, where $\mathcal{C}'$ and $\mathcal{V}'$ are defined as follows. The number of inputs and outputs of $\mathcal{C}'$ and $\mathcal{V}'$ match those of $\mathcal{C}$ and $\mathcal{V}$ respectively. For $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$, define:

$$\mathcal{C}'(\mathbf{x} \parallel \mathbf{y}) = \begin{cases} \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) & \text{if } \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \in S \cap K \text{ or } \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \notin S \cup K \\ \text{idx}_{S'}(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) & \text{if } \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \in S' \\ S'(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) & \text{if } \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \in K' \end{cases}$$

For $\mathbf{z} \in \{0, 1\}^{n+1}$, define:

$$\mathcal{V}'(\mathbf{z}) = \begin{cases} \mathcal{V}(\mathbf{z}) & \text{if } \mathbf{z} \in S \cap K \text{ or } \mathbf{z} \notin S \cup K \\ \mathcal{V}(\text{idx}_{S'}(\mathbf{z})) & \text{if } \mathbf{z} \in S' \\ \mathcal{V}(S'(\mathbf{z})) & \text{if } \mathbf{z} \in K' \end{cases}$$

Suppose $\mathbf{x}^* \in \{0, 1\}^n$ is a solution for EDGE-MAJORITY-SET on $\langle \mathcal{C}, \mathcal{V}, S \rangle$. Let $\mathbf{y} \in \{0, 1\}^n$. We have that $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^* \mathbf{y}$ and $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \notin S$. If $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \notin S \cup K$, then $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}) \geq k$, and furthermore $\mathcal{V}'(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}'(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^* \mathbf{y}$. Otherwise, we have that $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \in K'$, and thus $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}) = S'(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) \in S'$. Hence, we have that $\mathcal{V}'(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}'(S'(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}))) = \mathcal{V}(\text{idx}_{S'}(S'(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})))) = \mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^* \mathbf{y}$. Thus, we have that $\mathbf{x}^*$ is a solution for EDGE-MAJORITY-BALANCED on $\langle \mathcal{C}', \mathcal{V}' \rangle$.

Conversely, suppose $\mathbf{x}^* \in \{0, 1\}^n$ is a solution for EDGE-MAJORITY-BALANCED on $\langle \mathcal{C}', \mathcal{V}', k \rangle$. Let $\mathbf{y} \in \{0, 1\}^n$. Then $\mathcal{V}'(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^* \mathbf{y}$ and $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}) \notin K$. If $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \in S'$ then $\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y}) = \text{idx}_{S'}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) \in K$, a contradiction. Hence, we have that $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \notin K \cup S' = S$. Additionally, this implies that $\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \notin S \cup K$. Thus, we have that $\mathcal{V}(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}'(\mathcal{C}(\mathbf{x}^* \parallel \mathbf{y})) = \mathcal{V}'(\mathcal{C}'(\mathbf{x}^* \parallel \mathbf{y})) = \mathbf{x}^* \mathbf{y}$. Therefore, $\mathbf{x}^*$ is a solution for EDGE-MAJORITY-SET on $\langle \mathcal{C}, \mathcal{V}, S \rangle$. $\square$

Proof. (Theorem 5.2). It is clear that $\mathbf{PMA} \subseteq \Sigma_2^P$, since the condition of EDGE-MAJORITY-BALANCED can be checked in polynomial time for any pair of strings $\mathbf{x}$ and $\mathbf{y}$. Thus, by Theorem 5.5 it suffices to show that any instance $\langle \mathcal{C}, \mathcal{V} \rangle$ of EDGE-MAJORITY-BALANCED[1] may have at most one solution. Assume towards contradiction there exist two strings $\mathbf{x}_1, \mathbf{x}_2 \in \{0, 1\}^n$ where $\mathbf{x}_1 \neq \mathbf{x}_2$ such that for all $\mathbf{y} \in \{0, 1\}^n$ and $\mathbf{x} \in \{\mathbf{x}_1, \mathbf{x}_2\}$ we have $\mathcal{V}(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) = \mathbf{x} \mathbf{y} \wedge \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \neq 0^{n+1}$. Then we have $\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y}) \in \{0, 1\}^{n+1} \setminus 0^{n+1}$ and $\mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y}) \in \{0, 1\}^{n+1} \setminus 0^{n+1}$ for all $\mathbf{y} \in \{0, 1\}^n$. Furthermore, since $\mathbf{x}_1 \neq \mathbf{x}_2$, we have $|\{\mathbf{x}_1 \mathbf{y} : \mathbf{y} \in \{0, 1\}^n\} \cup \{\mathbf{x}_2 \mathbf{y} : \mathbf{y} \in \{0, 1\}^n\}| = 2^n + 2^n = 2^{n+1}$. Thus, by the Pigeonhole Principle, there exists $\mathbf{y}' \in \{0, 1\}^n$ such that $\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y}') = \mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y}')$. Therefore, we must have that either $\mathcal{V}(\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y}')) \neq \mathbf{x}_1 \mathbf{y}'$ or $\mathcal{V}(\mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y}')) \neq \mathbf{x}_2 \mathbf{y}'$, a contradiction to the definition of $\mathbf{x}_1$ or $\mathbf{x}_2$. $\square$

Proof. (Theorem 5.3). To prove this, we show that $\text{EDGE-MAJORITY-BALANCED}[1] \in \mathbf{S}_2^P$. Let $I = \langle \mathcal{C}, \mathcal{V} \rangle$ be an instance of EDGE-MAJORITY-BALANCED[1]. For a pair $t = (\mathbf{x}, \mathbf{y})$ where $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$, let us say that t is *verified* if $\mathcal{V}(\mathcal{C}(\mathbf{x} \parallel \mathbf{y})) = \mathbf{x} \parallel \mathbf{y} \wedge \mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \neq 0^{n+1}$.

Consider the bipartite graph $G = (X, Y, E)$ with $X = Y = \{0, 1\}^n$ and $E = \{(\mathbf{x}, \mathbf{y}) \in X \times Y : (\mathbf{x}, \mathbf{y}) \text{ is verified}\}$. Indeed, a vertex has out-degree 2^n in G if and only if it is a winner in I . Since $\mathcal{V}$ has $n+1$ inputs, and any pair $(\mathbf{x}, \mathbf{y})$ that is mapped to 0^{n+1} by $\mathcal{C}$ is not verified by design, there are at most $2^{n+1} - 1$ verified pairs; this is also an upper bound on $|E|$. If all vertices in Y have

in-degree at least two, then we get $|E| \geq 2^{n+1}$, a contradiction to the above. Hence, there exists a vertex $y' \in Y$ with in-degree at most one.

Now, for $x \in V$ and $S \subseteq V$ with $|S| = 2$, define the polynomial-time predicate

$$P(G, S, x) = \begin{cases} 1 & \text{if } \forall y \in S \ (x, y) \in E \\ 0 & \text{otherwise} \end{cases}$$

If I is a No-instance, namely all vertices in X have out-degree at most $2^n - 1$, we show that there exists a set $S = \{y_1, y_2\}$ where $y_1, y_2 \in \{0, 1\}^n$, such that for all x we have $P(G, S, x) = 0$, namely either $(x, y_1) \notin E$ or $(x, y_2) \notin E$. Indeed, we can construct such a set S as follows. Take y' with $\text{in}_G(y') \leq 1$ as discussed above. If $\text{in}_G(y') = 1$, namely there exists a vertex x'' such that $(x'', y') \in E$, then pick $y'' \in V$ such that $(x'', y'') \notin E$ (there must exist such y'' since I is a No-instance). If $\text{in}_G(y') = 0$ then any arbitrary y'' will do. The set $S = \{y', y''\}$ then satisfies the above property.

Conversely, if I is a Yes-instance with a solution $x^* \in \{0, 1\}^n$, then for all sets $S = \{y_1, y_2\}$ we have that $P(G, S, x^*) = 1$. Hence, we conclude that $\text{EDGE-MAJORITY-BALANCED}[1] \in \mathbf{S}_2^P$. $\square$

Our next goal is to show a lower bound for **PMA**, and specifically we show that it contains **coNP**. To do so, we reduce from the canonical **coNP** version of SAT, defined as follows.

Problem C.2:
UNSAT
Input: A set $\mathcal{X} = \{x_1, \dots, x_n\}$ of Boolean variables and a Boolean formula $\psi(\mathcal{X})$ over $\mathcal{X}$.
Question: $\forall x \in \{0, 1\}^n \ \psi(x) = 0$?

Proof. (Theorem 5.4). We show that $\text{UNSAT} \leq_P \text{EDGE-MAJORITY-BALANCED}[1]$, implying any **coNP** problem is reducible to **EDGE-MAJORITY** and is thus in **PMA**. Given an instance ψ of UNSAT, we construct the instance $\langle \mathcal{C}, \mathcal{V} \rangle$ of **EDGE-MAJORITY-BALANCED**[1], defined by:

$$\forall x, y \in \{0, 1\}^n \ \mathcal{C}(x \parallel y) = \begin{cases} 1y & \text{if } x = 0^n \wedge \psi(y) = 0 \\ 0^{n+1} & \text{otherwise} \end{cases} \quad (23)$$

$$\forall z \in \{0, 1\}^{n+1}, \text{ where } z = zz' \text{ with } z \in \{0, 1\} \text{ and } z' \in \{0, 1\}^n, \ \mathcal{V}(z) = 0^n z' \quad (24)$$

First, suppose ψ is a Yes-instance of UNSAT. We wish to show 0^n is a solution for **EDGE-MAJORITY-BALANCED**[1] on $\langle \mathcal{C}, \mathcal{V} \rangle$. Let $y \in \{0, 1\}^n$. Since ψ is a Yes-instance, we have $\psi(y) = 0$, and therefore we have $\mathcal{C}(0^n \parallel y) = 1y \neq 0^{n+1}$. Furthermore, we have $\mathcal{V}(\mathcal{C}(0^n \parallel y)) = \mathcal{V}(1y) = 0^n y$. Therefore, 0^n is a solution for **EDGE-MAJORITY-BALANCED**[1] on $\langle \mathcal{C}, \mathcal{V} \rangle$ by definition.

Conversely, suppose $\langle \mathcal{C}, \mathcal{V} \rangle$ is a Yes-instance of **EDGE-MAJORITY-BALANCED**[1], and let x^* be its solution. Assume towards contradiction ψ is a No-instance, namely there exists $y' \in \{0, 1\}^n$ such that $\psi(y') = 1$. Then we have $\mathcal{C}(x^* \parallel y') = 0^{n+1}$, and therefore x^* is not a solution for **EDGE-MAJORITY-BALANCED**[1] on $\langle \mathcal{C}, \mathcal{V} \rangle$, a contradiction. $\square$

D Proofs of Section 6 (Δ_2^P)

In this section, we provide all proofs missing from Section 6. For our reductions, we build on the result of [Pap84] that determining the existence of a unique optimal Traveling Salesman Problem tour is Δ_2^P -complete; this was the first problem shown to be complete for Δ_2^P , after which several others have been identified (see, e.g., [Kre86]).

Problem D.1:

TSP-UNIQUE-OPT

Input: Graph G given as an adjacency matrix.**Question:** Is there a unique minimal-weight Hamiltonian cycle in G ?**Theorem D.1.** ([Pap84]). *TSP-UNIQUE-OPT is Δ_2^P -complete.*

Before addressing the main results of this section, we define a problem we denote CKT-UNIQUE-OPT. This is a more general Boolean-circuit formulation of TSP-UNIQUE-OPT, which we show is also Δ_2^P -complete. It will be convenient to use this formulation for the reductions throughout this section.

Problem D.2:

CKT-UNIQUE-OPT

Input: Boolean circuit $\mathcal{C}$, with n inputs and n outputs.**Question:** $\exists \mathbf{x} \in \{0, 1\}^n \forall \mathbf{x}' \in \{0, 1\}^n$ with $\mathbf{x}' \neq \mathbf{x} \mathcal{C}(\mathbf{x}) > \mathcal{C}(\mathbf{x}')$?**Lemma D.2.** *CKT-UNIQUE-OPT is Δ_2^P -complete.*

Proof. It is immediate from Theorem D.1 that CKT-UNIQUE-OPT is Δ_2^P -hard: Given a TSP instance we construct a circuit $\mathcal{C}$ that, given a string representing some permutation of the vertices, calculates the length of the Hamiltonian tour induced by this permutation, and multiplies it by -1 . Thus, there exists a uniquely minimal-weight TSP tour if and only if there exists a unique $\mathbf{x}$ maximizing the value $\mathcal{C}(\mathbf{x})$.

As for containment in Δ_2^P , we can use an algorithm analogous to the one by [Pap84] showing TSP-UNIQUE-OPT is in Δ_2^P : Use an **NP** oracle to obtain, with binary search, the optimal value obtained by $\mathcal{C}$. Use the **NP** oracle to find a string obtaining the optimal value, and an additional query to check if there is an optimizing string different from the one found. If the first string is the only optimizer, then the answer is Yes, otherwise it is No. $\square$

Proof. (**Theorem 6.1, CKT-CONDORCET[2]**). To show CKT-CONDORCET[2] is Δ_2^P -hard, we reduce from CKT-UNIQUE-OPT which, by Theorem D.2, suffices. Given a CKT-UNIQUE-OPT instance $\mathcal{C}$, we construct a circuit $\mathcal{C}'$ that outputs 0 for any input. Thus, a string $\mathbf{x}$ is a unique optimizer of $\mathcal{C}(\mathbf{x})$ if and only if it is a Condorcet string in $\langle \mathcal{C}, \mathcal{C}' \rangle$ (since the circuit $\mathcal{C}'$ is always indifferent between any two strings).

To show that CKT-CONDORCET[2] is in Δ_2^P , we construct a polynomial-time algorithm A that uses an **NP** oracle O . Given two circuits $\langle \mathcal{C}_1, \mathcal{C}_2 \rangle$, A works as follows.

1. Use O to obtain, with binary search, the optimal values o_1 and o_2 obtained by $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively.
2. Use O to find $\mathbf{x}$ such that $\mathcal{C}_1(\mathbf{x}) = o_1$ and $\mathcal{C}_2(\mathbf{x}) = o_2$, or determine there is none.
3. Use O to find $\mathbf{x}' \neq \mathbf{x}$, such that $\mathcal{C}_1(\mathbf{x}') = o_1$ and $\mathcal{C}_2(\mathbf{x}') = o_2$, or determine there is none.
4. If $\mathbf{x}$ exists but $\mathbf{x}'$ does not, output 1, otherwise 0.

We wish to prove that $A(\mathcal{C}_1, \mathcal{C}_2) = 1$ if and only if there exists a Condorcet string $\mathbf{x}$ in $\langle \mathcal{C}_1, \mathcal{C}_2 \rangle$. First, assume $A(\mathcal{C}_1, \mathcal{C}_2) = 1$. Then there is exactly one string $\mathbf{x}$ that optimizes both circuits. Let $\mathbf{x}' \neq \mathbf{x}$. Since $\mathbf{x}$ achieves the optimal value in both circuits, we have $\mathcal{C}_1(\mathbf{x}) \geq \mathcal{C}_1(\mathbf{x}')$ and $\mathcal{C}_2(\mathbf{x}) \geq \mathcal{C}_2(\mathbf{x}')$. Furthermore, since $\mathbf{x}$ is the unique string optimizing both circuits, one of these inequalities must be strict. Since this holds for any string $\mathbf{x}' \neq \mathbf{x}$, we have that $\mathbf{x}$ is a Condorcet string.

In the other direction, assume $\mathbf{x}$ is a Condorcet string. Then it must achieve the optimal value in both circuits, since if there existed a string $\mathbf{x}'$ with $\mathcal{C}_1(\mathbf{x}') > \mathcal{C}_1(\mathbf{x})$ or $\mathcal{C}_2(\mathbf{x}') > \mathcal{C}_2(\mathbf{x})$, then $\mathbf{x}$ is not a Condorcet string, a contradiction. Furthermore, if there exists another string $\mathbf{x}'$ that optimizes both circuits, then clearly $\mathbf{x}$ and $\mathbf{x}'$ tie on both circuits, a contradiction to $\mathbf{x}$ being a Condorcet string. Hence, $\mathbf{x}$ must be the only string optimizing both circuits, implying $A(\mathcal{C}_1, \mathcal{C}_2) = 1$. $\square$

Proof. (**Theorem 6.1, CKT-CONSENSUS**). By Theorem D.2, we have that CKT-CONSENSUS is Δ_2^P -hard because CKT-UNIQUE-OPT is a special case of CKT-CONSENSUS in which there is only one circuit.

