Game Theoretic Notions of Fairness in Multi-Party Coin Toss[*]

Kai-Min Chung[2], Yue Guo[1], Wei-Kai Lin[1], Rafael Pass[1], and Elaine Shi[1]

[1]Cornell/CornellTech, {yueguo,wklin,rafael,elaine}@cs.cornell.edu
[2]Academia Sinica, kmchung@iis.sinica.edu.tw

Abstract

Coin toss has been extensively studied in the cryptography literature, and the well-accepted notion of fairness (henceforth called *strong fairness*) requires that a corrupt coalition cannot cause non-negligible bias. It is well-understood that two-party coin toss is impossible if one of the parties can prematurely abort; further, this impossibility generalizes to multiple parties with a corrupt majority (even if the adversary is computationally bounded and fail-stop only).

Interestingly, the original proposal of (two-party) coin toss protocols by Blum in fact considered a weaker notion of fairness: imagine that the (randomized) transcript of the coin toss protocol defines a winner among the two parties. Now Blum's notion requires that a corrupt party cannot bias the outcome in its favor (but self-sacrificing bias is allowed). Blum showed that this weak notion is indeed attainable for two parties assuming the existence of one-way functions.

In this paper, we ask a very natural question which, surprisingly, has been overlooked by the cryptography literature: can we achieve Blum's weak fairness notion in *multi-party* coin toss? What is particularly interesting is whether this relaxation allows us to circumvent the corrupt majority impossibility that pertains to strong fairness. Even more surprisingly, in answering this question, we realize that it is not even understood how to define weak fairness for multi-party coin toss. We propose several natural notions drawing inspirations from game theory, all of which equate to Blum's notion for the special case of two parties. We show, however, that for multiple parties, these notions vary in strength and lead to different feasibility and infeasibility results.

[*]An extended abstract of this work appeared in TCC 2018.

Contents

1 Introduction

The study of coin toss protocols was initiated in Blum's ground-breaking work [16]. Consider the following scenario: Alice and Bob had concurrent and independent results that solved a difficult open question in cryptography. Both submitted their papers to the prestigious Theory of Cryptography Conference (TCC) 2018 conference with the most amazing program committee (PC). The wise PC urged Alice and Bob to merge their results into one paper and provided them with a single presentation slot at the conference. Now Alice and Bob would like to toss a random coin to decide who goes to the most fabulous conference venue ever, Goa, and present the paper. Since Alice and Bob are not in the same room, they would like to complete the coin toss by sending messages to each other (slowly) over the Ethereum blockchain, such that anyone who observes the transcript can determine the outcome of the coin flip. Now either party would like to make sure that he/she has a fair chance of winning even when the other cheats and deviates from the protocol. The academic literature has since referred to Blum's notion of fairness as *weak fairness*; and Blum showed that assuming the existence of one-way functions a weakly-fair, 2-party coin toss protocol can be constructed [16]. Interestingly, however, the vast majority of subsequent cryptography literature has focused on a stronger notion of fairness than Blum's, that is, a corrupt party cannot bias the outcome of the coin toss — henceforth we refer to this notion as *strong fairness* [18]. It is not difficult to see that a strongly fair coin toss protocol must also be weakly fair; but not the other way round. In particular, a weakly fair protocol allows a corrupt party to bias the outcome of the remaining honest party — but the bias must not be in the corrupt party's favor. Unfortunately for the strongly fair notion, Cleve's celebrated result [18] proved its impossibility in a 2-party setting even for computationally bounded, fail-stop adversaries.

In this paper, we consider multi-party extensions of Blum's notion of weak fairness. We ask a very natural question that seems to have been overlooked by the literature so far:

Can we achieve Blum's weak fairness notion in multi-party coin toss protocols?

By contrast, the strong fairness notion has been extensively studied in the multi-party context [12, 27]. Well-known results tell us that the strong notion is attainable assuming honest majority and existence of one-way functions. On the other hand, Cleve's 2-party impossibility extends to multiple parties with a corrupt majority [18]. Therefore, a more refined question is

Can we overcome Cleve's impossibility for corrupt majority multi-party coin toss with weak fairness?

Of course, to answer the above questions, we must first answer

How do we even define weak fairness in multi-party coin toss protocols?

Intriguingly, even the definition itself is non-trivial! In this paper, we propose several natural notions of fairness that are inspired by the line of work on game theory [15,35,36,48]. Interestingly, all of these notions equate to Blum's notion for the special case of 2 parties; however, in general, they differ in strengths for multiple parties and thus lead to differing feasibility and infeasibility results.

1.1 Our Results and Contributions

Consider the following scenario: n parties would like to play a 1-bit roulette game over the Internet: First, each party puts down 1 Ether as stake and places a *publicly visible*[1] bet (also referred to as

[1]Unless otherwise noted, we consider public preference profiles. For completeness, however, we present results for private preference profiles in the appendices, Section 7.

the party's *preference*) on one of the bits $b \in \{0,1\}$. Without loss of generality *we assume that not everyone bets on the same bit.* Next, they run an n-party coin toss protocol by exchanging messages over the Ethereum blockchain, and transcript of the protocol determines an outcome bit. Now, those who betted correctly are called winners; and those who betted wrongly are called losers. Finally, every loser loses its stake to the house (e.g., owner of the smart contract); and each winner gets paid 1 Ether by the house. We require that *in an honest execution, each bit is chosen with probability* $1/2$. Henceforth in the paper for simplicity we shall think of the Ethereum blockchain as a broadcast medium with identifiable abort, i.e., a public bulletin board that allows parties to post messages.

How should we define fairness for this 1-bit roulette game? Cryptography and game theory provide different answers. The standard notion from cryptography is again strong fairness [18], that is, any corrupt coalition should not be able to bias the outcome by more than a negligible amount. As mentioned strong fairness is unattainable under a corrupt majority even for fail-stop adversaries [18]. Most of game theory, on the other hand, considers (computational) Nash Equilibrium [48], that is, no corrupt *individual* can noticeably improve its expected reward by deviating, assuming that everyone else is playing honestly. Although Nash Equilibrium is indeed attainable by adopting a standard, strongly fair multi-party coin toss protocol that tolerates deviation by any single party [27], such a notion might be too weak. In particular, no guarantee is provided when two or more parties collude (e.g., in cryptocurrency applications, an individual user can always make up any number of pseudonyms and control the majority in a game). Therefore we would like to explore notions in between that allow us to resist majority coalitions and provide meaningful fairness guarantees in practical applications. In this paper, we define several notions of fairness — all of them equate to Blum's notion [16] for the special case of 2 parties. Thus for all of our notions, in the 2-party case Blum's result applies: assuming one-way functions, all notions are attainable against malicious, computationally bounded adversaries that control one of the two parties; moreover, for fail-stop adversaries, all our notions are attainable against even unbounded adversaries that control one of the two parties.

Henceforth for our fairness notions, we are concerned about feasibility and infeasibility results for 3 or more parties, and particularly for the case of *corrupt majority* (since for honest majority, feasibility is known even for strong fairness, against malicious, computationally bounded adversaries, due to the celebrated result by Goldreich et al. [27]). As a final remark before we introduce our notions, all our notions (as well as Blum's notion) can be easily ruled out for computationally unbounded, malicious adversaries [33, 44].

1.2 Maximin Fairness

Definition. A natural notion, which seems to be a good fit for cryptocurrency applications, is to require the following: an honest Alice should not be harmed even when everyone else is colluding against her. In other words, any individual's expected reward should not noticeably decrease (relative to an all-honest execution) even when all others are colluding against her. This notion has a game theoretic interpretation: the honest strategy maximizes a player's worst-case expected payoff (even when everyone else is colluding against her); and moreover, by playing honest, the player's worst-case expected payoff is not noticeably worse than an all-honest execution. For maximin fairness, we present a complete characterization of feasibilities/infeasibilties.

Feasibility and infeasibility for almost unanimous preference profiles. For 3 or more parties, if everyone agrees in preference except one party, we say that the preference profile is *almost unanimous.* When the preference profile is almost unanimous, maximin fairness is possible for fail-stop adversaries and without relying on cryptographic assumptions. Recall that a fail-stop adversary may prematurely abort from the protocol but would otherwise follow the honest protocol [18]. The corresponding protocol is very simple: without loss of generality assume that one party prefers 0 (called the 0-supporter) and all others prefer 1 (called the 1-supporters). Now, the 0-supporter chooses a random bit and broadcasts it. If the broadcast indeed happens, the bit broadcast is declared as the outcome. Otherwise, the outcome is defined to be 1.

We then prove that for an almost unanimous preference profile, maximin fairness is impossible for malicious adversaries even when allowing cryptographic assumptions. This result is somewhat counter-intuitive in light of the earlier feasibility for fail-stop (and the proof rather non-trivial too). In particular, in most of the cryptography literature, we are familiar with techniques that compile fail-stop (or semi-honest) protocols to attain full, malicious security [12,27] — but these compilation techniques *do not preserve maximin fairness* and thus are inapplicable here.

Note that for the special case of 3 parties, unless everyone has the same preference any preference profile is almost unanimous — thus for the case of 3 parties we already have a complete characterization. For 4 or more parties, we need to consider the case when the preference profile is more divided.

Infeasibility for amply divided preference profiles. If there are at least two 0-supporters and at least two 1-supporters, we say that the parties have an *amply divided* preference profile. Note that for 3 or more parties, unless everyone has the same preference, then every preference profile is either almost unanimous or amply divided. For an amply divided preference profile, we show infeasibility even against computationally bounded, fail-stop adversaries by reduction to Cleve's impossibility result for strong fairness [18].

We summarize our results for maximin fairness in the following theorems — although not explicitly noted, all theorems are concerned about an adversary that may control up to $n-1$ players.

Theorem 1 (Maximin fairness: upper bound (informal)). *For any $n \geq 3$ and any almost unanimous preference profile, there is an n-party coin toss protocol that achieves maximin fairness against fail-stop and computationally unbounded adversaries.*

Theorem 2 (Maximin fairness: lower bound (informal)). *For any $n \geq 3$ and any almost unanimous preference profile, no n-party coin toss protocol can achieve maximin fairness against malicious and even polynomially bounded adversaries. Further, for any $n \geq 4$ and any amply divided preference profile, no n-party coin toss protocol can achieve maximin fairness against fail-stop and even polynomially bounded adversaries.*

Summary. While maximin fairness appears to provide strong guarantees in cryptocurrency and smart contract applications, we showed rather broad infeasibility results. Nonetheless it gives us a glimpse of hope: for the case of almost unanimous preference profiles and fail-stop adversaries, we are able to achieve positive results for corrupt majority while strong fairness cannot! We thus continue to explore alternative notions in hope of finding one that leads to broader feasibility results. Our high-level idea is the following: earlier, maximin fairness aims to rule out coalitions that *harm honest parties*; instead we now consider notions that rule out coalitions capable of *improving its own wealth* — this gives rise to two new notions, cooperative-strategy-proof fairness and Strong Nash Equilibrium, as we discuss subsequently in Sections 1.3 and 1.4.

1.3 Cooperative-Strategy-Proof Fairness

Definition. Cooperative-strategy-proof (CSP) fairness requires that no deviation by a corrupt coalition of size up to $n-1$ can noticeably improve the coalition's total expected reward relative to an honest execution. It is not difficult to see that CSP fairness is equivalent to maximin fairness for zero-sum cases: when exactly half prefer 0 and half prefer 1. However, the two notions are incomparable in general.

Feasibility for almost unanimous preference profiles. When almost everyone prefers the same bit except for one party, we show that the following simple protocol achieves CSP fairness against malicious adversaries. For simplicity, our description below assumes an ideal commitment functionality $\mathcal{F}_{\text{idealcomm}}$ — but this idealized oracle can be replaced with suitable non-malleable concurrent commitment schemes [42, 43] with some additional work. Without loss of generality we assume that a single party prefers 0 and everyone else prefers 1: First, everyone picks a random bit upfront and commits the bit to $\mathcal{F}_{\text{idealcomm}}$. In round 0, the single 0-supporter opens its committed bit and broadcasts it. In round 1, everyone else opens its committed bit and broadcasts the opening. The outcome is defined to be 0 if one or more 1-supporter(s) aborted; else it is defined to be the XOR of all bits that have been correctly opened.

Finally, for fail-stop adversaries, a variant of the above protocol without commitment can achieve CSP fairness against even unbounded adversaries.

Infeasibility for amply divided preference profiles. For any amply divided preference profile, we prove that it is impossible to achieve CSP fairness against even fail-stop, polynomially bounded adversaries.

We summarize results for CSP fairness in the following theorem.

Theorem 3 (CSP fairness (informal))**.** *For any almost unanimous preference profile, it is possible to attain CSP fairness against fail-stop, unbounded adversaries, and against malicious, polynomially bounded adversaries assuming one-way permutations. By contrast, for any amply divided preference profile, it is impossible to attain CSP fairness against even fail-stop, polynomially bounded adversaries.*

1.4 Strong Nash Equilibrium

Due to earlier impossibility results for maximin fairness and CSP fairness, we ask if there is a notion for which we can enjoy broad feasibility. To this end we consider a fairness notion inspired by Strong Nash Equilibrium (SNE) [35], henceforth referred to as SNE fairness. SNE fairness requires that no deviation by a coalition can improve every coalition member's expected reward. It is not difficult to see that for SNE fairness, we only need to resist *unanimous* coalitions, i.e., coalitions in which every member prefers the same bit. Further, SNE fairness is also strictly weaker than CSP fairness in general.

We show that a simple dueling protocol achieves SNE fairness against malicious (but polynomially bounded) adversaries: pick two parties with opposing preferences (i.e., pick the two with smallest possible party identifiers), and then have the two run Blum's weak coin toss protocol. Further, the computational assumptions can be removed for fail-stop adversaries and thus SNE fairness can be guaranteed unconditionally for the fail-stop case. We summarize our results on SNE fairness in the following theorem.

Theorem 4 (SNE fairness (informal))**.** *For any $n \geq 3$ and any preference profile: 1) there is an n-party coin toss protocol that achieves SNE fairness against malicious, polynomially-bounded* *adversaries assuming the existence of one-way permutations; and 2) there is an n-party coin toss protocol that achieves SNE fairness against fail-stop, unbounded adversaries.*

Alternative formulation: cooperative-coalition-proof fairness. While SNE fairness aims to rule out coalitions that improve every coalition member's wealth, an alternative notion would be to resist *self-enforcing* coalitions that aim to improve the coalition's overall wealth. In particular, a coalition is said to be self-enforcing iff no *self-enforcing* sub-coalition can gain by deviating from the coalition's original strategy. Such coalitions are stable and will not implode due to internally misaligned incentives. We formalize this notion in Section C, which we call *cooperative-coalition-proof fairness* (CCP fairness). Since CCP fairness considers complex coalition and sub-coalition behavior, we can no longer use the familiar protocol execution model used in the standard cryptography literature — we instead propose a new, suitable protocol execution model that allows us to characterize complex coalition structures. Our CCP fairness notion is inspired by the notion of coalition-proof Nash equilibrium (CPNE) [15] in game theory — but unlike CPNE which considers self-enforcing coalitions that seek to improve every member's gain, our CCP notion considers self-enforcing coalitions that seek to improve its overall gain, and thus our notion is stronger (i.e., demands stronger solution concepts).

Although for general games, SNE fairness and CCP fairness are incomparable, we prove that for the special case of multi-party coin toss, the two notions are in fact equivalent! In this context both notions effectively rule out *unanimous* coalitions where everyone prefers the same outcome.

1.5 Technical Highlight

Conceptual, definitional contributions. First, we make a conceptual contribution by introducing several natural, game-theoretical notions of fairness for multi-party coin toss — our work thus opens a new avenue for connecting game theory and cryptography. Earlier efforts at connecting game theory and multi-party computation typically model the correctness and/or confidentiality of multi-party protocols as a game (see Section 8 for more discussions), whereas we consider a model in which each party independently declares the utility for various outcomes.

A new framework for proving lower bounds. Our upper bounds are simple and intuitive in hindsight (but note that several upper bounds were not immediately obvious to us in the beginning). Our main lower bound results, however, are rather non-trivial to prove. The most non-trivial proofs are 1) the impossibility of maximin fairness for almost unanimous preference profiles, against malicious, computationally bounded adversaries; and 2) the impossibility of CSP fairness for amply divided preference profiles, this time against fail-stop and computationally bounded adversaries.

We develop a new proof framework and apply this framework to rule out both maximin fairness (for almost unanimous, malicious) and CSP fairness (for amply divided, fail-stop)[2]. In this proof framework, we would carefully group nodes into three partitions such that we can view the execution as a 3-party protocol (between the partitions). In both impossibility proofs, we show that the requirements of maximin or CSP fairness imposes a set of conditions that are by nature self-contradictory and thus cannot co-exist.

Since the lower bound proofs are highly non-trivial, to help the reader we give an informal narrative of the maximin proof in Section 4.4. Then, in Section 5.3, we intuitively explain the additional challenges that arise for ruling out CSP fairness (for amply divided, fail-stop) — this proof is even more challenging than maximin fairness (for almost unanimous, malicious) partly

[2]Interestingly, later in Section 7, we again reuse the same proof framework to prove lower bounds for private-preference protocols too.

because we need to rule out even fail-stop adversaries in this case. The full formal proofs are deferred to the appendices due to lack of space.

2 Preliminaries

2.1 Protocol Execution Model

A protocol is a system of Interactive Turing Machines (ITMs) where each ITM is also referred to as a *party* or a *player*. Each party is either *honest* or *corrupt*. Honest parties correctly follow the protocol to the end without aborting. Corrupt parties, on the other hand, are controlled by an adversary $\mathcal{A}$. Corrupt parties forward all received messages to $\mathcal{A}$ and send messages or abort based on $\mathcal{A}$'s instructions. In this way, we can view the set of all corrupt parties as a single coalition that collude with one another.

A protocol's execution is parametrized by a security parameter $\kappa \in N$ that is public known to all parties including the adversary $\mathcal{A}$. A protocol's execution may be randomized where all parties and the adversary $\mathcal{A}$ receive and consume a string of random bits.