We wish to show that CKT-CONSENSUS $\in \Delta_2^P$. Suppose we are given a CKT-CONSENSUS instance $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$. By definition, $\mathbf{x}$ is a solution for $\mathfrak{C}$ if and only if it is a unique optimizer for all circuits $\mathcal{C}_i$. In particular, a solution would optimize $\mathcal{C}_1$. Thus, it suffices to find a string $\mathbf{x}^*$ optimizing $\mathcal{C}_1$ and use an additional **NP** query to check if there is a circuit not optimized by $\mathbf{x}^*$. This is formalized in the following algorithm, which uses an **NP** oracle O .

1. Use O to find, with binary search, the optimal value of $\mathcal{C}_1$.
2. Use O to find a string $\mathbf{x}^*$ optimizing $\mathcal{C}_1$.
3. Use a single call to O to answer the following:

$$\exists \mathbf{x}', i \text{ such that } \mathcal{C}_i(\mathbf{x}') \geq \mathcal{C}_i(\mathbf{x}^*)$$

4. Output the opposite of the previous query's result.

$\square$

Proof. (**Theorem 6.1, STRONG-DOMINANT-STRATEGY**). To show that STRONG-DOMINANT-STRATEGY is Δ_2^P -hard, we reduce from CKT-UNIQUE-OPT. Given a circuit $\mathcal{C}$ with n input bits, we construct a circuit $\mathcal{C}'$ with $2n$ input bits, which on input $(\mathbf{x} \parallel \mathbf{y})$ with $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$, ignores $\mathbf{y}$ and simply outputs $\mathcal{C}(\mathbf{x})$. Correctness follows from the definitions.

We now wish to show STRONG-DOMINANT-STRATEGY is in Δ_2^P . Given a circuit $\mathcal{C}$ for STRONG-DOMINANT-STRATEGY, we construct an algorithm which is essentially similar to the one in the Δ_2^P -completeness proof of CKT-CONSENSUS (Theorem 6.1).

1. Fix the string 0^n , and use an **NP** oracle to find, with binary search, the optimal value of $\mathcal{C}(\mathbf{x} \parallel 0^n)$.
2. Use the oracle to find a string $\mathbf{x}^*$ optimizing $\mathcal{C}(\mathbf{x} \parallel 0^n)$.
3. Use a single call to the oracle to answer the following:

$$\exists \mathbf{x}', \mathbf{y} \text{ such that } \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}) \geq \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}).$$

4. Output the opposite of the previous query's result.

$\square$

Proof. (**Theorem 6.1, CKT-WINNER-THRESHOLD**). We first show that CKT-WINNER-THRESHOLD $\in \Delta_2^P$. Given an instance $\mathcal{C}: \{0, 1\}^{2n} \rightarrow \{0, 1\}$, call a string $\mathbf{x}$ an *all-winner* if $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 1$ for any $\mathbf{y} \in \{0, 1\}^n$, and an *all-loser* if $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = 0$ for any $\mathbf{y} \in \{0, 1\}^n$. We construct the following algorithm using an **NP** oracle O .

1. Using O , find with binary search the smallest string $\mathbf{x}^* \in \{0, 1\}^n$ which satisfies:

$$\forall \mathbf{x}', \mathbf{y} \in \{0, 1\}^n \quad \mathbf{x}' \leq \mathbf{x}^* \vee \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}) = 0.$$

Namely, $\mathbf{x}^*$ is the minimal string such that any string above it is an all-loser. Observe that the largest n -bit string, $2^n - 1$, trivially satisfies this condition, and therefore $\mathbf{x}^*$ is well defined.

2. Use a single query to O asking whether $\forall \mathbf{y} \quad \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) = 1$; if so answer Yes, otherwise No.

We show that the algorithm returns Yes if and only if $\mathcal{C}$ is a Yes-instance. Suppose there exists a solution $\mathbf{x}$ for the instance $\mathcal{C}$. In particular, $\mathbf{x}$ is the smallest string such that all strings above it are all-losers (there cannot be a smaller string satisfying this, since the solution itself is not an all-loser). Hence, the string $\mathbf{x}^*$ found in Step 1 is the solution for $\mathcal{C}$. Therefore, the condition of Step 2 is satisfied by $\mathbf{x}^*$ and the algorithm returns Yes.

Conversely, assume the algorithm returns Yes. Consider the string $\mathbf{x}^*$ found in Step 1. All strings above $\mathbf{x}^*$ are all-losers (by the condition of Step 1), and $\mathbf{x}^*$ itself is an all-winner (by the condition of Step 2). Hence, $\mathbf{x}^*$ is a solution for $\mathcal{C}$.

To show CKT-WINNER-THRESHOLD is Δ_2^P -hard, we reduce from CKT-UNIQUE-OPT. Given an instance $\mathcal{C}$ of CKT-UNIQUE-OPT, we construct a circuit $\mathcal{C}'$, defined for every $\mathbf{y}, \mathbf{x}, \mathbf{y}', \mathbf{x}' \in \{0, 1\}^n$ by

$$\mathcal{C}'(\mathbf{y}\mathbf{x} \parallel \mathbf{y}'\mathbf{x}') = \begin{cases} 1 & \text{if } \mathcal{C}(\mathbf{x}) = \mathbf{y} \wedge (\mathbf{y} > \mathbf{y}' \vee \mathcal{C}(\mathbf{x}') \neq \mathbf{y}') \\ 0 & \text{otherwise} \end{cases}$$

We show $\mathcal{C}$ is a Yes-instance of CKT-UNIQUE-OPT if and only if $\mathcal{C}'$ is a Yes-instance of CKT-WINNER-THRESHOLD. Assume $\mathcal{C}$ is a Yes-instance. Then there exists $\mathbf{x}^* \in \{0, 1\}^n$ such that $\mathcal{C}(\mathbf{x}^*) > \mathcal{C}(\mathbf{x})$ for any $\mathbf{x} \neq \mathbf{x}^*$. Denoting $\mathbf{y}^* := \mathcal{C}(\mathbf{x}^*)$, we show that $\mathbf{y}^*\mathbf{x}^*$ is a solution for $\mathcal{C}'$.

Let $\mathbf{y}', \mathbf{x}' \in \{0, 1\}^n$ such that $\mathbf{y}'\mathbf{x}' \neq \mathbf{y}^*\mathbf{x}^*$. We wish to prove that $\mathbf{y}^*\mathbf{x}^*$ is an all-winner, and that if $\mathbf{y}'\mathbf{x}' > \mathbf{y}^*\mathbf{x}^*$ then $\mathbf{y}'\mathbf{x}'$ is an all-loser. For the first part, we make the following case distinction. If $\mathbf{x}' \neq \mathbf{x}^*$ then we have that $\mathcal{C}'(\mathbf{y}^*\mathbf{x}^* \parallel \mathbf{y}'\mathbf{x}') = 1$. To see why, observe that we either have that $\mathcal{C}(\mathbf{x}') \neq \mathbf{y}'$ or $\mathcal{C}(\mathbf{x}') = \mathbf{y}'$, where in the latter case we must have $\mathbf{y}^* > \mathbf{y}'$ since $\mathbf{y}^*$ is the uniquely-achieved optimum of $\mathcal{C}$. If $\mathbf{x}' = \mathbf{x}^*$ then we must have that $\mathbf{y}' \neq \mathbf{y}^*$ (otherwise $\mathbf{y}'\mathbf{x}' = \mathbf{y}^*\mathbf{x}^*$, a contradiction). Since $\mathbf{y}^* = \mathcal{C}(\mathbf{x}^*)$, we have that $\mathbf{y}' \neq \mathcal{C}(\mathbf{x}^*) = \mathcal{C}(\mathbf{x}')$, once again implying that $\mathcal{C}'(\mathbf{y}^*\mathbf{x}^* \parallel \mathbf{y}'\mathbf{x}') = 1$.

For the second part, suppose $\mathbf{y}'\mathbf{x}' > \mathbf{y}^*\mathbf{x}^*$. Then, in particular we have that $\mathbf{y}' \geq \mathbf{y}^*$ (as $\mathbf{y}'$ and $\mathbf{y}^*$ comprise of the more significant bits of $\mathbf{y}'\mathbf{x}'$ and $\mathbf{y}^*\mathbf{x}^*$ respectively). Therefore, since $\mathbf{y}^*$ is the optimal value of $\mathcal{C}$, and it is uniquely achieved, we must have $\mathcal{C}(\mathbf{x}') \neq \mathbf{y}'$. Thus, we have $\mathcal{C}'(\mathbf{y}'\mathbf{x}' \parallel \mathbf{y}''\mathbf{x}'') = 0$ for any $\mathbf{y}'', \mathbf{x}'' \in \{0, 1\}^n$, namely $\mathbf{y}'\mathbf{x}'$ is an all-loser. We conclude that $\mathbf{y}^*\mathbf{x}^*$ is a solution for CKT-WINNER-THRESHOLD on $\mathcal{C}$.

Conversely, let $\mathbf{x}^*, \mathbf{y}^* \in \{0, 1\}^n$ such that $\mathbf{y}^*\mathbf{x}^*$ is a solution for CKT-WINNER-THRESHOLD on $\mathcal{C}'$. In particular, we have that $\mathcal{C}'(\mathbf{y}^*\mathbf{x}^* \parallel \mathbf{y}\mathbf{x}) = 1$ for all $\mathbf{y}, \mathbf{x} \in \{0, 1\}^n$, implying $\mathcal{C}(\mathbf{x}^*) = \mathbf{y}^*$. Assume towards contradiction that there exists $\mathbf{x}' \in \{0, 1\}^n$ such that $\mathbf{y}' := \mathcal{C}(\mathbf{x}') \geq \mathcal{C}(\mathbf{x}^*)$. Then we have that $\mathcal{C}'(\mathbf{y}^*\mathbf{x}^* \parallel \mathbf{y}'\mathbf{x}') = 0$, a contradiction to the choice of $\mathbf{y}^*\mathbf{x}^*$. Hence, $\mathbf{x}^*$ is the unique optimizer of $\mathcal{C}$. $\square$

E Proofs of Section 7

In this section, we provide all proofs missing from Section 7. We begin with a definition of a canonical Σ_2^P satisfiability problem proved to be Σ_2^P -complete by [Sto76], which we will then use for our reductions.

Problem E.1: $\exists\forall$ -SAT

Input: Two sets $\mathcal{X} = \{x_1, \dots, x_n\}$ and $\mathcal{Y} = \{y_1, \dots, y_n\}$ of Boolean variables and a Boolean formula $\psi(\mathcal{X} \parallel \mathcal{Y})$ over $\mathcal{X} \cup \mathcal{Y}$.

Question: $\exists \mathbf{x}^* \in \{0, 1\}^n \forall \mathbf{y}' \in \{0, 1\}^n \psi(\mathbf{x}^* \parallel \mathbf{y}') = 1$?

Proof. (Theorem 7.1). Clearly CKT-UNIQUE-VALUE is contained in Σ_2^P , as for given strings $\mathbf{x}$ and $\mathbf{y}$ we can verify in polynomial time whether they obtain different values when given to the circuit. To show CKT-UNIQUE-VALUE is Σ_2^P -hard, we reduce from $\exists\forall$ -SAT. Throughout the proof, we interpret bit string as non-negative binary numbers ranging from 0 to $2^n - 1$, and we denote $\bar{y} = 2^n - 1$ (i.e., the all-ones sequence). We first provide a short proof sketch. Given a Boolean formula ψ over variables sets $\mathcal{X} = \{x_1, \dots, x_n\}$ and $\mathcal{Y} = \{y_1, \dots, y_n\}$, we construct a circuit $\mathcal{C}$ with $2n$ input bits, denoted $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$. We design $\mathcal{C}$ such that, if $\mathbf{x} \neq \mathbf{x}'$ then $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) \neq \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}')$ for all $\mathbf{y}$ and $\mathbf{y}'$. We do so by ensuring that different strings $\mathbf{x}$ are allocated disjoint ranges of values when evaluated by $\mathcal{C}$. Thus, when focusing on a specific string $\mathbf{x}'$ we need not worry about collisions with other $\mathbf{x}$ strings. Once separate ranges are guaranteed, we wish to ensure $\mathbf{x}'$ is a solution for ψ if and only if $\mathcal{C}(\mathbf{x}' \parallel \bar{y})$ is obtained uniquely. To do so, we partition all possible $\mathbf{y}$ values into pairs $(0, 1), (2, 3), \dots$, except for $\bar{y}$ and $\bar{y} - 1$, which remain unpaired (and are treated separately, as described in the formal proof). We then ensure that if $\mathbf{y}$ and $\mathbf{y}'$ are a pair, then $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{x} \parallel \mathbf{y}')$. More specifically, we ensure that if $\psi(\mathbf{x} \parallel \mathbf{y}) = \psi(\mathbf{x} \parallel \mathbf{y}') = 0$, then $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{x} \parallel \mathbf{y}') = \mathcal{C}(\mathbf{x} \parallel \bar{y})$, but if $\psi(\mathbf{x} \parallel \mathbf{y}) = \psi(\mathbf{x} \parallel \mathbf{y}') = 1$, then $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{x} \parallel \mathbf{y}') \neq \mathcal{C}(\mathbf{x} \parallel \bar{y})$. It then follows that $\mathcal{C}(\mathbf{x} \parallel \bar{y})$ is uniquely obtained if and only if for all $\mathbf{y}$ we have $\psi(\mathbf{x} \parallel \mathbf{y}) = 1$.

We proceed with the formal proof. Throughout the proof, we treat all strings as binary numbers. Thus, e.g., $\mathbf{x} \parallel 0$ would be the concatenation of the bit strings representing the numbers $\mathbf{x}$ and 0 (in particular, 0 is not a single bit here). For all $\mathbf{y} \in [2^n - 2]_0$, we define the following function:

$$p(\mathbf{y}) = \begin{cases} \mathbf{y} + 1 & \text{if } \mathbf{y} \text{ is even} \\ \mathbf{y} - 1 & \text{if } \mathbf{y} \text{ is odd} \end{cases}$$

Thus, for all $\mathbf{y} \in [2^{n-1} - 1]_0$, we have that $p(2\mathbf{y}) = 2\mathbf{y} + 1$ and $p(2\mathbf{y} + 1) = 2\mathbf{y}$, and we say $2\mathbf{y}$ and $2\mathbf{y} + 1$ are *paired*. Thus, all $\mathbf{y} \in [2^n - 2]_0$ have a pair. With ψ and $\bar{y}$ defined as above, we construct the following circuit $\mathcal{C}$ on $2n$ input bits. For $\mathbf{x}, \mathbf{y} \in [2^n]_0$, define:

$$\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \begin{cases} 2\mathbf{x} & \text{if } ((\mathbf{y} \leq \bar{y} - 2) \wedge (\psi(\mathbf{x} \parallel \mathbf{y}) = \psi(\mathbf{x} \parallel p(\mathbf{y})) = 1)) \text{ or} \\ & ((\mathbf{y} = \bar{y} - 1) \wedge (\psi(\mathbf{x} \parallel \mathbf{y}) = \psi(\mathbf{x} \parallel \bar{y}) = \psi(\mathbf{x} \parallel 0) = \psi(\mathbf{x} \parallel 1) = 1)) \\ 2\mathbf{x} + 1 & \text{otherwise} \end{cases}$$

We observe some properties of $\mathcal{C}$.

Observation 1. For all $\mathbf{x} \in [2^n]_0$, we have $\mathcal{C}(\mathbf{x} \parallel \bar{y}) = 2\mathbf{x} + 1$.

Observation 2. For all $\mathbf{x} \in [2^n]_0$ and $\mathbf{y} \in [2^n - 2]_0$, we have that $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \mathcal{C}(\mathbf{x} \parallel p(\mathbf{y}))$.

Observation 3. For $\mathbf{x}_1, \mathbf{x}_2 \in [2^n]_0$, if $\mathbf{x}_1 \neq \mathbf{x}_2$, then $\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y}_1) \neq \mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y}_2)$ for all $\mathbf{y}_1$ and $\mathbf{y}_2$.

Observation 4. For all $\mathbf{x} \in [2^n]_0$, we have that either $\mathcal{C}(\mathbf{x} \parallel \bar{y} - 1) = \mathcal{C}(\mathbf{x} \parallel \bar{y})$ or $\mathcal{C}(\mathbf{x} \parallel \bar{y} - 1) = \mathcal{C}(\mathbf{x} \parallel 0)$.

Assume there exists $\mathbf{x}'$ such that for all $\mathbf{y}$ we have $\psi(\mathbf{x}' \parallel \mathbf{y}) = 1$. Then for all $\mathbf{y} < \bar{y}$ we have that $\mathcal{C}(\mathbf{x}' \parallel \mathbf{y}) = 2\mathbf{x}'$, while we have $\mathcal{C}(\mathbf{x}' \parallel \bar{y}) = 2\mathbf{x}' + 1$ by Observation 1. By Observation 3, we have that $\mathcal{C}(\mathbf{x}' \parallel \bar{y})$ is uniquely obtained.