We assume a *round-based* execution model. In each round, every honest party can perform any polynomial in κ amount of computation. At the end of the round, every party may broadcast a message whose length must be polynomial in κ as well. We assume a *synchronous broadcast* medium (with identifiable abort) for parties to communicate with each other. Messages sent by honest parties in round r will be delivered to all honest parties at the beginning of round $r+1$. If a party i aborts the protocol in round r without sending any message, then all honest parties can detect such abort by detecting the absence of i's message at the beginning of round $r+1$. As an example, one can imagine that parties communicate by posting messages to a public blockchain such as Bitcoin [25, 47, 50, 51][3].

2.2 Corruption Models

The adversary can corrupt any number of parties. Without loss of generality, we assume that for any fixed adversary algorithm $\mathcal{A}$, the set of parties it wants to corrupt is deterministically encoded in the description of $\mathcal{A}$ (i.e., for any fixed adversary $\mathcal{A}$, there is no randomness in the choice of the corrupt coalition). We assume that the adversary is capable of a *rushing* attack[4], i.e., in any round r, the adversary is allowed to view messages sent by honest parties in round r, before deciding what messages corrupt parties will send in round r.

Depending on the adversary's capability, we say that the adversary is fail-stop or malicious. More formally, let Π denote the honest protocol under consideration. An adversarial algorithm $\mathcal{A}$ is said to be *fail-stop* or *malicious* w.r.t. Π iff the following holds:

- *Fail-stop:* Corrupt nodes always follow the honest protocol but may abort in the middle of the protocol. The decision to abort (or not) can depend on the corrupt parties' view in the protocol so far.
- *Malicious:* The adversary can make corrupt parties deviate arbitrarily from the prescribed protocol, including sending arbitrary messages, choosing randomness arbitrarily, and aborting prematurely.

[3]Although a blockchain typically requires honest majority assumptions to retain security, the parties involved in the coin-toss protocol can be majority corrupt.

[4]We note that in a simultaneous message model where the adversary is not capable of rushing attacks, even the standard notion of (strong) fairness [18] (which is stronger than all notions considered in this paper) is trivial to achieve for 2-party or multi-party coin toss, even against any majority corrupt coalition.

2.3 Additional Notations and Assumptions

Throughout the paper, we assume that the number of parties is polynomially bounded, i.e., $n = \mathsf{poly}(\kappa)$ for some polynomial function $\mathsf{poly}(\cdot)$. We consider protocols that terminate in polynomially many rounds. Specifically, there exists some polynomial $R(\cdot)$ that denotes the round complexity of the protocol, such that with probability 1, honest parties complete execution in $R(\kappa)$ even in the presence of any (possibly computationally unbounded) adversary controlling any corrupt coalition.

We say that a function $\nu(\cdot)$ is a *negligible* function iff for every polynomial function $p(\cdot)$, there exists some $\kappa_0 \in \mathbb{N}$ such that $\nu(\kappa) \leq 1/p(\kappa_0)$ for all $\kappa \geq \kappa_0$.

3 Definitions: Multi-Party Coin Toss

As in the standard cryptography literature, we model protocol execution as a system of Interactive Turing Machines. We consider a synchronous model with a broadcast medium. Messages broadcast by honest parties in the current round are guaranteed to be delivered at the beginning of the next round. We assume *identifiable abort*, that is, failure to send a message is publicly detectable.

We assume that the adversary, denoted $\mathcal{A}$, can control any number of parties. Without loss of generality, we assume that the set of parties $\mathcal{A}$ wants to corrupt is hard-wired in the description of $\mathcal{A}$. We assume a simultaneous messaging model with the possibility of *rushing attacks*, that is, the adversary can observe honest nodes' messages before deciding corrupt nodes' actions (including what messages to send and whether to abort) in any round.

Recall that a *fail-stop* party is one that could abort prematurely but would otherwise follow the honest protocol. By contrast, a *malicious* party is one that can deviate arbitrarily from the honest protocol.

3.1 Multi-Party Coin Toss

Preference profile. Suppose that each party starts with a *preference* among the two outcomes 0 and 1. The vector of all parties' preferences, denoted $\mathcal{P} := \{0, 1\}^n$, is referred to as a preference profile. We sometimes refer to a party that prefers 1 as a 1-*supporter* and we refer to one that prefers 0 as a 0-*supporter*. In a preference profile $\mathcal{P} := \{0, 1\}^n$, if the number of 0-supporters and the number of 1-supporters are the same, we say that $\mathcal{P}$ is *balanced*; else we say that it is *unbalanced*.

Unless otherwise noted, we assume that all parties' preferences are predetermined and *public*. We discuss the private-preference case in the appendices, Section 7.

Coin-toss protocol. Consider a protocol Π where n parties jointly decide an outcome between 0 and 1. Such a protocol Π is said to be a coin toss protocol, there is a polynomial-time computable deterministic function, which, given the transcript of the protocol execution, outputs a bit $b \in \{0, 1\}$, often said to be the *outcome* of the protocol. For correctness, we require that an honest execution outputs each bit with probability exactly $\frac{1}{2}$ unless all parties have the same preference. More formally, correctness requires that

1. If some parties have differing preferences, in an all-honest execution (when all parties are honest), the probability that the outcome is 0 (or 1) is exactly[5] 1/2.
2. If all parties happen to prefer the same bit $b \in \{0, 1\}$, the honest execution should output the preferred bit b with probability 1.

[5]Our upper bounds achieve perfect correctness, but our lower bounds in fact extend easily even when allowing negligible correctness failure.

Payoff function. If the protocol's outcome is b, a party who prefers b receive a reward (or payoff) of 1; else it receives a reward (or payoff) of 0. Note that earlier in Section 1, our 1-bit roulette example had a -1 utility (rather than 0) for losing, but the two definitions are in fact equivalent; and for simplicity the remainder of the paper will assume 0 utility for losing.

3.2 Discussions

Trivial case: unanimous preference profile. When everyone has the same preference, we say that the preference profile is *unanimous*; otherwise we say that it is *divided*. In this case, we do not require that an honest execution produce an unbiased coin, since it makes sense for the outcome to be the bit that is globally preferred. In the remainder of the paper, for the case of public preference: if everyone prefers the same bit $b \in \{0, 1\}$, we assume that *the protocol simply fixes the outcome to be the universally preferred bit* b regardless of how parties act. In this way, everyone obtains a payoff of 1, and no deviation from the protocol can influence the outcome — therefore all game-theoretic fairness notions we consider are trivially satisfied when the preference profile is unanimous.

On public verifiability. Our definition implies public verifiability of the coin toss's outcome. Anyone who can observe messages sent over the broadcast medium (e.g., a public blockchain) can independently compute the outcome of the protocol. Note that under this definition, the outcome of the protocol is well-defined even when all parties are corrupt. Alternatively, we can define a weaker notion where we do not require such public verifiability — instead we require that honest parties output a bit at the end of the execution, and that they output the same bit (said to be the *outcome* of the execution) with probability 1 even in the presence of an arbitrary (possibly unbounded) adversary that corrupts up to $n - 1$ parties. Under this weaker notion, the outcome of an execution is not well-defined when all parties are corrupt. We note that all lower bounds in this paper in fact apply to this weaker notion too (which makes the lower bounds stronger).

3.3 Strong Fairness

We quickly review the classical notion of strong fairness [18]. Roughly speaking, strong fairness requires that the outcome of the coin toss protocol be unbiased even in the presence of an adversary (assuming that parties have divided preferences). In the definition of strong fairness, we consider a single adversarial coalition that corrupts up to $n - 1$ parties.

Definition 1 (Strong fairness [18])**.** Let $\mathfrak{A}$ a family of adversaries that corrupt at most $n - 1$ parties. An n-party coin toss protocol is said to be *strongly fair* against the family $\mathfrak{A}$, iff for every adversary $\mathcal{A} \in \mathfrak{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that (as long as not all parties have the same preference) the probability that the outcome is 1 is within $[\frac{1}{2} - \mathsf{negl}(\kappa), \frac{1}{2} + \mathsf{negl}(\kappa)]$ when playing with $\mathcal{A}$.

4 Maximin Fairness: Feasibilities and Infeasibilities

4.1 Definition of Maximin Fairness

Maximin fairness requires that no honest party should be harmed by any corrupt coalition. In other words, a corrupt coalition should not be able to (non-negligibly) decrease the expected payoff for any honest party relative to an all-honest execution. In maximin fairness, we consider a single adversarial coalition that controls up to $n - 1$ parties.

Definition 2 (Maximin fairness)**.** Let $\mathfrak{A}$ be a family of adversaries that corrupt up to $n-1$ parties; and let $\mathcal{P} \in \{0,1\}^n$ denote any divided preference profile. We say that an n-party coin toss protocol is maximin fair for $\mathcal{P}$ against the family $\mathfrak{A}$, iff for every adversary $\mathcal{A} \in \mathfrak{A}$, there exists some negligible function $\mathsf{negl}(\cdot)$ such that in an execution with the preference profile $\mathcal{P}$ and the adversary $\mathcal{A}$, the expected reward for any honest party is at least $\frac{1}{2} - \mathsf{negl}(\kappa)$. More specifically, we have the following special cases:

- *Computational maximin fairness.* If $\mathfrak{A}$ is the family of all non-uniform, probabilistic polynomial-time (henceforth denoted p.p.t.) fail-stop (or malicious resp.) adversaries that can corrupt as many as $n-1$ parties, we say that the protocol is *computationally* maximin fair for $\mathcal{P}$ against fail-stop (or malicious resp.) adversaries.
- *Statistical maximin fairness.* If $\mathfrak{A}$ is the family of all fail-stop (or malicious resp.) adversaries (including even computationally unbounded ones) that can corrupt as many as $n-1$ parties, we say that the protocol is *statistically* maximin fair for $\mathcal{P}$ against fail-stop (or malicious resp.) adversaries.
- *Perfect maximin fairness.* If a protocol is statistically maximin fair against fail-stop (or malicious resp.) adversaries, and moreover the above definition is satisfied with a choice of 0 for the negligible function, we say that the protocol is *perfectly* maximin fair for $\mathcal{P}$ against fail-stop (or malicious resp.) adversaries. A perfectly maximin fair protocol does not allow any single honest party to have even negligibly small loss in its expected payoff in comparison with an all-honest execution.

A straightforward observation is that classical strong fairness (Definition 1) implies maximin fairness:

Fact 1. *If an n-party coin toss protocol Π is strongly fair against a family of adversaries $\mathcal{F}$, then Π is maximin fair against $\mathcal{F}$ for any divided preference profile $\mathcal{P} \in \{0,1\}^n$.*

Sometimes we also say that a protocol is computationally (or statistically, perfectly resp.) maximin fair for $\mathcal{P}$ against any fail-stop (or malicious resp.) coalition of size K — and this means the most obvious where in the above definitions, the family of adversaries $\mathfrak{A}$ we consider is additionally restricted to corrupting exactly K parties.

Claim 1. *Let $\mathcal{P} \in \{0,1\}^n$ be any divided preference profile. An n-party coin toss protocol Π satisfies computational (or statistical, perfect resp.) maximin fairness for $\mathcal{P}$ against any fail-stop (or malicious resp.) coalition, iff Π satisfies computational (or statistical, perfect resp.) maximin fairness for $\mathcal{P}$ against any fail-stop (or malicious resp.) coalition of size exactly $n-1$.*

Proof. Deferred to Appendix A.2. $\square$

Game theoretic interpretation. If a coin-toss protocol is maximin fair, then the following hold:

1. First, the honest strategy maximizes a player's worst-case expected payoff (even when everyone else is colluding against the player); this explains the name "maximin fairness".
2. Moreover, when playing the honest strategy, a player's worst-case payoff is what it would have gained in an all-honest execution — note that a player's worst-case (expected) payoff obviously cannot be more than its payoff in an all-honest execution.

Equivalence to group maximin fairness. An alternative way to define "no-harm to honest parties" is to require that any corrupt coalition cannot decrease (by more than a negligible amount) the expected overall wealth (i.e., total payoff) of the honest parties. We prove that this notion, called group maximin fairness, is in fact equivalent to maximin fairness in the context of coin toss. We defer the formal definition and proofs to Appendix A.3.

4.2 The Case of Amply Divided Preference Profiles

As mentioned, feasibility for 2 parties or multiple parties but honest majority are already implied by existing literature [16, 27]. Henceforth we focus on the case of three or more parties and corrupt majority.

First, we consider amply divided preference profiles, where at least two people prefer 0 and at least two prefer 1 respectively (and hence there must be at least 4 people). It is not too difficult to rule out maximin fairness for amply divided preference profiles, even against fail-stop, computationally bounded adversaries, leading to the following theorem.

Theorem 5 (Maximin fairness: amply divided preference profiles)**.** *For any $n \geq 4$ and for any amply divided preference profile $\mathcal{P} \in \{0,1\}^n$, no n-party coin toss protocol can achieve even computational maximin fairness for $\mathcal{P}$ against even fail-stop adversaries.*

Proof. (sketch.) We show that if there is a maximin fair protocol for any amply divided preference profile, we can construct a 2-party strongly fair coin toss protocol (and thus violating Cleve's lower bound [18]). The proof follows from a standard partitioning argument: consider two partitions, each containing at least one 0-supporter and at least one 1-supporter. Now, we can view the protocol as a two-party protocol between the two partitions, and by maximin fairness, if either partition aborts, it must not create any non-negligible bias towards either direction. We defer the full proof to Appendix A.5. □

4.3 The Case for Almost Unanimous Preference Profiles

Possibility of perfect maximin fairness for fail-stop adversaries. First, we show that for fail-stop adversaries, we can achieve perfect maximin-fairness for almost unanimous preference profiles. Without loss of generality, assume that a single party prefers 0 and everyone else prefers 1. The following simple protocol can guarantee perfect maximin fairness:

1. In the first round, the single 0-supporter flips a random coin b and broadcasts b;
2. If the single 0-supporter successfully broadcast a message b, then the outcome is b; else the outcome is 1.

It is not difficult to see that this simple protocol satisfies perfect maximin fairness against fail-stop adversaries: all the 1-supporters do not take any actions and they do not influence the outcome of the protocol. For the single 0-supporter, if it deviates (by aborting), then the outcome will be 1 with probability 1, and all honest parties utility are guaranteed to be 1. Thus we derive the following theorem:

Theorem 6 (Possibility of perfect maximin fairness for almost unanimous preferences and fail-stop adversaries.)**.** *For any $n \geq 3$, any almost unanimous preference profile $\mathcal{P} \in \{0,1\}^n$, there exists an n-party coin toss protocol that achieves perfect maxmin fairness for $\mathcal{P}$ against fail-stop adversaries.*

Impossibility of computational maximin fairness for malicious adversaries. Next, we show that maximin fairness is impossible to achieve for almost unanimous preference profiles against malicious adversaries, even when allowing computational assumptions.

Theorem 7 (Impossibility of maximin fairness for almost unanimous preferences and malicious adversaries)**.** *For $n \geq 3$ and any almost unanimous preference profile $\mathcal{P} \in \{0,1\}^n$, no n-party coin-toss protocol Π can ensure computational maximin fairness for $\mathcal{P}$ against malicious adversaries.*

4.4 Informal Proof Roadmap for Theorem 7

We in fact prove a stronger lower bound than stated in Theorem 7: we show that maximin fairness is impossible for any almost unanimous preference profile (for 3 or more parties), even against *semi-malicious*, polynomially bounded adversaries. In particular, a semi-malicious adversary can 1) choose corrupt parties' random coins arbitrarily upfront, and 2) prematurely abort; but otherwise it follows the honest protocol.

For simplicity, we focus on the case of 3 parties but the proof generalizes directly to more parties. Suppose that the 3 parties are called P_1, P_2 and P_3, and they come with the preferences $1, 0$, and 1 respectively.

We now present an informal proof roadmap, deferring the formal proof to Appendix A.4. We begin by assuming that a maximin fair protocol exists for 3 parties, resisting semi-malicious, computationally bounded adversaries. Our proof will seek to reach a contradiction, effectively showing that the various conditions imposed by maximin fairness cannot co-exist.

4.4.1 Almost All Random Coins of a Lone Semi-Malicious 0-Supporter are Created Equal

By a direct application of maximin fairness, if the single 0-supporter is semi-malicious and allowed to program his random coins, then he should not bias the remaining two parties towards 0. However, perhaps somewhat surprisingly at first sight, we can prove a result that is much stronger, that the single 0-supporter in fact (almost) cannot cause bias towards *either* direction by programming its random coins!

Henceforth, we shall use the notations T_1, T_2, T_3 to denote the three parties' random coins, where T_2 belongs to the single 0-supporter P_2. Consider an honest execution of the protocol conditioned on the fact that the single 0-supporter has its randomness fixed to T_2, and let $f(T_2)$ denote the expected outcome (where the probability is taken over P_1 and P_3's randomness). We prove the following lemma stating that (except for a negligible fraction of choices), all choices of T_2 are equal if P_2 is the lone semi-malicious party.

Lemma 1 (Almost all random coins of a lone semi-malicious P_2 are created equal)**.** *Suppose that the protocol under consideration satisfies computational maximin fairness against semi-malicious adversaries. Then, there exists a negligible function $\mathsf{negl}(\cdot)$ such that except for $\mathsf{negl}(\kappa)$ fraction of T_2's, it must be that $|f(T_2) - 0.5|$ is a negligible function in κ.*

Proof. (sketch.) We present a proof sketch: by maximin fairness, we know that for all T_2's, $f(T_2) \geq 0.5 - \mathsf{negl}(\kappa)$. Now, notice that $\mathbf{E}_{T_2} f(T_2) = \frac{1}{2}$ by honest execution, i.e., the expected value of $f(T_2)$ is $\frac{1}{2}$ when averaging over T_2. This means that if there is a non-negligible fraction of T_2's that cause non-negligible bias towards 1, then there must be a non-negligible fraction of T_2's that cause non-negligible bias towards 0 and the latter violates maximin fairness for a semi-malicious P_2. The formal proof is presented in Appendix A.4. □

4.4.2 The Lone-Wolf Condition and Wolf-Minion Conditions

Henceforth, our general plan is to show that if the above T_2-equality lemma holds, then the following two conditions, implied by the definition of maximin fairness, cannot co-exist.