In the other direction, assume there exist $\mathbf{x}', \mathbf{y}' \in [2^n]_0$ such that $\mathcal{C}(\mathbf{x}' \parallel \mathbf{y}')$ is uniquely obtained. By Observations 2 and 4, we must have $\mathbf{y}' = \bar{\mathbf{y}}$. We wish to show that $\psi(\mathbf{x}' \parallel \mathbf{y}) = 1$ for all $\mathbf{y} \in [2^n]_0$. Assume towards contradiction that there exists $\mathbf{y}_0 \in [2^n]_0$ such that $\psi(\mathbf{x}' \parallel \mathbf{y}_0) = 0$. If $\mathbf{y}_0 < \bar{\mathbf{y}} - 1$, then we have $\mathcal{C}(\mathbf{x}' \parallel \mathbf{y}_0) = 2\mathbf{x}' + 1 = \mathcal{C}(\mathbf{x}' \parallel \bar{\mathbf{y}}) = \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}')$, a contradiction to the assumption that $\mathcal{C}(\mathbf{x}' \parallel \mathbf{y}')$ is obtained uniquely. Similarly, if $\mathbf{y}_0 \in \{\bar{\mathbf{y}} - 1, \bar{\mathbf{y}}\}$, then we have $\mathcal{C}(\mathbf{x}' \parallel \bar{\mathbf{y}} - 1) = \mathcal{C}(\mathbf{x}' \parallel \bar{\mathbf{y}})$, a contradiction. $\square$

Proof. (Theorem 7.2). Clearly 2-CKT-PARETO is contained in Σ_2^P , as for given strings $\mathbf{x}$ and $\mathbf{y}$ we can verify in polynomial time whether $\mathbf{x}$ obtains a higher output than $\mathbf{y}$ in one of the circuits. We reduce from CKT-UNIQUE-VALUE to show it is Σ_2^P -hard. Given an instance $\mathcal{C}$ of CKT-UNIQUE-VALUE, we construct the circuits $\mathcal{C}_1$ and $\mathcal{C}_2$, where for all $\mathbf{x} \in \{0, 1\}^n$ we set $\mathcal{C}_1(\mathbf{x}) = \mathcal{C}(\mathbf{x})$ and $\mathcal{C}_2(\mathbf{x}) = -\mathcal{C}(\mathbf{x})$. If there exists $\mathbf{x}^*$ such that for all $\mathbf{y} \neq \mathbf{x}^*$ we have $\mathcal{C}(\mathbf{x}^*) \neq \mathcal{C}(\mathbf{y})$, then for all $\mathbf{y} \neq \mathbf{x}^*$ we have that either $\mathcal{C}_1(\mathbf{x}^*) > \mathcal{C}_1(\mathbf{y})$ or $\mathcal{C}_2(\mathbf{x}^*) > \mathcal{C}_2(\mathbf{y})$. Therefore, $\langle \mathcal{C}_1, \mathcal{C}_2 \rangle$ is a Yes-instance of 2-CKT-PARETO. If there is no such string $\mathbf{x}^*$, then for all $\mathbf{x}$ there is some $\mathbf{y}'$ such that $\mathcal{C}(\mathbf{x}) = \mathcal{C}(\mathbf{y}')$, implying $\mathcal{C}_1(\mathbf{x}) = \mathcal{C}_1(\mathbf{y}')$ and $\mathcal{C}_2(\mathbf{x}) = \mathcal{C}_2(\mathbf{y}')$. Hence, $\langle \mathcal{C}_1, \mathcal{C}_2 \rangle$ is a No-instance of 2-CKT-PARETO. $\square$

F Proofs of Section 8

In this section, we provide all proofs missing from Section 7.

Proof. (Theorem 8.2). Showing that $\text{U}\exists\forall\text{-SAT} \leq_P \text{WDOM-STRATEGY}$ can be obtained via a trivial reduction: Given a Boolean formula ψ on variables $\mathcal{X} \cup \mathcal{Y}$, we construct a Boolean circuit $\mathcal{C}$ with $2n$ inputs and 1 output, defined for all $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$ by $\mathcal{C}(\mathbf{x} \parallel \mathbf{y}) = \psi(\mathbf{x} \parallel \mathbf{y})$. Let $\mathbf{x}^* \in \{0, 1\}^n$. It holds that

$$\forall \mathbf{y} \in \{0, 1\}^n \psi(\mathbf{x}^* \parallel \mathbf{y}) = 1 \wedge \forall \mathbf{x}' \neq \mathbf{x}^* \exists \mathbf{y}' \psi(\mathbf{x}' \parallel \mathbf{y}') = 0$$

if and only if

$$\forall \mathbf{x}', \mathbf{y} \in \{0, 1\}^n \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) \geq \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}) \wedge \forall \mathbf{x}' \neq \mathbf{x}^* \exists \mathbf{y}' \mathcal{C}(\mathbf{x}^* \parallel \mathbf{y}) > \mathcal{C}(\mathbf{x}' \parallel \mathbf{y}).$$

Thus, $\mathbf{x}^*$ is a unique solution of $\text{U}\exists\forall\text{-SAT}$ on ψ if and only if it is a unique solution of WDOM-STRATEGY on $\mathcal{C}$.

We now show $\text{WDOM-STRATEGY} \leq_P \text{U}\exists\forall\text{-SAT}$. In the proof, we treat the instance ψ of $\text{U}\exists\forall\text{-SAT}$ as a Boolean circuit rather than a formula, since we have from Cook's theorem [Coo71] that we can reduce polynomial-time Boolean circuits to polynomial-size Boolean formulae that preserve solutions. Given a circuit $\mathcal{C}$ for WDOM-STRATEGY with $2n$ inputs $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$ and n outputs, we construct a circuit ψ for $\text{U}\exists\forall\text{-SAT}$ with $4n$ inputs and 1 output. On inputs $\mathbf{x}', \mathbf{x}_1, \mathbf{x}_2, \mathbf{y} \in \{0, 1\}^n$ we define

$$\psi(\mathbf{x}' \parallel \mathbf{x}_1 \parallel \mathbf{x}_2 \parallel \mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{x}' = 0^n \wedge \mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y}) \geq \mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y}) \\ 0 & \text{otherwise.} \end{cases}$$

If there exists a solution $\mathbf{x}_1$ for WDOM-STRATEGY on $\mathcal{C}$, then $\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y}) \geq \mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y})$ for all $\mathbf{x}_2, \mathbf{y} \in \{0, 1\}^n$, and thus $0^n \parallel \mathbf{x}_1$ satisfies ψ for any choice of $\mathbf{x}_2, \mathbf{y}$. Further, since WDOM-STRATEGY is unambiguous, and since any $\mathbf{x}' \neq 0^n$ yields an output of 0 from ψ , we have that $0^n \parallel \mathbf{x}_1$ is the unique solution for $\text{U}\exists\forall\text{-SAT}$ on ψ .

In the opposite direction, assume there exists a solution $\mathbf{x}' \parallel \mathbf{x}_1$ for $\text{U}\exists\forall\text{-SAT}$ on ψ . We must have that $\mathbf{x}' = 0$ as otherwise ψ outputs 0 for any choice of $\mathbf{x}_2, \mathbf{y}$. Additionally, we have that $\mathcal{C}(\mathbf{x}_1 \parallel \mathbf{y}) \geq \mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y})$ for all $\mathbf{x}_2, \mathbf{y} \in \{0, 1\}^n$, by definition of ψ . Therefore, $\mathbf{x}_1$ satisfies $\mathcal{C}$ for any choice of $\mathbf{x}_2, \mathbf{y}$. Hence, it remains to show $\mathbf{x}_1$ is unique. Since $\mathbf{x}_1$ is a solution for $\text{U}\exists\forall\text{-SAT}$, it is by

definition the unique string satisfying $\forall \mathbf{y} \in \{0, 1\}^n \psi(\mathbf{x}_1, \mathbf{y}) = 1$. Thus, for any $\mathbf{x}'_1 \neq \mathbf{x}_1$, there exists $\mathbf{x}_2, \mathbf{y} \in \{0, 1\}^n$ such that $\psi(0^n \parallel \mathbf{x}'_1 \parallel \mathbf{x}_2 \parallel \mathbf{y}) = 0$, which implies $\mathcal{C}(\mathbf{x}'_1 \parallel \mathbf{y}) < \mathcal{C}(\mathbf{x}_2 \parallel \mathbf{y})$. This disqualifies $\mathbf{x}'_1$ from being another solution for WDOM-STRATEGY on $\mathcal{C}$, thereby establishing the uniqueness of $\mathbf{x}_1$. $\square$

Proof. (Theorem 8.3). There exists a solution for $\text{U}\exists\forall\text{-SAT}$ on formula ψ if and only if there exists $\mathbf{x}^*$ such that for all $\mathbf{y}$ we have $\psi(\mathbf{x}^* \parallel \mathbf{y}) = 1$, but there do not exist two such $\mathbf{x}^*$. Since the former condition is a Σ_2^P statement while the latter is a Π_2^P statement, we have that $\text{U}\exists\forall\text{-SAT} \in \mathbf{D}_2^P$. By Theorem 8.2, we have that $\text{WDOM-STRATEGY} \in \mathbf{D}_2^P$ as well. $\square$

To show $\text{U}\exists\forall\text{-SAT}$ and WDOM-STRATEGY are lower bounded by Π_2^P , we consider the following canonical Π_2^P SAT problem ([Sto76]).

Problem F.1:

$\forall\exists\text{-SAT}$

Input: Two sets $\mathcal{X} = \{x_1, \dots, x_n\}$ and $\mathcal{Y} = \{y_1, \dots, y_n\}$ of Boolean variables and a Boolean formula $\psi(\mathcal{X} \parallel \mathcal{Y})$ over $\mathcal{X} \cup \mathcal{Y}$.

Question: $\forall \mathbf{x} \in \{0, 1\}^n \exists \mathbf{y} \in \{0, 1\}^n \psi(\mathbf{x} \parallel \mathbf{y}) = 0$?

Proof. (Theorem 8.4). We reduce from $\forall\exists\text{-SAT}$ to $\text{U}\exists\forall\text{-SAT}$. Our reduction is analogous to the one showing that USAT is **coNP**-hard ([BG82]). Given a Boolean formula ψ over variables $\mathcal{X} = \{x_1, \dots, x_n\}, \mathcal{Y} = \{y_1, \dots, y_n\}$, we construct the formula ψ' on variables $\mathcal{X}' := \mathcal{X} \cup \{x_0\}$ and $\mathcal{Y}' := \mathcal{Y} \cup \{y_0\}$, where x_0 and y_0 are new variables. For all assignments $\mathbf{x}' = \mathbf{x}'_0 \parallel \dots \parallel \mathbf{x}'_n$, $\mathbf{y}' = \mathbf{y}'_0 \parallel \dots \parallel \mathbf{y}'_n$, we denote $\mathbf{x} =: \mathbf{x}'_1 \parallel \dots \parallel \mathbf{x}'_n$ and $\mathbf{y} =: \mathbf{y}'_1 \parallel \dots \parallel \mathbf{y}'_n$, and define:

$$\psi'(\mathbf{x}' \parallel \mathbf{y}') = (\mathbf{x}'_0 \wedge \dots \wedge \mathbf{x}'_n) \vee (\neg \mathbf{x}'_0 \wedge \psi(\mathbf{x} \parallel \mathbf{y}))$$

Notice that $\mathbf{y}'_0$ is ignored, and is only there to ensure $|\mathcal{X}'| = |\mathcal{Y}'|$. Clearly the all-ones assignment to $\mathcal{X}'$, denoted $\mathbf{x}^*$, satisfies ψ' regardless of the assignment to $\mathcal{Y}'$. Thus, we are interested in characterizing when it is the *unique* assignment satisfying ψ' for all assignments to $\mathcal{Y}'$.

Assume ψ is a Yes-instance of $\forall\exists\text{-SAT}$, and let $\bar{\mathbf{x}} \in \{0, 1\}^n$ and $\bar{\mathbf{x}}_0 \in \{0, 1\}$ such that $\bar{\mathbf{x}}_0 \parallel \bar{\mathbf{x}} \neq \mathbf{x}^*$. Then there exists $\bar{\mathbf{y}} \in \{0, 1\}^n$ such that $\psi(\bar{\mathbf{x}} \parallel \bar{\mathbf{y}}) = 0$. Hence, the term $(\neg \bar{\mathbf{x}}_0 \wedge \psi(\bar{\mathbf{x}} \parallel \bar{\mathbf{y}}))$ evaluates to 0 by definition of $\bar{\mathbf{y}}$. Furthermore, the term $(\bar{\mathbf{x}}_0 \wedge \dots \wedge \bar{\mathbf{x}}_n)$ evaluates to 0, as only the all-ones assignment satisfies it. Therefore, $\bar{\mathbf{x}}_0 \parallel \bar{\mathbf{x}}$ does not satisfy ψ' for all assignments to $\mathcal{Y}'$, implying $\mathbf{x}^*$ is the unique solution.

Conversely, assume ψ' is a Yes-instance of $\text{U}\exists\forall\text{-SAT}$. Then there is a unique $(n+1)$ -bit string satisfying ψ' for all assignments to $\mathcal{Y}'$, and by the above reasoning it must be the all-one assignment, $\mathbf{x}^*$. Let $\bar{\mathbf{x}} \in \{0, 1\}^n$. Assume towards contradiction that for all $\mathbf{y} \in \{0, 1\}^n$ we have $\psi(\bar{\mathbf{x}} \parallel \mathbf{y}) = 1$. Then, consider the string $0 \parallel \bar{\mathbf{x}} \neq \mathbf{x}^*$. By assumption, we have that $(\neg 0 \wedge \psi(\bar{\mathbf{x}} \parallel \mathbf{y}))$ evaluates to 1 for all $\mathbf{y} \in \{0, 1\}^n$, implying $0 \parallel \bar{\mathbf{x}}$ satisfies ψ' for all assignments to $\mathcal{Y}'$, a contradiction to the uniqueness of $\mathbf{x}^*$. Hence, there must exist $\bar{\mathbf{y}} \in \{0, 1\}^n$ such that $\psi(\bar{\mathbf{x}} \parallel \bar{\mathbf{y}}) = 0$, and thus ψ is a Yes-instance of $\forall\exists\text{-SAT}$. $\square$

G Strong Popularity in ASHG is PCW-complete

In this section we prove Theorem 4.5, which states that $\text{ASHG-STRONG-POPULARITY}$ is **PCW**-complete. We do so by reducing from CKT-CONDORCET . As the proof requires notation and terminology that do not concern other parts of the paper, we include a preliminaries section specific to this result. We then proceed to the reduction and proof of its correctness, as detailed in Sections G.2 to G.4.

G.1 Preliminaries

Let N be a set of agents. A *coalition* is a non-empty subset of N . A *singleton* coalition is a coalition of size one, and the *grand* coalition is simply N . Let $\mathcal{N}_i = \{S \subseteq N : i \in S\}$ denote the set of all coalitions to which agent i belongs. A *coalition structure*, or a *partition*, is a partition π of N into coalitions. For an agent $i \in N$, we denote by $\pi(i)$ the coalition i belongs to in π .

A *hedonic game* is a pair $\langle N, \succsim \rangle$, where $\succsim = (\succsim_i)_{i \in N}$ is a preference profile specifying for each agent i their preferences as a complete and transitive preference order $\succsim_i$ over $\mathcal{N}_i$. In hedonic games, agents' preferences are only affected by the members of their own coalition, and therefore $\succsim_i$ induces an underlying preference order over partitions as well, given by $\pi \succsim_i \pi'$ if and only if $\pi(i) \succsim_i \pi'(i)$. For coalitions $S, S' \in \mathcal{N}_i$, we say that agent i *weakly prefers* S over S' if $S \succsim_i S'$. Moreover, we say that i *prefers* S over S' if $S \succ_i S'$. We use the same terminology for preferences over partitions.

While generally agents' preferences may be arbitrary, in this paper we focus on *additively separable* hedonic games (ASHG), following [BJ02]. An ASHG is specified by a pair $\langle N, \mathbf{v} \rangle$ where N is a set of agents, and $\mathbf{v} = (\mathbf{v}_i : N \rightarrow \mathbb{R})_{i \in N}$ is a collection of *valuation functions*. The quantity $\mathbf{v}_i(j)$ denotes the value agent i assigns to agent j . We define the utility of agent i in coalition S by $u_i(S) = \sum_{j \in S \setminus \{i\}} \mathbf{v}_i(j)$, namely the sum of values she assigns to the other members of her coalition. We then define the preference orders $\succsim$ of the agents by $S \succ_i S'$ if and only if $u_i(S) > u_i(S')$, for any two coalitions S and S' . We extend the utilities for partitions by setting $u_i(\pi) = u_i(\pi(i))$, for any agent i and partition π .

Given two partitions π and π' and a coalition $S \subseteq N$, we define the *popularity margin* of S on the ordered pair (π, π') by

$$\phi_S(\pi, \pi') = |\{a \in S : u_a(\pi) > u_a(\pi')\}| - |\{a \in S : u_a(\pi') > u_a(\pi)\}|.$$

Note that in this definition, agents who are indifferent between the two partitions do not contribute to any of the two terms. For the grand coalition we write $\phi(\pi, \pi') = \phi_N(\pi, \pi')$, and for a singleton coalition containing only one agent p we write $\phi_p(\pi, \pi') = \phi_{\{p\}}(\pi, \pi')$. The definition of popularity margins is useful as it is often convenient to consider restricted subsets of agents, and then aggregate their margins (since, if S and S' are disjoint coalitions, then their popularity margins are additive, namely $\phi_S(\pi, \pi') + \phi_{S'}(\pi, \pi') = \phi_{S \cup S'}(\pi, \pi')$). We say partition π is *more popular* than π' if $\phi(\pi, \pi') > 0$. We say π is *strongly popular* if π is more popular than any partition $\pi' \neq \pi$.

Problem G.1:

ASHG-STRONG-POPULARITY

Input: Additively separable hedonic game $\langle N, \mathbf{v} \rangle$.

Question: Does $\langle N, \mathbf{v} \rangle$ admit a strongly popular partition? Namely, does the following hold:

$$\exists \text{partition } \pi^* \quad \forall \text{partition } \pi \text{ with } \pi \neq \pi^* \quad \phi(\pi^*, \pi) > 0$$

In the context of popularity it is useful to discuss Pareto optimality. We say that π' is a *Pareto improvement* from π if all agents weakly prefer π' over π , and at least one agent strictly prefers π' over π . If there exists no Pareto improvement from π , we say π is *Pareto-optimal*. Clearly, strongly popular partitions are Pareto-optimal. Indeed, every Pareto improvement is a more popular partition. By contrast, Pareto-optimal partitions need not be strongly popular.

We briefly introduce some notation concerning Boolean circuits. Let $\mathcal{C}$ be a Boolean circuit with n input bits, let $\mathcal{G}$ be a gate in $\mathcal{C}$, and let $\mathbf{x} = (\mathbf{x}_i)_{i \in [n]} \in \{0, 1\}^n$. We denote by $\mathbf{x}(\mathcal{G})$ the value (0 or 1) that the gate $\mathcal{G}$ outputs when the circuit $\mathcal{C}$ is given the string $\mathbf{x}$ as input. Moreover, we denote by $|\mathcal{C}|$ the number of gates in $\mathcal{C}$. A solution for CKT-CONDORCET is denoted a *Condorcet string*.

Lastly, we introduce a useful definition to help us construct an ASHG for the reduction.

Definition G.1. Consider an ASHG $\langle N, \mathbf{v} \rangle$, and let $p, p' \in N$ with $p \neq p'$. We say that p' is a *one-way replica* of p (or simply a *replica*), and we say that p is the *origin agent* of p' , if the following statements hold:

- For every agent $b \notin \{p, p'\}$, we have $\mathbf{v}_p(b) = \mathbf{v}_{p'}(b)$.
- We have $\mathbf{v}_p(p') = \mathbf{v}_{p'}(p) = 10$.
- For every agent $b \notin \{p, p'\}$ with $\mathbf{v}_b(p) \geq 0$, we have $\mathbf{v}_b(p') = 0$.
- For every agent $b \notin \{p, p'\}$ with $\mathbf{v}_b(p) < 0$, we have $\mathbf{v}_b(p') = \mathbf{v}_b(p)$.

Notice that in this definition p and p' , have the same valuation towards other agents, and, intuitively, they strongly wish to belong to the same coalition. The mutual value of 10 is specifically tailored to our reduction and designed to be greater than the sum of all other positive values the replicated agents in the constructed game assign. Furthermore, any coalition whose members like the presence of p (i.e. have non-negative valuation to p) will not object to including p' as well, though her inclusion will not affect the utilities of other agents in this coalition (apart from p and p'). Thus, one may interpret a replica as a “weight multiplier”, i.e., a tool that enables us to add more weight to the opinion of agent p . Lastly, we observe that if an agent has more than one replica, then, by definition, all its replicas must assign a value of 10 to each other as well.

G.2 Setup of the Reduction

Suppose that we are given a CKT-CONDORCET instance $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$, where each circuit $\mathcal{C}_j$ is composed of gates $\mathcal{G}_1^j, \dots, \mathcal{G}_{|\mathcal{C}_j|}^j$ (recall that $|\mathcal{C}_j|$ denotes the number of gates in $\mathcal{C}_j$). Since we usually focus on an individual circuit in our analysis, we often omit the superscripts of the gates, writing, e.g., $\mathcal{G}_i$ for $\mathcal{G}_i^j$. We assume without loss of generality that the outputs of all circuit are interpreted as non-negative binary numbers, as otherwise we can add a sufficiently large constant to all outputs. Furthermore, we assume that the gates are topologically ordered, so that, in a specific circuit, if $\mathcal{G}_i$ is an input to $\mathcal{G}_j$ then $i < j$. Moreover, we assume that the first n gates of each circuit $\mathcal{G}_1, \dots, \mathcal{G}_n$ simply copy the inputs $\mathbf{x}_1, \dots, \mathbf{x}_n$ respectively, and these are the only gates where $\mathbf{x}_1, \dots, \mathbf{x}_n$ appear; we call those gates *Copy-gates*. Apart from those, we assume circuits are only composed of And- and Not-gates. The output gates of each circuit are simply some subset of size n of its gates, but we will denote them also by $\mathcal{O}_1, \dots, \mathcal{O}_n$, in reverse order of significance (where $\mathcal{O}_n$ is the most significant bit). We construct an ASHG $\langle N, \mathbf{v} \rangle$. We let $N = X \cup A \cup G \cup L \cup W \cup Z \cup V$, where those disjoint sets induce different *types* of agents denoted *T-agents* for $T \in \{X, A, G, L, W, Z, V\}$, as detailed below. For L -agents, we make a further distinction between two sub-types, so that $L = L^a \cup L^g$. Note that all one-way replicas in the construction are assumed to belong to the same type as their origin agent.

- We create X -agents $x_1, \dots, x_n$, representing the input bits $\mathbf{x}_1, \dots, \mathbf{x}_n$ respectively. Furthermore, we create additional X -agents (not corresponding to any input bit) to increase the overall number of X -agents to $1 + 9mn + 3n + \sum_{i=1}^m (10|\mathcal{C}_i| + 1)$.
- For each input bit $\mathbf{x}_i$ we create two *assignment agents* (or A -agents) a_i^1 and a_i^0 , and an *assignment-alternative agent* (or L^a -agent) ℓ_i^a . Furthermore, for each of those three agents we create m one-way replicas. We denote by T_i^a the set that includes a_i^1 and all its replicas, by F_i^a the set that includes a_i^0 and all its replicas, and by L_i^a the set that includes ℓ_i^a and all its replicas. Agents a_i^1 and a_i^0 , i.e., A -agents corresponding to the same input bit $\mathbf{x}_i$, are called

complementary agents of one another. We will use their coalitions to derive an assignment to the input string, and vice versa.

- For each gate $\mathcal{G}_i$ (Copy-, Not-, or And-gate), we create two *gate agents* (or *G-agents*) g_i^1 and g_i^0 , and a *gate-alternative agent* (or *L^g-agent*) ℓ_i^g . Furthermore, we create one-way replicas $\hat{g}_i^1$, $\hat{g}_i^0$, and $\hat{\ell}_i^g$, of each of those respective agents. We denote $T_i^g = \{g_i^1, \hat{g}_i^1\}$, $F_i^g = \{g_i^0, \hat{g}_i^0\}$, and $L_i^g = \{\ell_i^g, \hat{\ell}_i^g\}$. Agents g_i^1 and g_i^0 corresponding to the same gate $\mathcal{G}_i$ are called *complementary* agents of one another. We will use their coalitions to derive the outputs of the gates, and vice versa.
- For each And-gate $\mathcal{G}_i$ we additionally construct a *W-agent* w_i and a *Z-agent* z_i . Furthermore, we construct one-way replicas $\hat{w}_i$ and $\hat{z}_i$ for each of them respectively. We denote $W_i = \{w_i, \hat{w}_i\}$ and $Z_i = \{z_i, \hat{z}_i\}$.
- For each circuit $\mathcal{C}_j$, we create a *voter agent* (or *V-agent*), denoted v_j . When considering a specific circuit we may omit the subscript of this agent. v should not be confused with the valuation function $\mathbf{v}$.

For each input bit $\mathbf{x}_i$, the set $T_i^a \cup F_i^a \cup L_i^a$ is called the *assignment gadget*, or *A-gadget*, of $\mathbf{x}_i$, denoted AG_i . For each gate $\mathcal{G}_i$, the set $T_i^g \cup F_i^g \cup L_i^g$ (or, if $\mathcal{G}_i$ is an And-gate, then $T_i^g \cup F_i^g \cup L_i^g \cup W_i \cup Z_i$) is called the *gate gadget* of $\mathcal{G}_i$, denoted GG_i . We sometimes refer to them as Copy-, Not-, or And-gadgets when referring to a specific type of gate. The *X*-, *A*-, *V*-, and *G*-agents are referred to as *core agents*.

Given a circuit $\mathcal{C}_j$, we denote by C_j its corresponding set of agents (which includes the gadgets of all gates of $\mathcal{C}_j$, and its voter agent). The number of non-*X*-agents is at most $9mn + 3n + \sum_{i=1}^m (10|\mathcal{C}_i| + 1)$, where we bound with the case that all gates are Copy- and And-gates. Hence, we have that $|X| > \frac{|N|}{2}$, which will be helpful during the proof.

We now describe the valuations $\mathbf{v} = (\mathbf{v}_i: N \rightarrow \mathbb{R})_{i \in N}$ of the agents. Any value between two agents not described below implies a valuation of 0 between those agents. We use $-\infty$ to indicate a large negative number, satisfying the property that an agent who assigns $-\infty$ to an agent in her coalition will have a negative utility regardless of the composition of the rest of her coalition. For instance, we can regard ∞ as denoting $|N|$ times the largest (positive) valuation in the game. In the following description we do not include the valuations of all one-way replicas, since those can be derived from their definition, using the valuations of their origin agents.

- Each *X-agent* x assigns 1 to any other *X-agent*.
- For each input bit $\mathbf{x}_i$:
 - agents a_i^1 and a_i^0 assign value $-\infty$ to each other, and value 1 to x_i and to ℓ_i^a ;
 - agent ℓ_i^a assigns value 1 to agents a_i^1 and a_i^0 .
- For every gate $\mathcal{G}_i$:
 - agents g_i^1 and g_i^0 assign value $-\infty$ to each other;
 - agent ℓ_i^g assigns value 1 to agents g_i^1 and g_i^0 .
- For every Copy-gate $\mathcal{G}_i = \mathbf{x}_i$:
 - agent g_i^1 assigns value 1 to agents a_i^1 and ℓ_i^g , and $-\infty$ to a_i^0 ;
 - agent g_i^0 assigns value 1 to agents a_i^0 and ℓ_i^g , and $-\infty$ to a_i^1 ;

- agent a_i^1 assigns $-\infty$ to g_i^0 .
- agent a_i^0 assigns $-\infty$ to g_i^1 .
- For every Not-gate $\mathcal{G}_i = \neg \mathcal{G}_j$:
 - agent g_i^1 assigns value 1 to agents g_j^0 and ℓ_i^g , and $-\infty$ to g_j^1 ;
 - agent g_i^0 assigns value 1 to agents g_j^1 and ℓ_i^g , and $-\infty$ to g_j^1 ;
 - agent g_j^1 assigns $-\infty$ to any g_i^1 ;
 - agent g_j^0 assigns $-\infty$ to any g_i^0 .
- For every And-gate $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$:
 - agent g_i^1 assigns 1 to g_j^1 and to g_k^1 , 2 to ℓ_i^g , and $-\infty$ to g_j^0 , g_k^0 , w_i , and z_i .
 - agent g_i^0 assigns 1 to g_j^0 and to w_i , 2 to g_k^0 and to z_i , and 3 to ℓ_i^g ;
 - agent w_i assigns $-\infty$ to g_i^1 , g_k^1 , g_j^0 , and z_i (in particular, w_i assigns 0 to g_k^0 and g_j^1);
 - agent z_i assigns $-\infty$ to g_i^1 , g_j^1 , g_k^0 , and w_i (in particular, z_i assigns 0 to g_j^0 and g_k^1);
 - agent g_j^1 assigns $-\infty$ to z_i ;
 - agent g_k^1 assigns $-\infty$ to w_i ;
 - agent g_j^0 assigns $-\infty$ to g_i^1 , and w_i ;
 - agent g_k^0 assigns $-\infty$ to g_i^1 , and z_i ;
- Any L -agent assigns $-\infty$ to all agents and vice versa, apart from the cases where other values were specified above.
- Every voter agent v assigns value 2^{n+1} to any other voter agent, and to x_1 . Furthermore, if the outputs of the circuit corresponding to v are $\mathcal{O}_1, \dots, \mathcal{O}_n$ (where $\mathcal{O}_n$ is the most significant bit), then for every $\mathcal{O}_j$, $j \in [n]$, with corresponding G -agents o_j^1 and o_j^0 , agent v assigns value 2^j to o_j^1 .

Note that the A -agents are not associated with any specific circuit, but can be interpreted as a barrier between the inputs and the circuits. This helps guarantee that circuits follow the same assignment to the input string. The role of the alternative agents is to provide an alternative coalition for each gate agent or assignment agent representing the negation of the assignment of that bit or gate (whereas, as we will see, agents who correspond to their gate's/bit's truth assignment will all be grouped in a single main coalition). Notice that the value that a V -agent assigns to a corresponding output gate-agent o_j^1 dominates the sum of all values she assigns to agents associated with less significant bits of the output. This will be crucial to establish that the strong popularity property of an input string is preserved in the corresponding partition in the constructed ASHG, and vice versa. In the following two sections, we show that a Condorcet string exists in $\mathfrak{C}$ if and only if a strongly popular partition exists in the constructed ASHG. Illustrations of the gadgets and overview of the reduction can be seen in Figures 6 and 7 respectively.

Definition G.2. Let $\mathbf{x}_i$ be an input bit with corresponding A -gadget AG_i , let π' be a partition of N , and let $MC := \pi'(x_1)$. If $a_i^1 \in MC$ and $a_i^0 \in \pi'(\ell_i^a)$ then we say that AG_i π' -corresponds to assignment $\mathbf{x}_i = 1$, and if $a_i^0 \in MC$ and $a_i^1 \in \pi'(\ell_i^a)$ then we say that AG_i π' -corresponds to assignment $\mathbf{x}_i = 0$. If AG_i π' -corresponds to either 0 or 1, we say it is a π' -valid gadget.

Let $\mathcal{G}_i$ be a gate. If $g_i^1 \in MC$ and $g_i^0 \in \pi'(\ell_i^g)$, we say that GG_i π' -corresponds to value $\mathcal{G}_i = 1$, and if $g_i^0 \in MC$ and $g_i^1 \in \pi'(\ell_i^g)$, we say that GG_i π' -corresponds to value $\mathcal{G}_i = 0$. If GG_i π' -corresponds to either 0 or 1, we say it is a π' -valid gadget.

Definition G.3. Let $\mathcal{G}_i$ be a gate with corresponding G -gadget GG_i , and let π' be a partition of N . Suppose the inputs (or input) of $\mathcal{G}_i$ have π' -valid corresponding gadgets. We say that GG_i π' -complies with its inputs if GG_i π' -corresponds to the value $\mathcal{G}_i$ obtains when evaluated on the values its input gadgets π' -correspond to (e.g., if $\mathcal{G}_i = \neg \mathcal{G}_j$ and GG_j π' -corresponds to $\mathcal{G}_j = 1$, then GG_i π' -complies with its input if it π' -corresponds to $\mathcal{G}_i = 0$).

Furthermore, given a string $\mathbf{x} \in \{0, 1\}^n$ we say that GG_i π' -complies with $\mathbf{x}$ if GG_i π' -corresponds to the value $\mathcal{G}_i$ obtains when its circuit is evaluated on $\mathbf{x}$.

In both Theorems G.2 and G.3, when it is clear what partition is under consideration we often omit its name from the statement (e.g., we may write *valid* instead of π' -valid).

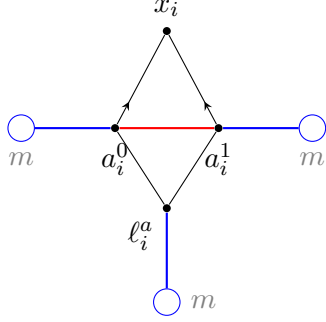


Figure 2: Assignment gadget of bit x_i . Dots represent single agents; circles represent sets of m replicas.