- *Lone-wolf condition.* When P_1 (or P_3) is the only fail-stop party, it cannot cause non-negligible bias towards either direction. Such an attack is also called a lone-wolf attack.
- *Wolf-minion condition.* When P_1 and P_2 (or P_2 and P_3) form a fail-stop coalition, they cannot cause non-negligible bias towards 0. In fact we only care about attacks where P_2 is a silent accomplice (called a *minion* [24]) that never aborts but shares information with P_1 (or P_3); and P_1 (or P_3) may abort depending on its view in the execution (called a *wolf* [24]). Such attacks are called wolf-minion attacks.

Note that both conditions above consider only fail-stop adversaries, and in fact in the entire proof the only place we rely on a semi-malicious adversary is in the proof of the aforementioned T_2-equality lemma.

4.4.3 Non-Blackbox Application of Cleve's Lower Bound Conditioned on T_2

Recall that we assume that a maximin fair, 3-party protocol Π exists for the sake of reaching a contradiction. Now consider an execution of this protocol when P_2's randomness is fixed to T_2, and further, assume that P_2 never aborts and always follows the honest protocol to completion. We now view this 3-party protocol as a 2-party protocol between P_1 and P_3, where P_2's randomness T_2 is public and hard-wired in P_1 and P_3's program — more specifically both P_1 and P_3 would run the 3-party protocol Π, and they each independently simulate the actions of P_2 and compute all messages that P_2 wants to send.

Due to the T_2-equality lemma, if P_1 and P_3 are honest, we know that the expected outcome would be $\frac{1}{2}$ for almost all T_2's. Now, in this 2-party protocol defined by a fixed T_2 (that does not belong to the negligible fraction of bad T_2's), Cleve [18] showed that there must exist a polynomial-time attack by one of the parties, that causes non-negligible bias — but the bias can be either towards 0 or 1. Unfortunately, the direct implication of Cleve's lower bound is not quite so useful for us: it shows that a semi-malicious P_2 can collude with a fail-stop P_1 (or P_3) and cause bias for the remaining honest party, that is P_3 (or P_1) — but unless this bias is towards 0, it does not lend to a contradiction.

Our plan is the following: we will nonetheless apply Cleve's impossibility, but in a non-blackbox manner. First, we will show that for any fixed T_2 (except for a negligible fraction of bad ones), either P_1 or P_3 can bias towards 1 with an aborting attack. Specifically, we define a sequence of adversaries like in Cleve's proof, denoted $\{\mathcal{A}_i^b(1^\kappa, T_2), \mathcal{B}_i^b(1^\kappa, T_2)\}_{i\in[R]}, \cup\{\mathcal{A}_0(1^\kappa, T_2)\}$ where R is the protocol's round complexity. Adversaries $\mathcal{A}_i^b(1^\kappa, T_2)$, $\mathcal{B}_i^b(1^\kappa, T_2)$, and $\mathcal{A}_0(1^\kappa, T_2)$ are defined when P_2's randomness is fixed to T_2:

- Adversary $\mathcal{A}_i^b(1^\kappa, T_2)$:
  - $\mathcal{A}_i^b$ executes the honest protocol on behalf of P_1 and P_2 (whose randomness is fixed to T_2) until the moment right before P_1 is going to broadcast its i-th message.
  - At this moment, $\mathcal{A}_i^b$ computes α_i, that is, imagine that P_3 aborted right after sending its $(i-1)$-th message, what would be the outcome of parties P_1 and P_2.
  - If $\alpha_i = b$, then P_1 aborts after sending the i-th message; else P_1 aborts right now without sending the i-th message.

- Adversary $\mathcal{B}_i^b(1^\kappa, T_2)$: The definition is symmetric to that of $\mathcal{A}_i^b(1^\kappa, T_2)$ but now P_3 is the fail-stop party.
- Adversary $\mathcal{A}_0(1^\kappa, T_2)$: P_1 aborts upfront prior to speaking at all.

Cleve [18] showed that one of these above adversaries must be able to cause non-negligible bias towards either 0 or 1. However, due to the requirement of maximin fairness, we may conclude that the bias must be towards 1 except for a negligible fraction of the T_2's. Suppose this is not the case, i.e., the bias is towards 0 for a non-negligible fraction of the T_2's — then we could easily construct an attack (for the 3-party protocol) where a semi-malicious P_2 colluding with a fail-stop P_1 (or P_3) can bias the remaining party towards 0 — in fact, in our formal proof later, we show that such an attack is even possible with a fail-stop P_1 and a silent accomplice P_2 who just shares information with P_1 but would otherwise follow the protocol honestly (i.e., a wolf-minion attack). Proving this stronger statement would require a little more effort — but looking forward, later we would like to rule out CSP fairness for even fail-stop adversaries. There we have a similar agenda: 1) reprove the T_2-equality lemma but for fail-stop adversaries and CSP fairness, and 2) show that under the T_2-equality lemma, the lone-wolf condition and the wolf-minion condition cannot co-exist. Thus in our formal proof later we will actually rely on a wolf-minion (fail-stop) attack to rule out the 0-bias attack.

4.4.4 Averaging over T_2: A Wolf-Minion Attack with Benign Bias

Next, we consider the above adversaries but now averaging over T_2. In other words, let $\overline{\mathcal{A}}_i^b(1^\kappa)$ be the following attacker: choose a random T_2, consider the protocol execution with P_2's randomness fixed to T_2 and with the adversary $\mathcal{A}_i^b(1^\kappa, T_2)$. $\overline{\mathcal{B}}_i^b(1^\kappa)$ and $\overline{\mathcal{A}}_0(1^\kappa)$ are similarly defined by averaging over T_2.

Now, we prove that among these adversaries $\{\overline{\mathcal{A}}_i^b(1^\kappa), \overline{\mathcal{B}}_i^b(1^\kappa)\}_{i\in[R]}$ and $\overline{\mathcal{A}}_0(1^\kappa)$, one of them must be able to bias the remaining party, either P_1 or P_3, towards 1. This proof follows in a somewhat standard manner from an averaging argument and we defer the details to Appendix A.4. Note that reflecting in the 3-party protocol, this corresponds to a wolf-minion attack that creates benign bias: P_1 (or P_3) acts as a fail-stop wolf, and P_2 acts as a silent accomplice (i.e., the minion) that follows the honest protocol to completion but shares information with P_1 (or P_3). Although this wolf-minion is able to create bias, the bias is benign and does not violate the definition of maximin fairness. Thus to reach a contradiction, it still remains to show an attack that creates harmful bias.

4.4.5 Applying the Lone-Wolf Condition: A Wolf-Minion Attack with Harmful Bias

We now argue that if there is a wolf-minion attack that creates benign bias, there must be one that creates harmful bias, assuming that the lone-wolf condition holds. To show this, we consider the adversary that flips the decisions (to abort in the present or next round) of the benign wolf-minion attack. Without loss of generality, assume that $\overline{\mathcal{A}}_i^1$ is the successful wolf-minion attack that creates non-negligible bias towards 1. We now consider $\overline{\mathcal{A}}_i^0$ which flips $\overline{\mathcal{A}}_i^1$'s decision whether to abort in round i or $i+1$, and we argue that $\overline{\mathcal{A}}_i^0$ must create non-negligible bias towards 0. At a very high level, the proof will show that the lone-wolf condition acts like a balancing condition.

Let Q be the set of sample paths (defined by choices of T_1, T_2, and T_3) over which $\overline{\mathcal{A}}_i^1$ decides to abort in round i, and let $\overline{Q}$ be the remaining sample paths. Now, consider a hybrid adversary that takes $\overline{\mathcal{A}}_i^1$'s decisions on Q and takes $\overline{\mathcal{A}}_i^0$'s decisions on $\overline{Q}$: in other words, P_1 basically always aborts in round i! Due to the lone-wolf condition, whatever average bias towards 1 $\overline{\mathcal{A}}_i^1$ has on Q,

$\overline{\mathcal{A}}_i^0$ must create almost the same bias towards 0 on $\overline{Q}$. By a symmetric argument and considering a lone wolf P_1 that always aborts in round $i+1$, whatever average bias towards 1 $\overline{\mathcal{A}}_i^1$ has on $\overline{Q}$, $\overline{\mathcal{A}}_i^0$ must create almost the same bias towards 0 on Q. With this, it is not difficult to see that $\overline{\mathcal{A}}_i^0$ can bias towards 0 (almost) as well as $\overline{\mathcal{A}}_i^1$ can bias towards 1.

5 Cooperative-Strategy-Proof Fairness

5.1 Definition of Cooperative-Strategy-Proof Fairness

In a cooperative strategy, a corrupt coalition deviates from the honest protocol in an attempt to improve the coalition's overall wealth (i.e., the total reward). Cooperative strategies naturally arise in contexts where a corrupt coalition is allowed to have binding side contracts that allow the coalition to redistribute (e.g., equally) the overall wealth among its members. If a protocol is cooperative-strategy-proof fair (or CSP-fair), it intuitively means that any corrupt coalition should not be able to improve its overall wealth by more than negligible amounts (if the remaining parties are faithfully following the honest protocol).