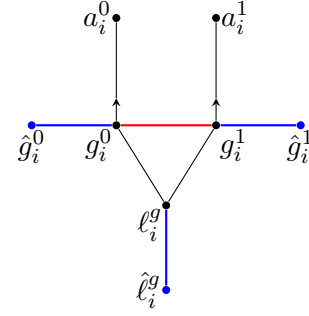


Figure 3: Copy-gadget of gate $\mathcal{G}_i = x_i$.

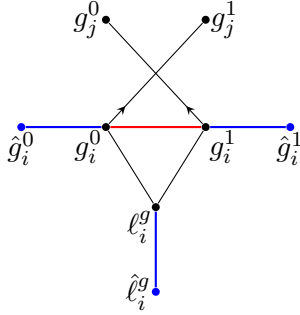


Figure 4: Not-gadget of gate $\mathcal{G}_i = \neg \mathcal{G}_j$.

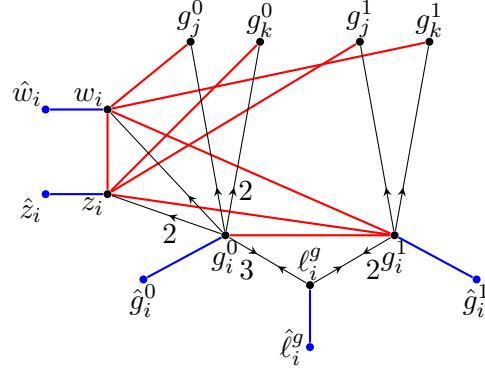


Figure 5: And-gadget of gate $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$.

Figure 6: The main gadgets of the proof of Theorem 4.5. One-way replica agents are represented as blue dots with a blue edge to their origin. Valuations concerning them can be derived from their origin. Omitted edges indicate a valuation of 0. Red edges indicate mutual valuation of $-\infty$. Unlabeled black directed edges indicate valuation of 1 in the shown direction, and unlabeled, black, undirected edge implies mutual valuation of 1. Other valuations are written explicitly. If an edge appears only in one direction, the valuation in the opposite direction is 0.

G.3 Condorcet string implies Strongly Popular Partition

Suppose that we have a Condorcet string $\mathbf{x}^* = (x_1^*, \dots, x_n^*)$ for $\mathfrak{C} = \langle \mathcal{C}_1, \dots, \mathcal{C}_m \rangle$. Let π^* be the following partition:

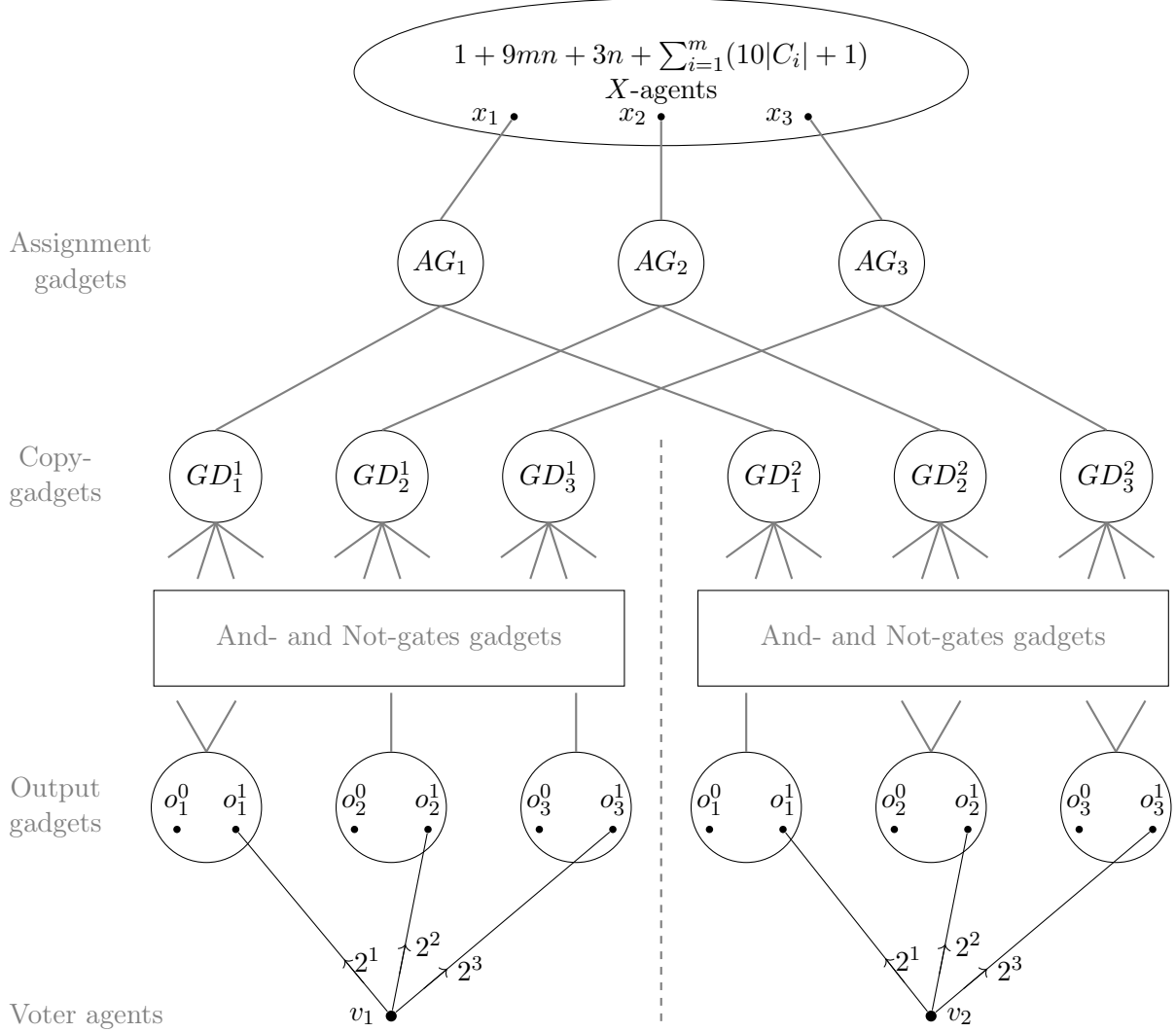


Figure 7: High-level illustration of the reduction used in the proof of Theorem 4.5, for $n = 3$. Gray edges indicate interactions between gadgets (rather than individual edges within the ASHG). The vertical dashed line separates C_1 and C_2 . Only the core agents of the output gates are shown, illustrating their connections to the voters. Black edges represent the voters' valuations, while omitted (v_i, o_j^0) edges correspond to valuations of 0. For clarity, some output gates are arbitrarily depicted as And-gates and others as Not-gates, as can be inferred from the number of incoming gates.

- All X -agents are in a coalition denoted MC^* (for denoting a *main coalition* to which we will add further agents).
- For every input bit x_i^* , if $x_i^* = 1$ then $T_i^a \subseteq MC^*$ and $F_i^a \cup L_i^a \in \pi^*$, and if $x_i^* = 0$ then $F_i^a \subseteq MC^*$ and $T_i^a \cup L_i^a \in \pi^*$.
- For every gate $\mathcal{G}_i$, if $x^*(\mathcal{G}_i) = 1$ then $T_i^g \subseteq MC^*$ and $F_i^g \cup L_i^g \in \pi^*$, and if $x^*(\mathcal{G}_i) = 0$ then $F_i^g \subseteq MC^*$ and $T_i^g \cup L_i^g \in \pi^*$.
- For every And-gate $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$:

- if $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 1$ or $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 0$, then $W_i \in \pi^*$, and $Z_i \in \pi^*$.
- if $\mathbf{x}^*(\mathcal{G}_j) = 0$ and $\mathbf{x}^*(\mathcal{G}_k) = 1$ then $W_i \in \pi^*$, and $Z_i \subseteq MC^*$.
- if $\mathbf{x}^*(\mathcal{G}_j) = 1$ and $\mathbf{x}^*(\mathcal{G}_k) = 0$ then $Z_i \in \pi^*$, and $W_i \subseteq MC^*$.

- All V -agents are in MC^* .

We want to show that π^* is a strongly popular partition. To that end, assume towards contradiction there exists a partition $\pi \neq \pi^*$ such that $\phi(\pi^*, \pi) \leq 0$. We may assume without loss of generality that π is Pareto-optimal⁴ (note that, however, π need not be strongly popular). We denote $MC = \pi(x_1)$ (we will see that MC will be the “main” coalition in π , similarly to MC^* in π^*). In the following lemmas, we establish some properties of π , and analyze the popularity margins among several groups of agents.

Lemma G.4. *Let $\ell, \ell' \in L$, corresponding to different gadgets (they may also be replicas). Then $\ell' \notin \pi(\ell)$.*

Proof. Assume for contradiction that $\ell' \in \pi(\ell)$. Notice that, since they belong to different gadgets, every agent assigns $-\infty$ to at least one of ℓ and ℓ' , and therefore every agent in $\pi(\ell)$ obtains negative utility. Hence, it is a Pareto improvement to dissolve this coalition into singletons, a contradiction to the Pareto optimality of π . $\square$

Lemma G.5. *Let $x \in X$. We have that $\phi_x(\pi^*, \pi) \geq 0$.*

Proof. Agent x only assigns a positive value to other X -agents, and only assigns a negative value to L -agents. Recall that by definition of π^* we have that $X \subseteq MC^*$ and $L \cap MC^* = \emptyset$. Thus, x obtains the her maximum possible utility in π^* . Therefore we have $\phi_x(\pi^*, \pi) \geq 0$. $\square$

Lemma G.6. *It holds that $X \subseteq MC$.*

Proof. Assume for contradiction that there exists some $x \in X \setminus \{x_1\}$ such that $x \notin \pi(x_1) = MC$. Then clearly all X -agents prefer π^* over π , and so $\phi_X(\pi^*, \pi) = |X|$. By design, $|X| > \frac{|N|}{2}$, and therefore π^* is more popular than π , a contradiction. $\square$

Lemma G.7. *It holds that $L \cap MC = \emptyset$.*

Proof. Assume otherwise. Then by Theorem G.6 we have that all X -agents obtain a negative utility in π , and thus prefer π^* . By design, $|X|$ is more than half the total number of agents, and therefore π^* is more popular than π , a contradiction. $\square$

Lemma G.8. *It holds that $V \subseteq MC$.*

Proof. Let us fix some voter agent $v \in V$ and assume towards contradiction that $v \notin MC$. Recall that voter agents assign a valuation of 2^{n+1} to other voter agents and to x_1 , and positive valuations of $2^1, 2^2, \dots, 2^n$ to agents corresponding to n output gates in their circuit. All their other valuations are non-positive. Hence, for every voter agent $v' \in V$, we have $u_{v'}(\pi) < m \cdot 2^{n+1}$ (since either $v' \notin \pi(v)$ or $v' \notin MC = \pi(x_1)$, and so even the sum of all other positive values v' assigns is less than $m \cdot 2^{n+1}$). Hence, it is a Pareto improvement to extract all V -agents from their coalitions and add them to MC : All V -agents will prefer this since after this change MC contains no L -agents (Theorem G.7), all X -agents, and all V -agents, and thus they would obtain a utility of at least

⁴This is a well-known argument in the context of popularity ([BB22, BG25]). Briefly, if π' is a Pareto improvement from π , then π' is also more popular than π^* .

$m \cdot 2^{n+1}$. Furthermore observe that agents in $N \setminus V$ never assign a positive value to V -agents, and agents in $N \setminus L$ (recall that L -agents are not present in MC by Theorem G.7), never assign a negative value to V -agents. Therefore, no agent will be worse off by applying this change. Hence, we have a contradiction to Pareto-optimality of π . $\square$

Lemma G.9. *Let $\mathbf{x}_i$ be an input bit, and denote the corresponding assignment gadget by $AG_i = F_i^a \cup T_i^a \cup L_i^a$. Then we have that $\phi_{AG_i}(\pi^*, \pi) \geq 0$.*

Proof. Assume for contradiction that $\phi_{AG_i}(\pi^*, \pi) < 0$. Note that, by Theorems G.6 and G.7, we have that $\ell_i^a \notin \pi(x_i)$, and therefore agents in T_i^a and F_i^a cannot prefer π over π^* . Hence, some agent $\ell \in L_i^a$ must prefer π , and so we must have $\{a_i^1, a_i^0\} \subseteq \pi(\ell)$. This implies that all agents in $T_i^a \cup F_i^a$ prefer π^* , a contradiction to $\phi_{AG_i}(\pi^*, \pi) < 0$. $\square$

Lemma G.10. *Let GG_i be an And-gate. Then no agent $W_i \cup Z_i$ prefers π over π^* .*

Proof. This is immediate because such agents obtain maximum utility in π^* . $\square$

Lemma G.11. *Let $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$ be an And-gate. If $F_i^g \cap \pi(\ell_i^g) = \emptyset$ or $T_i^g \cap \pi(\ell_i^g) \neq \emptyset$, then $\phi_{F_i^g \cup W_i \cup Z_i}(\pi^*, \pi) \geq 0$.*

Proof. In this lemma, we consider the popularity margin of agents in $F_i^g \cup W_i \cup Z_i$. Hence, if all agents in F_i^g do not prefer π over π^* , then we are done by Theorem G.10. Assume, therefore, that some agent $p \in F_i^g$ prefers π . Then we must have $\ell_i^g \notin \pi(p)$ (since if $\ell_i^g \in \pi(p)$ then by the statement of this lemma we have $T_i^g \cap \pi(\ell_i^g) \neq \emptyset$, and so p prefers π^* , a contradiction). Furthermore, one may verify that we must have either $\{w_i, g_i^0\} \subseteq \pi(p)$ or $\{z_i, g_i^0\} \subseteq \pi(p)$, and thus either all agents in W_i or all agents in Z_i prefer π^* . Hence, by Theorem G.10, we have $\phi_{F_i^g \cup W_i \cup Z_i}(\pi^*, \pi) \geq 0$. $\square$

Lemma G.12. *Let $\mathcal{G}_i$ be a gate with corresponding gadget GG_i . Then we have $\phi_{GG_i}(\pi^*, \pi) \geq 0$.*

Proof. Assume for contradiction that $\phi_{GG_i}(\pi^*, \pi) < 0$. We make a case distinction based on the type of gate.

Case 1: Suppose $\mathcal{G}_i = \mathbf{x}_i$ is a Copy-gate. If some $p \in T_i^g$ prefers π over π^* then $\{a_i^1, \ell_i^g\} \subseteq \pi(p)$. As agents in L_i^g and a_i^1 have a mutual valuation of $-\infty$, this implies that all agents in L_i^g prefer π^* over π . Moreover, all agents in F_i^g obtain at most the utility they obtain in π^* . Thus, $\phi_{GG_i}(\pi^*, \pi) \geq 0$, a contradiction. If some $p \in F_i^g$ prefers π over π^* we reach a similar contradiction. Hence, some $\ell \in L_i^g$ must prefer π , and so we must have $\{g_i^1, g_i^0\} \subseteq \pi(\ell)$. This implies that all agents in $T_i^g \cup F_i^g$ prefer π^* , which contradicts $\phi_{GG_i}(\pi^*, \pi) < 0$.

Case 2: If $\mathcal{G}_i = \neg \mathcal{G}_j$ is a Not-gate, the proof is analogous to Case 1.

Case 3: Suppose $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$ is an And-gate. If some $\ell \in L_i^g$ prefers π over π^* then $\{g_i^1, g_i^0\} \subseteq \pi(\ell)$, implying all agents in $T_i^g \cup F_i^g$ prefer π^* . Hence, by Theorem G.10, we reach a contradiction to $\phi_{GG_i}(\pi^*, \pi) < 0$. If some $p \in T_i^g$ prefers π over π^* then $\ell_i^g \in \pi(p)$ and also $|\{g_j^1, g_k^1\} \cap \pi(p)| \geq 1$. This implies that any $\ell \in L_i^g$ prefers π^* . Moreover, since $\ell_i^g \in \pi(p)$ we may apply Theorem G.11, and we reach a contradiction. So agents in $L_i^g \cup T_i^g$ cannot prefer π , and thus we are left only with the possibility that some $p \in F_i^g$ prefers π . Hence, all of F_i^g are in the same coalition. Furthermore, if $F_i^g \cap \pi(\ell_i^g) = \emptyset$ then we are done by Theorem G.11. Otherwise, we have $\ell_i^g \in \pi(p)$ (since all of F_i^g are in the same coalition). Moreover, since p prefers π , there must be another agent in $\pi(p)$ who p assigns positive value to. This implies that all agents in L_i^g prefer π^* , and, by Theorem G.10, we again conclude that $\phi_{GG_i}(\pi^*, \pi) \geq 0$, a contradiction. $\square$

Lemma G.13. *Let $\mathcal{C}_j$ be a circuit. Then $\phi_{\mathcal{C}_j}(\pi^*, \pi) \geq -1$.*

Proof. Since C_j is composed only of the gadgets of the gates of $\mathcal{C}_j$, and its voter agent, the result follows from Theorem G.12. $\square$

Lemma G.14. *Let $\mathcal{G}_i$ be a gate with corresponding G -gadget GG_i . Suppose the inputs (or input) of $\mathcal{G}_i$ have π -valid corresponding gadgets. Then if GG_i does not π -comply with its inputs, we have that $\phi_{GG_i}(\pi^*, \pi) \geq 2$.*

Proof. First, notice that any agent $\ell \in L_i^g$ can only prefer π if $\{g_i^1, g_i^0\} \subseteq \pi(\ell_i^g)$, but then all agents in $T_i^g \cup F_i^g$ prefer π and thus the claim holds (if GG_i is an And-gadget we also utilize Theorem G.10). So we may assume agents in L_i^g do not prefer π . We now make a case distinction based on the gadget type of GD_i .

Case 1: Suppose GG_i is a Copy-gadget, with input A -gadget AG_i . Since AG_i is π -valid, assume without loss of generality that $a_i^1 \in MC$ and $a_i^0 \in \pi(\ell_i^a)$. Then, by assumption that GG_i does not π -comply with AG_i , we have that either $g_i^0 \notin \pi(\ell_i^g)$ or $g_i^1 \notin MC$. If $g_i^0 \notin \pi(\ell_i^g)$, it is easy to verify that all agents in F_i^g prefer π^* . Furthermore, the utility of all agents $p \in T_i^g$ is at most what they obtained in π^* (since $a_i^1 \in MC$, and $\ell_i^g \notin MC$ by Theorem G.7), and thus they cannot prefer π . Hence, the claim holds. If $g_i^1 \notin MC$, then the utility of all $p \in T_i^g$ is at most what they obtained in π^* , so none of them prefers π . Furthermore, if $g_i^1 \in \pi(\ell_i^g)$, then all agents in F_i^g prefer π^* . Otherwise, all agents in T_i^g prefer π^* (while agents in F_i^g cannot prefer π , as established above). Hence, the claim holds.

Case 2: If GG_i is a Not-gadget, the proof is analogous to Case 1.

Case 3: Suppose GG_i is an And-gadget with π -valid input G -gadgets GG_j and GG_k , and that GG_i does not π -comply with its inputs. Recall that agents in L_i^g do not prefer π . We consider the following three sub-cases corresponding to the possible values of the input gates.

Case 3.1 Suppose GG_j and GG_k π -correspond to $\mathcal{G}_j = 1$ and $\mathcal{G}_k = 1$. Then, by Theorem G.7, we have that no agent in T_i^g prefers π . Moreover, since GG_i does not π -comply with its input, we have that either $g_i^0 \notin \pi(\ell_i^g)$, or $g_i^1 \notin MC$. If $g_i^0 \notin \pi(\ell_i^g)$, then either all agents in F_i^g prefer π^* (and we are done by Theorem G.10), or there is some $p \in F_i^g$ who is indifferent (p cannot prefer π since $g_j^0 \in \pi(\ell_j^g)$ and $g_k^0 \in \pi(\ell_k^g)$). But if p is indifferent, then $\{w_i, z_i\} \subseteq \pi(p)$, and thus all agents in $W_i \cup Z_i$ prefer π^* , and we are again done.

So we must have $g_i^0 \in \pi(\ell_i^g)$, and be in the case where $g_i^1 \notin MC$. But then all agents in T_i^g prefer π^* . Furthermore, if some agent in F_i^g prefers π , we must have that all agents in W_i or all agents in Z_i prefer π^* , and by Theorem G.10 we are done. Otherwise, we are again done using Theorem G.10.

Case 3.2: Suppose GG_j and GG_k correspond to $\mathcal{G}_j = 0$ and $\mathcal{G}_k = 0$. Then $g_j^1 \in \pi(\ell_j^g)$ and $g_k^1 \in \pi(\ell_k^g)$, and thus no agent in T_i^g prefers π . Since GG_i does not comply with its input, we have that either $g_i^0 \notin MC$ or $g_i^1 \notin \pi(\ell_i^g)$. First, suppose $g_i^0 \notin MC$. If $\ell_i^g \notin \pi(g_i^0)$ then $\phi_{F_i^g}(\pi^*, \pi) \geq 0$. Moreover, either $\{w_i, z_i\} \subseteq \pi(g_i^0)$ —and thus $\phi_{W_i \cup Z_i}(\pi^*, \pi) = 4$ and we are done—or $\{w_i, z_i\} \not\subseteq \pi(g_i^0)$ —and thus $\phi_{F_i^g}(\pi^*, \pi) = 2$, and, by Theorem G.10, we are done.

If $\ell_i^g \in \pi(g_i^0)$, then $\phi_{T_i^g}(\pi^*, \pi) = 2$. Furthermore, if $w_i \in \pi(g_i^0)$ then $\phi_{W_i}(\pi^*, \pi) = 2$, if $z_i \in \pi(g_i^0)$ then $\phi_{Z_i}(\pi^*, \pi) = 2$, and if $\{w_i, z_i\} \not\subseteq \pi(g_i^0)$ then $\phi_{F_i^g}(\pi^*, \pi) \geq 0$ (and by Theorem G.10, we have $\phi_{W_i \cup Z_i}(\pi^*, \pi) \geq 0$ as well). Hence, we have that $\phi_{F_i^g \cup W_i \cup Z_i}(\pi^*, \pi) \geq 0$, and we are done.

So we must have $g_i^0 \in MC$, and be in the case where $g_i^1 \notin \pi(\ell_i^g)$. Thus, we have $\phi_{T_i^g}(\pi^*, \pi) = 2$. Moreover, if $w_i \in MC$ then $\phi_{W_i}(\pi^*, \pi) = 2$, if $z_i \in MC$ then $\phi_{Z_i}(\pi^*, \pi) = 2$, and if $\{w_i, z_i\} \not\subseteq MC$ then, since $\{g_j^0, g_k^0\} \subseteq MC$, we have $\phi_{F_i^g}(\pi^*, \pi) \geq 0$ (and by Theorem G.10, we have $\phi_{W_i \cup Z_i}(\pi^*, \pi) \geq 0$ as well). Hence, we have that $\phi_{F_i^g \cup W_i \cup Z_i}(\pi^*, \pi) \geq 0$, and we are done.

Case 3.3: Suppose GG_j and GG_k correspond to $\mathcal{G}_j = 0$ and $\mathcal{G}_k = 1$ respectively (although the gadget is not completely symmetric with respect to j and k , the proof of the case $\mathcal{G}_j = 1$

and $\mathcal{G}_k = 0$ is analogous). Since GG_i does not comply with its input, we have that $g_i^0 \notin MC$ or $g_i^1 \notin \pi(\ell_i^g)$. First, suppose $g_i^0 \notin MC$. Since the input gadgets are valid, we have $g_j^0 \in MC \neq \pi(g_i^0)$, and $g_k^0 \in \pi(\ell_k^g)$. If $\ell_i^g \notin \pi(g_i^0)$, we observe that the agents of T_i^g and L_i^g form a trade-off, where one group prefers π if and only if the other prefers π^* . Then, either we have $\{w_i, z_i\} \subseteq \pi(g_i^0)$, and then all agents in $W_i \cup Z_i$ prefer π^* and we are done, or $\{w_i, z_i\} \not\subseteq \pi(g_i^0)$, and thus all agents in F_i^g prefer π^* , and, by Theorem G.10, we are done. If $\ell_i^g \in \pi(g_i^0)$, then all agents in T_i^g prefer π^* , and it is easy to see that $\phi_{F_i^g \cup W_i \cup Z_i}(\pi^*, \pi) \geq 0$, so we are done.

Hence, we must have $g_i^0 \in MC$ and be in the case where $g_i^1 \notin \pi(\ell_i^g)$. Therefore, we have that all agents in T_i^g prefer π^* . Moreover, since $g_k^1 \in MC$, we have that any agent in $W_i \cup Z_i$ would obtain negative utility if they were in MC . Hence, it is easy to see that $\phi_{F_i^g \cup W_i \cup Z_i}(\pi^*, \pi) \geq 0$, so we are done. $\square$

Lemma G.15. *Let $i \in [n]$. Then $|\{a_i^1, a_i^0\} \cap \pi(x_i)| = 1$.*

Proof. First, assume for contradiction that $|\{a_i^1, a_i^0\} \cap \pi(x_i)| = 2$. Then it is easy to see that all agents in AG_i prefer π^* , i.e., $\phi_{AG_i}(\pi^*, \pi) = 3m + 3$. Hence, by Lemmas G.5, G.9, and G.12, we have that $\phi(\pi^*, \pi) > 0$ (even if all V -agents prefer π), a contradiction. Hence, we have that $|\{a_i^1, a_i^0\} \cap \pi(x_i)| \leq 1$.

Second, Assume for contradiction that $|\{a_i^1, a_i^0\} \cap \pi(x_i)| = 0$. If there exist $a \in T_i^a$ and $a' \in F_i^a$ such that $a \in \pi(a')$, it is easy to see that all agents in $T_i^a \cup F_i^a$ prefer π^* over π . Otherwise, no agent in AG_i can prefer π , while all of T_i^a or all of F_i^a must strictly prefer π^* (since only one of those sets can have ℓ_i^a). Thus, in both cases we get $\phi_{AG_i}(\pi^*, \pi) \geq m + 1$. Hence, by Lemmas G.5, G.9, and G.12, we have that $\phi(\pi^*, \pi) > 0$ (even if all V -agents prefer π), a contradiction. Thus, $|\{a_i^1, a_i^0\} \cap \pi(x_i)| = 1$. $\square$

Lemma G.16. *All assignment gadgets are π -valid.*

Proof. Let $i \in [n]$. By Theorem G.15 we have $|\{a_i^1, a_i^0\} \cap MC| = 1$. Without loss of generality assume $a_i^1 \in MC$. If $a_i^0 \notin \pi(\ell_i^a)$, then all agents in $F_i^a \cup L_i^a$ prefer π^* over π , and agents in T_i^a cannot prefer π over π^* . Therefore, $\phi_{AG_i}(\pi^*, \pi) \geq 2m + 2$. Hence, by Lemmas G.5, G.9, and G.12, we have that $\phi(\pi^*, \pi) > 0$ (even if all V -agents prefer π), a contradiction. Hence, $a_i^0 \in \pi(\ell_i^a)$, and the gadget is π -valid. $\square$

Notice that at this point, we can elicit from π an input string to our circuit, namely the string that the partition of assignment gadgets corresponds to. For the remainder of this section, we call this string $\mathbf{x}'$.

We are ready to prove statements concerning whole circuits. Recall that C_j is the set of all agents associated with circuit $\mathcal{C}_j$, which are the agents of the gates of $\mathcal{C}_j$ and its voter agent.

Lemma G.17. *Let $\mathcal{C}_j$ be a circuit. If some gate of $\mathcal{C}_j$ does not π -comply with $\mathbf{x}'$, then there exists a gate $\mathcal{G}_i$ of $\mathcal{C}_j$ such that $\phi_{GG_i}(\pi^*, \pi) \geq 2$.*

Proof. Consider the gate $\mathcal{G}_i$ with the smallest index that does not π -comply with $\mathbf{x}'$. Since the gates are topologically ordered, its inputs are π -valid and π -comply with $\mathbf{x}'$ (if $\mathcal{G}_i$ happens to be a Copy-gate, its inputs are π -valid by Theorem G.16, and they π -comply with $\mathbf{x}'$ by definition of $\mathbf{x}'$). Thus, by Theorem G.14, we have that $\phi_{GG_i}(\pi^*, \pi) \geq 2$. $\square$

Lemma G.18. *Let $\mathcal{C}_j$ be a circuit. If some gate of $\mathcal{C}_j$ does not π -comply with $\mathbf{x}'$, then $\phi_{C_j}(\pi^*, \pi) \geq 1$.*

Proof. By Theorem G.17 there exists a gate $\mathcal{G}_i$ with $\phi_{GG_i}(\pi^*, \pi) \geq 2$. Thus, by Theorem G.12, we have that $\phi_{C_j}(\pi^*, \pi) \geq 1$ (even if the voter agent of $\mathcal{C}_j$ prefers π). $\square$

Lemma G.19. *Let $\mathcal{C}_j$ be a circuit with voter agent v . If all gates of $\mathcal{C}_j$ π -comply with $\mathbf{x}'$ then $\text{sgn}(\phi_v(\pi^*, \pi)) = \text{sgn}(\mathcal{C}_j(\mathbf{x}^*) - \mathcal{C}_j(\mathbf{x}'))$.*

Proof. Since all gates of $\mathcal{C}_j$ comply with $\mathbf{x}'$, the set of gates whose positive representatives are in MC are exactly the gates $\mathcal{G}_i$ for which $XX'(\mathcal{G}_i) = 1$. By definition of π^* , the set of gates whose positive representatives are in MC^* are exactly the gates $\mathcal{G}_i$ for which $XX^*(\mathcal{G}_i) = 1$. In particular, these two observation hold for the output gates of $\mathcal{C}_j$. Furthermore, by Theorems G.6 to G.8, and by definition of π^* , in both MC and MC^* these positive representatives of the gates are the only agents that v assigns a non-zero value to. The result then follows from the definition of the valuation function of $\mathbf{v}$. $\square$

Lemma G.20. *Let $\mathcal{C}_j$ be a circuit. Then the following holds.*

- If $\phi_{\mathcal{C}_j}(\pi^*, \pi) < 0$ then $\mathcal{C}_j(\mathbf{x}^*) < \mathcal{C}_j(\mathbf{x}')$.
- If $\phi_{\mathcal{C}_j}(\pi^*, \pi) = 0$ then $\mathcal{C}_j(\mathbf{x}^*) = \mathcal{C}_j(\mathbf{x}')$.

Proof. First, in both cases all gates of $\mathcal{C}_j$ π -comply with $\mathbf{x}'$, as otherwise by Theorem G.18 we have $\phi_{\mathcal{C}_j}(\pi^*, \pi) \geq 1$, a contradiction. Let v be the voter agent of $\mathcal{C}_j$. Suppose $\phi_{\mathcal{C}_j}(\pi^*, \pi) < 0$. Then, by Theorem G.12, v must prefer π . Hence, by Theorem G.19 we have that $\mathcal{C}_j(\mathbf{x}^*) < \mathcal{C}_j(\mathbf{x}')$. Suppose $\phi_{\mathcal{C}_j}(\pi^*, \pi) = 0$. Then, by Theorem G.12, v must be indifferent between π and π^* . Hence, by Theorem G.19 we have that $\mathcal{C}_j(\mathbf{x}^*) = \mathcal{C}_j(\mathbf{x}')$. $\square$

Lemma G.21. *Let $\mathcal{C}_j$ be a circuit. If $\mathcal{C}_j(\mathbf{x}^*) > \mathcal{C}_j(\mathbf{x}')$ then $\phi_{\mathcal{C}_j}(\pi^*, \pi) > 0$.*

Proof. Let v be the V -agent corresponding to $\mathcal{C}_j$. By Theorems G.6 to G.8, and by definition of π^* , we have that MC and MC^* both contain all V - and X -agents, and no L -agent. Assume for contradiction that $\phi_{\mathcal{C}_j}(\pi^*, \pi) \leq 0$. Then by Theorem G.18 we have that all gates π -comply with $\mathbf{x}'$. Hence, since $\mathcal{C}_j(\mathbf{x}^*) > \mathcal{C}_j(\mathbf{x}')$, by Theorem G.19 we have $\phi_v(\pi^*, \pi) > 0$. Hence, by Theorem G.12 we have $\phi_{\mathcal{C}_j}(\pi^*, \pi) > 0$. $\square$

For the following lemmas, we observe that the sets $C := \bigcup_{j \in [m]} \mathcal{C}_j$, $AG := \bigcup_{i \in [n]} AG_i$, and X are disjoint, and their union is N . Thus, we have that

$$\phi(\pi^*, \pi) = \phi_C(\pi^*, \pi) + \phi_{AG}(\pi^*, \pi) + \phi_X(\pi^*, \pi) \quad (25)$$