Definition 3 (Cooperative-strategy-proof fairness or CSP-fairness)**.** Let $\mathfrak{A}$ be a family of adversaries that corrupt up to $n-1$ parties and let $\mathcal{P} \in \{0,1\}^n$ denote any divided preference profile. We say that an n-party coin toss protocol is cooperative-strategy-proof fair (or CSP-fair) for $\mathcal{P}$ and against the family $\mathfrak{A}$, iff for any adversary $\mathcal{A} \in \mathfrak{A}$, there exists some negligible function $\mathsf{negl}(\cdot)$, such that in an execution with the preference profile $\mathcal{P}$ and the adversary $\mathcal{A}$, the expected total reward for the set of corrupt parties (denoted C) is at most $\sigma(C) + \mathsf{negl}(\kappa)$ where $\sigma(C)$ denotes the expected total reward for all nodes in C in an all-honest execution.

Similar as before, now depending on the family $\mathfrak{A}$ of adversaries that we are concerned about, we can define computational, statistical, or perfect notions for cooperative-strategy-proof fairness, and for fail-stop, semi-malicious, or malicious adversaries respectively. We omit the detailed definitions for conciseness.

Remark 1 (The case of a global coalition for CSP-fairness)**.** Unless otherwise noted, the definition of CSP-fairness considers coalitions of size up to $n-1$. One could alternatively define a variant of CSP-fairness where the corrupt coalition can contain up to n parties, i.e., CSP-fairness is desired even against a global coalition where everyone is corrupt. For any *balanced* preference profile, this variant is equivalent to the definition where not all can be corrupt since the global coalition is indifferent to either outcome. For any *unbalanced* preference profile, this variant where all can be corrupt is a stronger notion — in fact, one could easily rule out feasibility against (even computationally bounded) semi-malicious adversaries due to the following argument. By correctness, there must exist some joint randomness $\vec{\rho}$ of all parties, such that an honest execution fixing the randomness to $\vec{\rho}$ would lead to the outcome that is preferred by the global coalition. Now a semi-malicious adversary can receive this $\vec{\rho}$ as advice and program the parties' joint randomness to $\vec{\rho}$. For fail-stop adversaries, we will show that *perfect* CSP-fairness is possible for any almost unanimous preference profile even when all parties can be corrupt (see Corollary 2).

For any *balanced* preference profile, if the corrupt coalition gains in terms of overall wealth (i.e., total payoff) then honest overall wealth must be harmed (relative to an honest execution in both cases). Therefore, CSP-fairness is equivalent to maximin fairness for balanced preference profiles. The following fact is therefore straightforward:

Fact 2 (Equivalence of maximin fairness and CSP fairness for balanced preference profiles)**.** *Let $\mathfrak{A}$ denote a family of adversaries that corrupt up to $n-1$ parties and let $\mathcal{P} \in \{0,1\}^n$ denote any balanced profile. Then, an n-party coin toss protocol Π is maximin fair for $\mathcal{P}$ against the family $\mathfrak{A}$ iff Π is CSP-fair for $\mathcal{P}$ against the family $\mathfrak{A}$.*

For *unbalanced* preference profiles, however, the two notions are not equivalent (and this will become obvious later in the paper).

As mentioned, for two parties, all our fairness notions equate to Blum's weak fairness notion [16], and therefore the results stated later in Appendix A.1 directly apply to CSP fairness too. In the remainder of this section, we focus on three or more parties.

5.2 Almost Unanimous Preference Profile

Recall that we consider 3 or more parties, i.e., $n \geq 3$.

Possibility of perfect CSP-fairness against semi-malicious adversaries. First, we show that for almost unanimous preference profiles and any $n \geq 3$, perfect CSP-fairness is possible against any coalition of size up to $n-1$.

Let $P_0, \ldots, P_{n-1}$ denote the $n \geq 3$ players. Without loss of generality, suppose that P_0 is the single 0-supporter (i.e., prefers 0), and everyone else prefers 1 (all other cases are equivalent by flipping the bit and renumbering players). Consider the following simple protocol denoted Π_{csp}.

1. In the first round, every party i where $i \in [0, 1, \ldots, n-1]$ locally tosses a random coin b_i. Further, the single 0-supporter P_0 reveals its coin b_0.
2. In the second round, every 1-supporter (i.e., P_i where $i \neq 0$) reveals coin b_i.
3. The outcome of the protocol is defined as follows: if any 1-supporter aborted without revealing its bit, output 0. Else, output the XOR of all bits that have been revealed by the parties — note that if P_0 aborted without revealing its bit b_0, then we simply do not include b_0 in the XOR.

It is straightforward that under an honest execution, the expected outcome is $\frac{1}{2}$.

Theorem 8 (Possibility of perfect CSP-fairness against semi-malicious corruptions for almost unanimous preference profiles)**.** *For any $n \geq 3$, there is an n-party coin toss protocol that achieves perfect CSP-fairness for any almost unanimous preference profile $\mathcal{P} \in \{0,1\}^n$ against the family of all semi-malicious adversaries that control at most $n-1$ parties.*

Proof. We analyze the aforementioned protocol Π_{csp} by considering the following cases:

1. **P_0 is the lone corrupt party**. In this case, all parties who prefer 1 are honest, and since P_0 makes its decision to abort prior to seeing the remaining parties' random bits, equivalently, we can think of the remaining parties flip their random coins after P_0 makes its decision whether to abort. Thus, regardless of P_0's strategy, the expected outcome must be $\frac{1}{2}$.
2. **P_0 and a single 1-supporter are corrupt.** In this case, the definition of CSP-fairness is trivially satisfied since the corrupt coalition would obtain a payoff of exactly 1 no matter what the outcome of the protocol is.
3. **P_0 is honest and one or more 1-supporters are corrupt.** Let $\mathbf{b} := (b_0, \ldots, b_{n-1})$ denote the random coin tosses of all the parties. For semi-malicious corruption, we can imagine that each party P_i chooses b_i and other randomness related to aborting decisions upfront prior to

protocol start — honest parties sample them at random and corrupt parties choose the random strings arbitrarily. Let C denote the corrupt coalition and let $-C$ denote its complement. We consider an alternative adversary $\mathcal{B}$ that just receives $\mathbf{b}^C := \{b_i\}_{i \in C}$ as advice but all corrupt parties follow the protocol to the end — note that such a $\mathcal{B}$ needs to consume only $\mathbf{b}^C$ and no additional randomness. For any fixed $\mathbf{b}^C$, and for any fixed $\mathbf{b}^{-C}$, if playing with the adversary $\mathcal{B}$ who never aborts, the outcome is 0, then playing with any adversary $\mathcal{A}$ (who might abort), the outcome cannot be 1. Thus for every $(\mathbf{b}^C, \mathbf{b}^{-C})$, no adversary $\mathcal{A}$ can obtain a higher outcome than $\mathcal{B}$. The proof follows by seeing that for $\mathcal{B}$ and for any fixed $\mathbf{b}^C$, the expected outcome (averaging over honest parties' random coin flips) is $\frac{1}{2}$.

4. **P_0 and at least two 1-supporters are corrupt.** In this case it must be that $n \geq 4$ since if $n = 3$ all parties would be corrupt. Similar to the above case, here we can argue that for every fixed $(\mathbf{b}^C, \mathbf{b}^{-C})$ and P_0's decision whether to abort, the adversary $\mathcal{B}$ such that all other corrupt corrupt (besides P_0) execute to the end makes the outcome at least as high as any other adversary $\mathcal{A}$. Additionally, for $\mathcal{B}$ and for any fixed $\mathbf{b}^C$ and P_0's decision whether to abort, the expected outcome (averaging over honest parties' random coin flips) is $\frac{1}{2}$.

□

Corollary 1. *There is an* 3*-party coin toss protocol that achieves perfect CSP-fairness for any divided preference profile against the family of all semi-malicious adversaries that control at most* $n-1$ *parties.*

Proof. Note that for 3 parties, any divided preference profile must be almost unanimous. The corollary now follows from Theorem 8. □

We observe that for fail-stop adversaries, a variant of the aforementioned protocol Π_{csp} actually achieves perfect CSP-fairness even when all parties can be corrupt: Suppose that only parties in $\{P_1, \ldots, P_{n-1}\}$ flip a random coin and publish the coin; and P_0 does nothing. If any of these parties abort, the outcome is defined to be 0; else the outcome is defined to be the XOR of all published coins. We thus have the following corollary:

Corollary 2. *For any* $n \geq 3$*, there is an* n*-party coin toss protocol that achieves perfect CSP-fairness for any almost unanimous preference profile* $\mathcal{P} \in \{0,1\}^n$ *against the family of all fail-stop adversaries that control up to* n *parties.*

Proof. If no 1-supporter is corrupt, then obviously the expected outcome is $\frac{1}{2}$. If at least one 1-supporter is corrupt, then for every choice of the joint randomness of all 1-supporters, having any 1-supporter abort does no better for the adversary than having no 1-supporter abort. □

Possibility of computational CSP-fairness against malicious adversaries. It is also easy to see that for three or more parties, *statistical* CSP fairness is impossible against malicious adversaries much as the 2-party case [33,44]. Therefore for malicious adversaries we have to make computational assumptions.

For conceptual simplicity, we first describe our protocol assuming an idealized commitment scheme — in Appendix B, we describe how to dispense with this idealized primitive and realize it from concurrent non-malleable commitments that can be constructed one-way permutations. For the time being, imagine that there is a special trusted party called $\mathcal{F}_{\text{idealcomm}}$ that has the following interface:

- In the first round (i.e., the commitment round), if $\mathcal{F}_{\text{idealcomm}}$ receives ($\texttt{commit}\ b$) from some party i, it tells everyone $(\texttt{committed}, i)$.
- In any of the subsequent rounds (i.e., the opening rounds), if $\mathcal{F}_{\text{idealcomm}}$ receives $\texttt{open}$ from any party i who has committed b_i in the first round, it tells everyone $(\texttt{open}, i, b_i)$.

We can now upgrade our semi-malicious protocol earlier to resist even malicious adversaries (w.l.o.g. assume that there is a single 0-supporter and everyone else is a 1-supporter):

1. In round 0, everyone commits a bit to $\mathcal{F}_{\text{idealcomm}}$;
2. In round 1, the single 0-supporter opens its commitment;
3. In round 2, everyone else opens;
4. If any 1-supporter aborted, the outcome is 0; else the outcome is the XOR of all bits that have been opened.

Since the commitment round basically forces corrupt parties to commit to their randomness upfront; it is easy to see that this new protocol is CSP-fair against malicious adversaries (for the same reason why the earlier protocol is CSP-fair against semi-honest adversaries). Note that CSP fairness holds even for unbounded adversaries assuming the $\mathcal{F}_{\text{idealcomm}}$ ideal functionality; but in Appendix B, we show how to remove the $\mathcal{F}_{\text{idealcomm}}$ and replace it with concurrent non-malleable commitments [42], the resulting protocol would secure only against computationally bounded adversaries as stated in the following theorem.

Theorem 9 (Computational CSP fairness against malicious adversaries). *Assume that one-way permutations exist, then for any $n \geq 3$, there exists an n-party protocol that achieves computational CSP fairness for any almost unanimous preference profile $\mathcal{P} \in \{0,1\}^n$ against malicious coalitions of size up to $n-1$.*

The proof is deferred to Appendix B.

5.3 Amply Divided Preference Profile

For $n = 3$, any divided preference profile must be almost unanimous. For $n \geq 4$, we need to consider amply divided preference profiles: i.e., at least two parties prefer 0 and at least two parties prefer 1. We now show a strong impossibility for divided preference profiles, that is, for any divided preference profile $\mathcal{P}$, no n-party coin toss can achieve even computational CSP-fairness for $\mathcal{P}$ against even fail-stop adversaries.

We note that for the special case of amply divided and *balanced* preference profiles, the impossibility for CSP fairness is already implied by the impossibility of maximin fairness for the same preference profiles (Theorem 5) — recall that the two notions are equivalent for balanced preference profiles. However, this observation does not rule out the feasibility of CSP fairness for *unbalanced* and amply divided preference profiles. Thus the following theorem is non-trivial even in light of Theorem 5.

Theorem 10 (Impossibility of CSP-fairness for $n \geq 4$). *Let $n \geq 4$, and let $\mathcal{P} \in \{0,1\}^n$ be any amply divided preference profile. Then, no n-party coin-toss protocol can achieve even computational CSP-fairness for $\mathcal{P}$, against even fail-stop adversaries.*

Proof roadmap. Although for *balanced* and amply divided preference profiles, the infeasibility of CSP fairness is already implied by the infeasibility of maximin fairness for the same profiles (since the two notions are equivalent for balanced preference profiles), here we would like to prove impossibility for *any* amply divided preference profile, even *unbalanced* ones. At a very high level, our approach is to group the parties into three partitions called P_1, P_2, and P_3, such that we can view the execution as a 3-party protocol. This partitioning is carefully crafted such that the definition of CSP fairness would imply the T_2-equality lemma, the lone-wolf condition, and the wolf-minion conditions like in the impossibility proof for maximin fairness — and if this is the case, the same proof would apply and rule out CSP fairness.

Among these conditions, the T_2-equality lemma is the most challenging to prove. Specifically, earlier we relied on maximin fairness against a *semi-malicious* P_2 to prove the T_2-equality lemma; and here would like to prove the same lemma for CSP fairness but now against a *fail-stop* adversary[6]. This seems almost counter-intuitive at first sight since at the surface, the T_2-equality lemma is stating that *if a semi-malicious adversary were to program T_2 to specific strings, almost for all such strings it would not help.* But now how can we prove it by relying on CSP fairness against only *fail-stop* adversaries? In our formal proof later, we will show that for any two neighboring T_2 and T_2' (except for a negligibly small bad fraction), it must be that $|f(T_2) - f(T_2')| \leq \mathsf{negl}(\kappa)$, where T_2 and T_2' are said to be neighboring iff they differ only in one party's contribution of random coins, and $f(T_2)$ is defined similarly as before, i.e., the expected outcome of an honest execution conditioned on P_2's randomness being fixed to T_2. Now if we can show this, we can then show, through a hybrid argument, that $|f(T_2) - f(T_2')| \leq \mathsf{negl}(\kappa)$ for any T_2 and T_2' (except for a negligibly small bad fraction), and this would complete the proof.

Thus the challenge is to show $|f(T_2) - f(T_2')| \leq \mathsf{negl}(\kappa)$ for almost all *neighboring* T_2 and T_2' pairs. To do this, suppose that T_2 and T_2' differ in the i-th player's contribution where $i \in P_2$ — our intuition is to compare an honest execution involving T_2 with the execution where the i-th player aborts upfront (and P_2's randomness still fixed to T_2). Let $g^i(T_2)$ denote the expected outcome in the latter execution. Through a somewhat non-trivial argument, we will prove that for almost all T_2s, it must be that $|f(T_2) - g^i(T_2)| \leq \mathsf{negl}(\kappa)$ — otherwise we can construct a *fail-stop* adversary in control of P_2, and this adversary, upon generating an honest random T_2, emulates polynomially many honest executions conditioned on T_2 to estimate $f(T_2)$ and $g^i(T_2)$ respectively, and informed by the estimates, decide to either have i abort upfront or not. We prove that such an adversary can cause non-negligible bias that improves P_2's overall wealth.

Similarly, for T_2' that is almost identical as T_2 but differing in the i-th coordinate, we also have that $|f(T_2') - g^i(T_2')| \leq \mathsf{negl}(\kappa)$. Finally, the proof follows by observing that, if the i-th party aborts upfront, then its random coins do not affect the expected outcome of the execution, i.e., $g^i(T_2) = g^i(T_2')$.

We defer the full proof of this theorem to Appendix B.3.

6 Fairness by Strong Nash Equilibrium

6.1 Definition of Strong Nash Equilibrium (SNE)

Strong Nash Equilibrium (SNE) requires that no coalition, corrupting up to n parties, can noticeably (i.e., non-negligibly) increase the payoff of all members of the coalition. SNE is weaker than the earlier CSP notion since the former only needs to resist a subset of the coalition strategies that latter must resist — CSP must not only defend against coalition strategies that benefit all of

[6]Note that the T_2-equality lemma does not even hold for maximin fairness against *fail-stop* adversaries since we have an explicit construction for almost unanimous preference profiles and fail-stop.

its members, but also defend against strategies that benefit coalition members on average[7]. More formally, we define SNE-fairness below.

Definition 4 (Strong Nash Equilibrium or SNE-fairness)**.** Let $\mathfrak{A}$ be a family of adversaries that corrupt up to n parties and let $\mathcal{P} \in \{0,1\}^n$ be any divided preference profile. We say that an n-party coin toss protocol is SNE-fair for $\mathcal{P}$ and against the family of adversaries $\mathfrak{A}$ iff for any $\mathcal{A} \in \mathfrak{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$, such that in an execution with the preference profile $\mathcal{P}$ and the adversary $\mathcal{A}$, there is at least one corrupt party whose expected payoff is less than $\frac{1}{2} + \mathsf{negl}(\kappa)$.

Note that the definition of SNE-fairness requires that the notion be satisfied *even when all parties are corrupt.* Similar as before, depending on the family $\mathfrak{A}$ of adversaries that we are concerned about, we can define computational, statistical, or perfect notions for SNE-fairness, and for fail-stop, semi-malicious, or malicious adversaries respectively. We omit the detailed definitions for conciseness.

A coalition of parties is said to be *unanimous* iff every party in the coalition prefers the same bit.

Fact 3. *Let $\mathfrak{A}$ be a family of adversaries corrupting up to n parties and let $\mathfrak{A}' \subset \mathfrak{A}$ be the (maximal) subset of $\mathfrak{A}$ that corrupts only unanimous coalitions*[8]*. Let $\mathcal{P} \in \{0,1\}^n$ be any divided preference profile. Then, an n-party coin toss protocol Π is CSP-fair for $\mathcal{P}$ against the family $\mathfrak{A}'$ iff Π is SNE-fair for $\mathcal{P}$ against the family $\mathfrak{A}$.*

Proof. For any adversary $\mathcal{A} \in \mathfrak{A}$ that corrupts a coalition that has divided preferences, if the coalition memebers that prefer 0 have expected payoff more than $\frac{1}{2}$, then those who prefer 1 must have payoff at most $\frac{1}{2}$ — thus SNE-fairness is trivially satisfied for divided coalitions. We therefore conclude that a protocol Π to be SNE-fair for $\mathcal{P}$ against $\mathfrak{A}$, if and only if Π is SNE-fair for $\mathcal{P}$ against those adversaries in $\mathfrak{A}$ that control unanimous coalitions — and this latter notion is equivalent to CSP-fair for unanimous coalitions, by observing the following: since $\mathcal{P}$ is divided, any adversary in $\mathfrak{A}$ that controls unanimous coalitions corrupts only up to $n-1$ parties (recall that the definition of CSP-fair considers adversaries that corrupts upto $n-1$ parties). ☐

6.2 Feasibility Results for SNE Fairness

We show that for any $n \geq 2$, there is an n-party coin toss protocol that is computationally SNE-fair for any divided preference profile $\mathcal{P} \in \{0,1\}^n$ against even malicious adversaries; further, there is an n-party coin toss protocol that is perfectly SNE-fair for any divided preference profile $\mathcal{P} \in \{0,1\}^n$ against semi-malicious adversaries. On the other hand, the impossibility of statistical SNE fairness against malicious adversaries is implied in a straightforward fashion by known lower bounds [33,44].