Lemma G.22. *It holds that $\mathbf{x}' = \mathbf{x}^*$.*

Proof. Denote by $\kappa_{\mathbf{x}^*}$ the number of circuits $\mathcal{C}_j$ with $\mathcal{C}_j(\mathbf{x}^*) > \mathcal{C}_j(\mathbf{x}')$, and by $\kappa_{\mathbf{x}'}$ the number of circuits $\mathcal{C}_j$ with $\mathcal{C}_j(\mathbf{x}^*) < \mathcal{C}_j(\mathbf{x}')$. Similarly, denote by $\lambda_{\mathbf{x}^*}$ and $\lambda_{\mathbf{x}'}$ the number of circuits $\mathcal{C}_j$ with $\phi_{\mathcal{C}_j}(\pi^*, \pi) > 0$ and $\phi_{\mathcal{C}_j}(\pi^*, \pi) < 0$, respectively. By Theorem G.20, it holds that any circuit $\mathcal{C}_j$ with $\phi_{\mathcal{C}_j}(\pi^*, \pi) < 0$ has $\mathcal{C}_j(\mathbf{x}^*) < \mathcal{C}_j(\mathbf{x}')$, and therefore

$$\kappa_{\mathbf{x}'} \geq \lambda_{\mathbf{x}'}. \quad (26)$$

By Theorem G.21, it holds that any circuit $\mathcal{C}_j$ with $\mathcal{C}_j(\mathbf{x}^*) > \mathcal{C}_j(\mathbf{x}')$ has $\phi_{\mathcal{C}_j}(\pi^*, \pi) > 0$, and therefore

$$\lambda_{\mathbf{x}^*} \geq \kappa_{\mathbf{x}^*}. \quad (27)$$

Assume for contradiction that $\mathbf{x}' \neq \mathbf{x}^*$. Then by strong popularity of $\mathbf{x}^*$ we have that $\kappa_{\mathbf{x}^*} > \kappa_{\mathbf{x}'}$. Hence, by Equations (26) and (27) we have $\lambda_{\mathbf{x}^*} \geq \kappa_{\mathbf{x}^*} > \kappa_{\mathbf{x}'} \geq \lambda_{\mathbf{x}'}$. In other words, there are more circuits whose popularity margin favors $\mathbf{x}^*$ over $\mathbf{x}'$ than the converse. Additionally, by

Theorem G.13 we have that gates whose popularity margin favors $\mathbf{x}'$ have exactly $\phi_{C_j}(\pi^*, \pi) = -1$. Therefore, counting all circuits, we have $\sum_{C_j} \phi_{C_j}(\pi^*, \pi) \geq 1$, that is $\phi_{\bigcup_{j \in [m]} C_j}(\pi^*, \pi) \geq 1$. Thus, by Theorems G.5 and G.9 and eq. (25) we have that $\phi(\pi^*, \pi) = \phi_{\bigcup_{j \in [m]} C_j}(\pi^*, \pi) + \phi_{\bigcup_{i \in [n]} AG_i}(\pi^*, \pi) + \phi_X(\pi^*, \pi) \geq 1$, a contradiction to the definition of π . $\square$

Lemma G.23. *Let GD_i be an assignment or gate gadget. Then $\phi_{GD_i}(\pi^*, \pi) = 0$.*

Proof. Fix some circuit C_j , with V -agent v . If all gates of C_j π -comply with $\mathbf{x}'$, by Theorem G.19 we have $\phi_v(\pi^*, \pi) = 0$, and thus by Theorem G.12 we have $\phi_{C_j}(\pi^*, \pi) = 0$. If not all gates of C_j π -comply with $\mathbf{x}'$, then by Theorem G.18 we have that $\phi_{C_j}(\pi^*, \pi) \geq 1$. Thus, either way we have $\phi_{C_j}(\pi^*, \pi) \geq 0$. This holds for all circuits. Hence, by Theorems G.5 and G.9 and eq. (25), if even one circuit has a gate which does not comply with $\mathbf{x}'$, then $\phi(\pi^*, \pi) > 0$, a contradiction. Hence, all gates comply with $\mathbf{x}'$, and we have that $\mathbf{x}' = \mathbf{x}^*$ by Theorem G.22, implying $C_j(\mathbf{x}^*) = C_j(\mathbf{x}')$ for all $j \in [m]$. Thus, by Theorem G.19 all V -agents are indifferent between π^* and π . Therefore, by Theorems G.5, G.9 and G.12 and the above observation, we conclude that each component— X -agents, assignment gadgets, gate gadgets, and V -agents—has a popularity margin of at least 0 with respect to (π^*, π) . Since these are disjoint components whose union is N , this implies that each assignment gadget and each gate gadget must have a popularity margin of exactly 0, as otherwise π^* is more popular than π , a contradiction. $\square$

Lemma G.24. *Let GD_i be an assignment or gate gadget, and let $p \in GD_i$. Then $\phi_p(\pi^*, \pi) = 0$.*

Proof. First, assume for contradiction that $\phi_p(\pi^*, \pi) < 0$. Then we must have either an X -agent and an L -agent together in a coalition (contradicting Theorem G.7), or GD_i or one of its inputs is invalid, a contradiction (if GD_i is an And-gadget, one needs also to consider coalitions involving ℓ_i^g and agents from W_i or Z_i , but this immediately gives $\phi_{GD_i}(\pi^*, \pi) > 0$, a contradiction to Theorem G.23). Therefore, we have $\phi_p(\pi^*, \pi) \geq 0$. Since this holds for all agents of GD_i , we cannot have $\phi_p(\pi^*, \pi) > 0$, as this would imply $\phi_{GD_i}(\pi^*, \pi) > 0$, a contradiction to Theorem G.23. Hence we have $\phi_p(\pi^*, \pi) = 0$. $\square$

Lemma G.25. *All gate gadgets π -comply with $\mathbf{x}'$ (and are therefore π -valid).*

Proof. Assume towards contradiction that some gate does not comply with $\mathbf{x}'$. Then by Theorem G.17 there exists a gate $\mathcal{G}_i$ with $\phi_{\mathcal{G}_i}(\pi^*, \pi) \geq 2$, a contradiction to Theorem G.23. $\square$

Lemma G.26. *In π , all one-way replicas are in the coalition of their respective origin agent.*

Proof. Any replica agent that is not with its origin agent would prefer π^* , a contradiction to Theorem G.24. $\square$

Using the knowledge we have gained about the structure of π , we will prove that, in fact, $\pi = \pi^*$, a contradiction.

Proof. In the partition π , we have that:

- All X agents are in MC (by Theorem G.6).
- For every input bit $\mathbf{x}_i^*$, if $\mathbf{x}_i^* = 1$ then $T_i^a \subseteq MC$, and F_i^a and L_i^a are in the same coalition, and if $\mathbf{x}_i^* = 0$ then $F_i^a \subseteq MC$ and T_i^a and L_i^a are in the same coalition (by Theorems G.16, G.22 and G.26).
- For every gate $\mathcal{G}_i$, if $\mathbf{x}^*(\mathcal{G}_i) = 1$ then $T_i^g \subseteq MC$ and $F_i^g \cup L_i^g \in \pi$, and if $\mathbf{x}^*(\mathcal{G}_i) = 0$ then $F_i^g \subseteq MC$ and $T_i^g \cup L_i^g \in \pi$ (by Theorems G.22, G.25 and G.26).

- All V -agents are in MC (by Theorem G.8).

Furthermore, by Theorems G.4 and G.7 we have that all coalitions containing L -agents are separated from one another and from MC . Thus, it remains to analyze the coalitions of the W - and Z -agents. Let $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$ be an And-gate. We make a case distinction based on the values $\mathcal{G}_j$ and $\mathcal{G}_k$ obtain by $\mathbf{x}' = \mathbf{x}^*$.

Case 1: Suppose $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 1$. Then $\mathbf{x}^*(\mathcal{G}_i) = 1$. Hence, $T_i^g \subseteq MC$, and any other coalition in π contains either L -agents, or W - or Z -agents corresponding to a different gate (by the analysis above). Hence, W_i and Z_i cannot intersect with any of the coalitions listed above, as that would yield a negative utility for them, implying they prefer π^* over π , contradicting Theorem G.24. Therefore, by Theorem G.26 we must have $\{W_i, Z_i\} \subseteq \pi$.

Case 2: Suppose $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 0$. Then $F_j^g \cup F_k^g \subseteq MC$, and any other coalition in π contains either L -agents, or W - or Z -agents corresponding to a different gate. Thus, by similar reasoning to the previous case, we must have $\{W_i, Z_i\} \subseteq \pi$.

Case 3: Suppose $\mathbf{x}^*(\mathcal{G}_j) = 1 \wedge \mathbf{x}^*(\mathcal{G}_k) = 0$. Then $T_j^g \cup F_k^g \subseteq MC$, and any other coalition in π contains either L -agents, or W - or Z -agents corresponding to a different gate. Thus, by similar reasoning to the previous cases, we must have $Z_i \in \pi$. Furthermore, we must have $w_i \in MC$, as otherwise $u_{g_i^0}(\pi) \leq 12 < 13 = u_{g_i^0}(\pi^*)$, a contradiction to Theorem G.24. Thus, by Theorem G.26, we in fact have that $W_i \subseteq MC$.

Case 4: Suppose $\mathbf{x}^*(\mathcal{G}_j) = 0 \wedge \mathbf{x}^*(\mathcal{G}_k) = 1$. Then $F_j^g \cup T_k^g \subseteq MC$, and any other coalition in π contains either L -agents, or W - or Z -agents corresponding to a different gate. Thus, by similar reasoning to the previous cases, we must have $W_i \in \pi$. Furthermore, we must have $z_i \in MC$, as otherwise $u_{g_i^0}(\pi) \leq 11 < 13 = u_{g_i^0}(\pi^*)$, a contradiction to Theorem G.24. Thus, by Theorem G.26, we in fact have that $Z_i \subseteq MC$.

Indeed, this shows that $\pi = \pi^*$, a contradiction to the choice of π . This concludes the proof that the existence of a Condorcet string entails the existence of a strongly popular partition. $\square$

G.4 Strongly Popular Partition implies Condorcet string

Suppose π^* is strongly popular. Then it is also Pareto-optimal. We want to establish some properties of π^* to help us derive a Condorcet string $\mathbf{x}^*$. Denote $MC^* = \pi^*(x_1)$.

Lemma G.27. *In each coalition $S \in \pi^*$, we have that $|\{a \in S : u_a(\pi^*) > 0\}| > |\{a \in S : u_a(\pi^*) < 0\}|$.*

Proof. If not, then it is at least as popular to dissolve S into singletons, a contradiction to the strong popularity of π^* . $\square$

Lemma G.28. *It holds that $X \subseteq MC^*$.*

Proof. Assume otherwise, then it is more popular to extract all X -agents from their coalitions and form X as a coalition (because all X -agents prefer this, and $|X| > \frac{|N|}{2}$), a contradiction to the strong popularity of π^* . $\square$

Lemma G.29. *It holds that $L \cap MC^* = \emptyset$.*

Proof. Assume otherwise, then by Theorem G.28 all X -agents obtain negative utility. Hence, it is more popular to remove all agent not in X from MC^* (because all X -agents prefer this, and $|X| > \frac{|N|}{2}$), a contradiction. $\square$

Lemma G.30. *It holds that $V \subseteq MC^*$.*

Proof. Let $V' := V \setminus MC^*$, and assume $V' \neq \emptyset$. Since $X \subseteq MC^*$ by Theorem G.28, and since are m voter agents, for all $v' \in V'$ we have $u_{v'}(\pi^*) < m \cdot 2^{n+1}$ (2^{n+1} for each of the $m - 1$ other V -agents, and strictly less than 2^{n+1} from all of its preferred output gates together). Consider the partition π obtained by extracting all V -agents not in MC^* from their coalitions, and adding them to MC^* . Since $|V| = m$ and $x_1 \in MC^*$, for all $v' \in V'$ we have $u_{v'}(\pi^*) \geq m \cdot 2^{n+1}$, which is an improvement for them. Moreover, no one objects that V -agents leave their coalitions because no one in $N \setminus V$ assigns a positive value to them, and no one objects that they join MC^* because only L -agents assign negative value to them, but $L \cap MC^* = \emptyset$ by theorem G.29. Hence, it is a Pareto improvement, a contradiction. $\square$

Lemma G.31. *The following statements hold.*

- Let $\mathcal{G}_i$ be a gate. We have that $\pi^*(\ell_i^g) \subseteq L_i^g \cup T_i^g$ or $\pi^*(\ell_i^g) \subseteq L_i^g \cup F_i^g$.
- Let $i \in [n]$. We have that $\pi^*(\ell_i^a) \subseteq L_i^a \cup T_i^a$ or $\pi^*(\ell_i^a) \subseteq L_i^a \cup F_i^a$.

Proof. We only prove the first statement, as the second proof is analogous. First, observe that $\pi(\ell_i^g)$ cannot contain an agent from T_i^g and an agent from F_i^g together, by Theorem G.27; indeed, agents outside the gadget GG_i have value $-\infty$ for ℓ_i^g , and within the gadget only ℓ_i^g and $\hat{\ell}_i^g$ may gain positive utility in such a coalition and at least two must gain negative utility. Second, assume some agent $p \notin GG_i$ is in $\pi^*(\ell_i^g)$. Then p and ℓ_i^g obtain negative utility. Notice that there may be at most two agents who obtain positive utility in $\pi^*(\ell_i^g)$, namely, either g_i^1 and $\hat{g}_i^1$ or g_i^0 and $\hat{g}_i^0$. Hence, we have a contradiction to Theorem G.27.

The first and second part of the proof combined imply the statement is true. $\square$

Lemma G.32. *Let $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$ be an And-gate. Then:*

- It holds that $\pi^*(w_i) \cap \{z_i, g_j^0\} = \emptyset$.
- It holds that $\pi^*(z_i) \cap \{w_i, g_k^0\} = \emptyset$.

Proof. We prove the first statement, as the second proof is analogous. If $z_i \in \pi^*(w_i)$, then it would be at least as popular to remove w_i and, if $\hat{w}_i \in \pi^*(w_i)$, remove $\hat{w}_i$ as well, from this coalition (since at least two agents, namely w_i and z_i , prefer this deviation, while at most two agents, namely g_i^0 and $\hat{g}_i^0$, prefer π^*), a contradiction to the strong popularity of π^* .

Similarly, if $g_j^0 \in \pi^*(w_i)$, then it would be at least as popular to remove w_i and, if $\hat{w}_i \in \pi^*(w_i)$, remove $\hat{w}_i$ as well (since at least two agents, namely w_i and g_j^0 , prefer this deviation, while at most two agents, namely g_i^0 and $\hat{g}_i^0$, prefer π^*), a contradiction to the strong popularity of π^* . $\square$

Lemma G.33. *Let $i \in [n]$. Then $a_i^0 \notin \pi^*(a_i^1)$.*

Proof. Assume $a_i^0 \in \pi^*(a_i^1)$. Notice that any agent, apart from ℓ_i^a and $\hat{\ell}_i^a$, who assigns a positive value to one of those two agents assigns a value of $-\infty$ to the other one. Furthermore, a_i^1 and a_i^0 assign a value of $-\infty$ to each other. Moreover, replica agents are only assigned a positive value by their origin agent. Now, consider the partition π obtained from π^* by extracting all agents of T_i^a and F_i^a from their coalitions, and forming the coalitions T_i^a and F_i^a . Clearly, all agents in $T_i^a \cup F_i^a$ prefer π (the replica agents have a utility of $10m$ in π , and strictly less than that in π^*). By the same reasoning as above, the only agents that may prefer π^* over π are ℓ_i^g and $\hat{\ell}_i^g$. Hence, we have $\phi(\pi^*, \pi) < 0$, contradicting the strong popularity of π^* . $\square$

Lemma G.34. *The following statements hold.*

1. Let $x \in X$. Then $u_x(\pi^*) \leq |X| - 1$.
2. Let $\ell \in L^g$. Then $u_\ell(\pi^*) \leq 11$.
3. Let $i \in [n]$. Then for agent $p \in T_i^a \cup F_i^a \cup L_i^a$ we have $u_p(\pi^*) \leq 10m + 1$.
4. Let $\mathcal{G}_i$ be a Not-or Copy-gate. Then for agent $p \in T_i^g \cup F_i^g$ we have $u_p(\pi^*) \leq 11$.
5. Let $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$ be an And-gate. Then
 - (a) for agent $p \in T_i^g$ we have $u_p(\pi^*) \leq 12$;
 - (b) for agent $p \in F_i^g$ we have $u_p(\pi^*) \leq 13$;
 - (c) for agent $p \in W_i \cup Z_i$ we have $u_p(\pi^*) \leq 0$.

Proof. One may verify this by summing the positive utilities available for each of those agents, while taking into account which agents may and may not form a coalition together, as established in previous lemmas. Specifically, Item 2 follows from Theorem G.31, Item 3 follows from Theorems G.29 and G.31, Items 4 and 5a follow from Theorem G.31, and Item 5b follows from Theorems G.31 and G.32 (Items 1 and 5c are immediate from the sum of positive utilities those agents assign). $\square$

Lemma G.35. *Let $i \in [n]$. Then $|\{a_i^1, a_i^0\} \cap MC^*| = 1$.*

Proof. By Theorem G.33 we have that $|\{a_i^1, a_i^0\} \cap MC^*| \leq 1$. Assume for contradiction that $\{a_i^1, a_i^0\} \cap MC^* = \emptyset$. Then by Theorem G.28 we have $x_i \notin \pi^*(a_i^1) \cup \pi^*(a_i^0)$. Take some arbitrary string $\mathbf{y}$, say $\mathbf{y} = 0^n$, and consider the following partition π . All X - and V -agents are in the same coalition, which we call $MC_{\mathbf{y}}$; all A - and G -gadgets π -comply with $\mathbf{y}$ (namely, one of the gadget's core agent is in $MC_{\mathbf{y}}$ while the other is with its alternative agent, such that all gadgets comply with their inputs); and all one-way replicas are in the coalitions of their origin agent. By Theorem G.34, one may verify that all but the V -agents cannot prefer π^* over π . Furthermore, since $\{a_i^1, a_i^0\} \cap MC^* = \emptyset$, we must have $\phi_{AG_i}(\pi^*, \pi) \leq -(m+1)$. An intuitive way to see this is that we have $u_p(\pi) = 10m + 1$ for all $p \in AG_i$, but in π^* we have a trade off between T_i^a and F_i^a , since they both need ℓ_i^a to match the utility they obtain in π . Thus, even if all V -agents prefer π^* we have $\phi(\pi^*, \pi) < 0$, contradicting the strong popularity of π^* . Thus, $|\{a_i^1, a_i^0\} \cap MC^*| = 1$. $\square$

By Theorems G.33 and G.35, we have that for each A -gadget AG_i , exactly one of a_i^1 and a_i^0 is in MC^* . For the rest of the proof, let $\mathbf{x}^* = \mathbf{x}_1^*, \dots, \mathbf{x}_n^*$ denote the string where $\mathbf{x}_i^* = 1$ if $a_i^1 \in MC^*$, and $\mathbf{x}_i^* = 0$ if $a_i^0 \in MC^*$.

Lemma G.36. *Let $i \in [n]$. Then the assignment gadget AG_i π^* -complies with $\mathbf{x}^*$.*

Proof. By Theorem G.35 we have that of the agents a_i^1 and a_i^0 are in MC^* while the other is not. Without loss of generality, assume $a_i^1 \in MC^*$. By Theorem G.29 we have that $\ell_i^a \notin MC^*$. Assume for contradiction that $a_i^0 \notin \pi^*(\ell_i^a)$. Then:

$$\forall p \in L_i^a u_p(\pi^*) < 10m + 1. \quad (28)$$

Denote by π the following partition:

- All X - and V -agents are in a coalition denoted $MC_{\mathbf{x}^*}$ (to which more agents will be added).
- all assignment and gate gadgets π -comply with $\mathbf{x}^*$.
- For an And-gadget $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$:

- If $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 1$ or $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 0$, then $W_i \in \pi$, and $Z_i \in \pi$.
- If $\mathbf{x}^*(\mathcal{G}_j) = 0$ and $\mathbf{x}^*(\mathcal{G}_k) = 1$ then $W_i \in \pi$, and $Z_i \subseteq MC_{\mathbf{x}^*}$.
- If $\mathbf{x}^*(\mathcal{G}_j) = 1$ and $\mathbf{x}^*(\mathcal{G}_k) = 0$ then $Z_i \in \pi$, and $W_i \subseteq MC_{\mathbf{x}^*}$.

- All one-way replicas are with their respective origin agent.

Notice that in π all agents $p' \in N \setminus V$ obtain the maximal utility specified for them in Theorem G.34, and therefore $u_{p'}(\pi^*, \pi) \leq 0$. In particular, by Equation (28) we even have that $\phi_{AG_i}(\pi^*, \pi) < -(m+1)$. Thus, since $|V| = m$, even if all V -agents prefer π^* over π , we have that π is more popular than π^* , a contradiction to the strong popularity of π^* . $\square$

Lemma G.37. *Let $i \in [n]$. Then $T_i^a \subseteq \pi^*(a_i^1)$, and $F_i^a \subseteq \pi^*(a_i^0)$.*

Proof. Without loss of generality, assume $p \in T_i^a \setminus \pi^*(a_i^1)$. Consider the partition π obtained from π^* by extracting all agents in $T_i^a \setminus \pi^*(a_i^1)$ from their coalitions, and adding them to $\pi^*(a_i^1)$. By Theorem G.36 we have that AG_i π^* -complies with $\mathbf{x}^*$, and therefore either $\pi^*(a_i^1) = MC^*$ or $\pi^*(a_i^1) = \pi^*(\ell_i^a)$. Either way, by Theorem G.31 and Theorem G.29 we have that no agent will object to adding the replicas of a_i^1 to $\pi^*(a_i^1)$. Furthermore, no agent would object to those replica agents leaving their coalitions, as any agent outside T_i^a assigns at most zero to them. However, a_i^1 clearly prefers π over π^* . Thus, π is a Pareto improvement from π^* , a contradiction. $\square$

Lemma G.38. *Let $\mathcal{C}_l$ be a circuit. Then all gates in $\mathcal{C}_l$ π^* -comply with $\mathbf{x}^*$.*

Proof. Assume otherwise. Let $\mathcal{G}_t$ be the gate with the smallest index in $\mathcal{C}_l$ which does not π^* -comply with $\mathbf{x}^*$. Then, since the gates are topologically ordered, the inputs of $\mathcal{G}_t$ π^* -comply with $\mathbf{x}^*$ (if $\mathcal{G}_t$ is a Copy-gate, its input is an assignment gadget, which all π^* -comply with $\mathbf{x}^*$ by Theorem G.36). Since $\mathcal{G}_t$ does not comply with $\mathbf{x}^*$, but its inputs do, one may verify that at least one agent in GG_t will obtain a utility strictly smaller than the maximal utility specified in Theorem G.34.

Let π be the partition obtained from π^* by rearranging only the agents of $\mathcal{C}_l$ such that:

- All gates comply with $\mathbf{x}^*$.
- For any core agent p that is not in MC^* , p is with her alternative agent.
- For an And-gadget $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$:
 - If $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 1$ or $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 0$, then we form the coalitions W_i and Z_i .
 - If $\mathbf{x}^*(\mathcal{G}_j) = 0$ and $\mathbf{x}^*(\mathcal{G}_k) = 1$ then we form the coalition W_i , and add Z_i to MC^* .
 - If $\mathbf{x}^*(\mathcal{G}_j) = 1$ and $\mathbf{x}^*(\mathcal{G}_k) = 0$ then we form the coalition Z_i , and add W_i to MC^* .
- All one-way replicas are with their respective origin agent.

Denote $MC_\pi = \pi(x_1)$. Let v denote the V -agent of $\mathcal{C}_l$. It is clear that agents outside $\mathcal{C}_l$ are not affected by this deviation (agents in $MC^* \setminus \mathcal{C}_l$ assign a value of zero to any agent that was added to MC^*). Within $\mathcal{C}_l$, according to Theorem G.34 all agents apart from v obtain utility at least as much as they obtain in π^* : All L -agents obtain a utility of 11, and G -agents who belong to a Copy- or Not-gate obtain a utility of 11. The And-gates require a slightly more careful analysis. Let $\mathcal{G}_i = \mathcal{G}_j \wedge \mathcal{G}_k$ be an And-gate in $\mathcal{C}_l$. It is easy to see that all agents in $W_i \cup Z_i$ obtain a utility of 10, regardless of the values of $\mathbf{x}^*(\mathcal{G}_j)$ and $\mathbf{x}^*(\mathcal{G}_k)$. Further, if $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 1$, then $T_i^g \cup \{g_j^1, g_k^1\} \subseteq MC_\pi$, and thus $\forall p \in T_i^g$ $u_p(\pi) = 12 \geq u_p(\pi^*)$; otherwise, we have $T_i^g \cup L_i^g \in \pi$, and thus $\forall p \in T_i^g$ we have $u_p(\pi) = 12 \geq u_p(\pi^*)$. It remains to consider the agents of F_i^g , for which we make the following case distinction.

- If $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 1$, then $F_i^g \cup L_i^g \in \pi$, and thus $\forall p \in F_i^g$ we have $u_p(\pi) = 13 \geq u_p(\pi^*)$.
- If $\mathbf{x}^*(\mathcal{G}_j) = \mathbf{x}^*(\mathcal{G}_k) = 0$, then $F_i^g \cup \{g_j^0, g_k^0\} \subseteq MC_\pi$, and thus $\forall p \in F_i^g$ we have $u_p(\pi) = 13 \geq u_p(\pi^*)$.
- If $\mathbf{x}^*(\mathcal{G}_j) = 0$ and $\mathbf{x}^*(\mathcal{G}_k) = 1$ then $Z_i \cup \{g_j^0\} \subseteq MC_\pi$, and thus $\forall p \in F_i^g$ we have $u_p(\pi) = 13 \geq u_p(\pi^*)$.
- If $\mathbf{x}^*(\mathcal{G}_j) = 1$ and $\mathbf{x}^*(\mathcal{G}_k) = 0$ then $W_i \cup \{g_k^0\} \subseteq MC_\pi$, and thus $\forall p \in F_i^g$ we have $u_p(\pi) = 13 \geq u_p(\pi^*)$.

Thus, since we know that in gadget GG_t at least one agent obtains a higher utility in π than in π^* , we have that π is at least as popular as π^* (even if v prefers π^* over π), a contradiction to the strong popularity of π^* . $\square$

Lemma G.39. *All replica agents are in the coalition of their respective origin agent.*

Proof. For replicas of A -agents, the statement holds by Theorem G.37. For replicas of G -agents, since we know that all gate and assignment gadget π^* -comply with $\mathbf{x}^*$, it can be easily seen that if, in π^* , some replicas are not in the coalition of their origin agent, then it would be a Pareto improvement to extract all replica agents from their coalitions and place them with their origin agent. $\square$

We now have a complete characterization of the structure of π^* . We will show that $\mathbf{x}^*$ must be a Condorcet string for the instance $\mathfrak{C}$.

Proof. Assume for contradiction that there exists another string $\mathbf{x}' \neq \mathbf{x}^*$ with

$$\sum_{i=1}^m (\text{sgn}(\mathcal{C}_i(\mathbf{x}^*) - \mathcal{C}_i(\mathbf{x}')))) \leq 0 \quad (29)$$

Consider the partition π obtained from π^* by rearranging the A -gadgets and G -gadgets such that they comply with $\mathbf{x}'$, while maintaining that replicas stay with their origin agents; for W - and Z -agents, we set their coalitions as defined for the partition π in the proof of Theorem G.38 for And-gates of the circuit $\mathcal{C}_j$. It is clear that all but the V -agents are indifferent between the partitions (the analysis is similar to that in the proof of Theorem G.38). Furthermore, by Equation (29), and by definition of the utility function of the V -agents, we have that the number of V -agents who prefer π over π^* is at least as much as the number of V -agents who prefer π^* over π , a contradiction to the strong popularity of π^* . This concludes the proof that the existence of a strongly popular partition entails the existence of a Condorcet string. $\square$

References

- [ACHI17] Gadi Aleksandrowicz, Hana Chockler, Joseph Y Halpern, and Alexander Ivrii. The computational complexity of structure-based causality. *Journal of Artificial Intelligence Research*, 58:431–451, 2017.
- [Aki20] Ethan Akin. Generalized intransitive dice: Mimicking an arbitrary tournament. *Journal of Dynamics and Games*, 8(1):1–20, 2020.

- [Bab85] László Babai. Trading group theory for randomness. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, pages 421–429, 1985.
- [BB22] Felix Brandt and Martin Bullinger. Finding and recognizing popular coalition structures. *Journal of Artificial Intelligence Research*, 74:569–626, 2022.
- [BG82] Andreas Blass and Yuri Gurevich. On the unique satisfiability problem. *Information and Control*, 55(1-3):80–88, 1982.
- [BG25] Martin Bullinger and Matan Gilboa. Settling the complexity of popularity in additively separable and fractional hedonic games. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 3771–3779, 8 2025.
- [BHZ87] Ravi B Boppana, Johan Hastad, and Stathis Zachos. Does co-np have short interactive proofs? *Information Processing Letters*, 25(2):127–132, 1987.
- [BJ02] Anna Bogomolnaia and Matthew O Jackson. The stability of hedonic coalition structures. *Games and Economic Behavior*, 38(2):201–230, 2002.
- [Cai07] Jin-Yi Cai. $S_2^P \subseteq ZPP^{NP}$. *Journal of Computer and System Sciences*, 73(1):25–35, 2007.
- [Can96] Ran Canetti. More on BPP and the polynomial-time hierarchy. *Information Processing Letters*, 57(5):237–241, 1996.
- [CFL⁺23] Luis G Coelho, Tertuliano Franco, Lael V Lima, João PC de Paula, João V A Pimenta, Guilherme L F Silva, and Daniel Ungaretti. A central limit theorem for intransitive dice. *arXiv preprint arXiv:2310.17083*, 2023.
- [CH20] Elisabetta Cornacchia and Jan Hązła. Intransitive dice tournament is not quasirandom. *arXiv preprint arXiv:2011.10067*, 2020.
- [CHR24] Lijie Chen, Shuichi Hirahara, and Hanlin Ren. Symmetric exponential time requires near-maximum circuit size. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1990–1999, 2024.
- [Coo71] Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC ’71, page 151–158. Association for Computing Machinery, 1971.
- [DC85] Marquis De Condorcet. *Essai sur l’application de l’analyse à la probabilité des décisions rendues à la pluralité des voix*. Imprimerie Royale, 1785. Facsimile published in 1972 by Chelsea Publishing Company, New York.
- [EM64] Paul Erdős and Leo Moser. On the representation of directed graphs as unions of orderings. *Math. Inst. Hung. Acad. Sci.*, 9:125–132, 1964.
- [Gar70] Martin Gardner. Paradox of nontransitive dice and elusive principle of indifference. *Scientific American*, 223(6):110, 1970.
- [Gär75] Peter Gärdenfors. Match making: assignments based on bilateral preferences. *Behavioral Science*, 20(3):166–173, 1975.
- [GS88] Joachim Grollmann and Alan L Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988.

- [HB76] Juris Hartmanis and Leonard Berman. On isomorphisms and density of NP and other complete sets. In *Proceedings of the eighth annual ACM symposium on Theory of computing*, pages 30–40, 1976.
- [HW19] Artem Hulko and Mark Whitmeyer. A game of nontransitive dice. *Mathematics Magazine*, 92(5):368–373, 2019.
- [Jur98] Marcin Jurdziński. Deciding the winner in parity games is in $UP \cap co-UP$. *Information Processing Letters*, 68(3):119–124, 1998.
- [KKMP21] Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos H Papadimitriou. Total functions in the polynomial hierarchy. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, pages 44–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2021.
- [Kor25] Oliver Korten. Range avoidance and the complexity of explicit constructions. *Bulletin of EATCS*, 145(1), 2025.
- [Kre86] Mark W Krentel. The complexity of optimization problems. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 69–76, 1986.
- [Li24] Zeyong Li. Symmetric exponential time requires near-maximum circuit size: Simplified, truly uniform. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 2000–2007, 2024.
- [LR94] Klaus-Jörn Lange and Peter Rossmanith. Unambiguous polynomial hierarchies and exponential size. In *Proceedings of IEEE 9th Annual Conference on Structure in Complexity Theory*, pages 106–115. IEEE, 1994.
- [NR98] Rolf Niedermeier and Peter Rossmanith. Unambiguous computations and locally definable acceptance types. *Theoretical Computer Science*, 194(1-2):137–161, 1998.
- [Pap84] Christos H Papadimitriou. On the complexity of unique solutions. *Journal of the ACM (JACM)*, 31(2):392–400, 1984.
- [Pap94] Christos H Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [PY82] Christos H Papadimitriou and Mihalis Yannakakis. The complexity of facets (and some facets of complexity). In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 255–260, 1982.
- [RMHH04] Kenneth B Reid, Alice A McRae, Sandra Mitchell Hedetniemi, and Stephen T Hedetniemi. Domination and irredundance in tournaments. *Australasian journal of combinatorics*, 29:157–172, 2004.
- [RS98] Alexander Russell and Ravi Sundaram. Symmetric alternation captures BPP. *computational complexity*, 7(2):152–162, 1998.
- [Sch17] Alex Schaefer. Balanced non-transitive dice ii: tournaments. *arXiv preprint arXiv:1706.08986*, 2017.
- [Sim77] Janos Simon. On the difference between one and many: preliminary version. In *International Colloquium on Automata, Languages, and Programming*, pages 480–491. Springer, 1977.

- [Ste59] Richard Stearns. The voting problem. *The American Mathematical Monthly*, 66(9):761–763, 1959.
- [Sto76] Larry J Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.
- [Val74] Leslie G Valiant. A reduction from satisfiability to hamiltonian circuits that preserves the number of solutions. *Manuscript, Leeds*, 1974.
- [Val76] Leslie G Valiant. Relative complexity of checking and evaluating. *Information processing letters*, 5(1):20–23, 1976.
- [VV85] Leslie G Valiant and Vijay V Vazirani. NP is as easy as detecting unique solutions. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, page 458–463, New York, NY, USA, 1985. Association for Computing Machinery.
- [Wra76] Celia Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):23–33, 1976.