Achieving perfect SNE-fairness against semi-malicious adversaries. Let $n \geq 3$ and let $\mathcal{P} \in \{0,1\}^n$ be a divided preference profile. We can consider a simple dueling protocol: pick two people with opposing preferences (i.e., the ones with the smallest party identifiers) and have them play the simple 2-party protocol: each party picks a random bit upfront and both broadcast their bit in the first round. Normally the outcome is the XOR of the two bits but if one party aborts, the outcome is the other party's preference.

[7]Since SNE only needs to defend against unanimous coalitions by Fact 3, for any divided preference profile we in fact only need to consider coalitions of size $n-1$ rather than n.

[8]Recall that we assume that the choice of corrupt parties is hard-wired in an adversary's algorithm.

Theorem 11 (Perfect SNE-fairness against semi-malicious adversaries). *For any $n \geq 2$, there is an n-party coin toss protocol that is perfectly SNE-fair for any divided preference profile $\mathcal{P} \in \{0,1\}^n$ against semi-malicious adversaries.*

Proof. By Fact 3, we only need to resist unanimous coalitions. Thus for the two parties selected to duel with opposing preferences, one of them must be honest. Further, recall that a semi-malicious adversary must select its random coins upfront without seeing any protocol message, and henceforth the only attack it can perform is aborting. Now in the 2-party protocol, for any choice of randomness of the 2 dueling parties, if the corrupt party aborts, it does no better than playing honestly till completion. □

Achieving computational SNE-fairness against malicious adversaries. The above protocol can be made secure against malicious adversaries using a cryptographic commitment scheme. The only change needed is that when the selected two parties duel, one of them (denoted P) commits to a bit in Phase 0, then the other party (denoted P') sends its bit in Phase 1, and finally P opens its commitment.

Theorem 12 (Computational SNE-fairness against malicious adversaries). *For any $n \geq 2$, there is an n-party coin toss protocol that achieves computational SNE-fairness for any divided preference profile $\mathcal{P} \in \{0,1\}^n$ against malicious adversaries.*

Proof. Consider the dueling protocol Π_{duel}. By Fact 3, it suffices to prove that any unanimous coalition cannot non-negligibly improve the coalition's total reward. Notice that any unanimous coalition controls at most one party in the two parties selected to duel. By maximin fairness of the 2-party protocol (which we argue in Appendix A.1), if one of the dueling parties deviates, the deviating party cannot improve its expected payoff by more than a negligible amount. □

7 The Case of Private Preference Profiles

Here we consider the case of private preference profiles, where each party's preference is private information only known to the party. In other words, we consider *private preference* coin toss protocols, where each party's preference is a private input, instead of public information. Clearly, this is a more challenging setting for achieving fairness. For example, a malicious party may lie about his preference or abort without revealing his preference. Indeed, as we shall see, we lose some feasibility results in the private preference setting.

Recall that in the public preference setting, coin toss protocols and fairness can be naturally defined with respect to a preference profile $\mathcal{P}$. However, this is not the case for private preference. Thus, we only consider (universal) n-party private preference coin toss protocols that are defined for every preference profiles $\mathcal{P} \in \{0,1\}^n$. All three fairness notions can be naturally defined for such protocols. Below we only state the definition of maximin fairness in the private preference setting formally for succinctness. The other two notions can be defined analogously.

Definition 5 (Maximin fairness). Let $\mathfrak{A}$ be a family of adversaries that corrupt up to $n-1$ parties. We say that an n-party private preference coin toss protocol is private maximin fair against the family $\mathfrak{A}$, iff for every adversary $\mathcal{A} \in \mathfrak{A}$, there exists some negligible function $\mathsf{negl}(\cdot)$ such that for every divided preference profile $\mathcal{P} \in \{0,1\}^n$, in an execution with the preference profile $\mathcal{P}$ and the adversary $\mathcal{A}$, the expected reward for any honest party is at least $\frac{1}{2} - \mathsf{negl}(\kappa)$. For unanimous preference profiles, the execution should output the common preference with probability 1.

We proceed to discuss the feasibility and impossibility of fair coin toss for private preference protocols. As this is harder to achieve, all impossibility results in the public preference setting trivially hold here, and it suffices to investigate cases that are feasible in the public preference setting.

SNE-fairness. Recall that even in the public preference setting, we can only achieve general feasibility result for the notion of SNE-fairness, where computational SNE-fairness against malicious adversary and statistical SNE-fairness against semi-malicious adversary are feasible for any $n \geq 2$ parties (whereas maximum and CSP-fairness are impossible for $n \geq 4$ even against fail stop adversary). In the private preference setting, we show that SNE-fairness against malicious adversary becomes impossible for $n \geq 3$ parties, whereas SNE-fairness against semi-malicious adversary remain feasible. Intuitively, the reason for the impossibility is that a malicious adversary may lie about his preference.

Theorem 13 (Impossibility of SNE-fairness against malicious adversary)**.** *For any $n \geq 3$, no n-party private preference coin-toss protocol can achieve even computational SNE-fairness against malicious adversaries.*

Proof. (sketch) We focus on the three-party case and discuss how to handle general $n \geq 4$ parties at the end of the proof. At a high level, the proof for the three-party case relies on the same argument as that of Theorem 7 for maximin-fairness. Recall that in the proof of Theorem 7, we consider preference profile $\mathcal{P} = (1, 0, 1)$. We show that maximin-fairness implies T_2-equality lemma (Lemma 1) and the lone-wolf and wolf-minion conditions. Then we use these properties to derive a contradiction by constructing an adversary that breaks the wolf-minion condition. Here, we follow the same strategy to consider preference profile $\mathcal{P} = (1, 0, 1)$. It suffices to show that private SNE-fairness implies the same set of properties, and a contradiction can be derived in the same way.

Let Π be a three-party private preference coin toss protocol. Recall that we use the notation $T_1, T_2, T_3 \in \{0, 1\}^{\ell(\kappa)}$ to denote the randomness of P_1, P_2 and P_3, respectively, and $f(T_2)$ to denote the expected outcome when P_2 uses the randomness T_2 whereas P_1 and P_3 executed the protocol honestly (when the preference profile is $\mathcal{P} = (1, 0, 1)$).

To show that Lemma 1 holds, we adopt the same proof as presented in Appendix A.4. Observe that both Fact 4 and 5 are implied by private SNE-fairness for the same reasons. Specifically, Fact 4 follows by the security against semi-malicious P_2 and Fact 5 follows by the correctness of the honest execution. Thus, T_2-equality lemma is implied by private SNE-fairness as well. It remains to check the lone-wolf and wolf-minion conditions.

For the lone-wolf conditions, it may seem that SNE-fairness only implies that P_1 (or P_3) cannot cause non-negligible (in κ) bias towards 1 by a fail-stop attack. This is the place that an adversary can take the advantage of private preference. Suppose there P_1 can cause non-negligible bias towards 0 by a fail-stop attack when the preference profile is $(1, 0, 1)$. Consider the case that the preference profile is $(0, 0, 1)$. An malicious P_1 (with preference 0) can participate the protocol with a pretended preference 1 and perform the fail-stop attack to cause bias toward 0 to violate fairness. Thus, a fail stop P_1 (or P_3) cannot cause non-negligible bias towards either direction.

Recall that the wolf-minion condition says that when P_1 and P_2 (or P_2 and P_3) form a fail-stop coalition, they cannot cause a non-negligible (in κ) bias towards 0. Suppose this is not the case, e.g., a fail-stop coalition P_1 and P_2 can cause a non-negligible bias towards 0. We show that SNE-fairness can be violated when the preference profile is $(0, 0, 1)$. Indeed, in this case, an malicious adversary corrupting P_1 and P_2 can pretend the preference of P_1 is 1 and use the assumed fail-stop attack to cause a non-negligible bias towards 0, which violates SNE-fairness.

The above shows that for three-party protocols, the properties needed in the proof of Theorem 7 are implied by private SNE-fairness. A contradiction can then be derived by the same arguments as in Theorem 7, which proves the impossibility.

Finally, for general $n \geq 4$ parties, we can use the standard trick to group $P_4, \ldots, P_n$ together with P_2 to form a supernode of 0-supporters. This effectively reduce the number of parties to 3 and the same argument can be applied to show impossibility. $\square$

Theorem 14 (Perfect private SNE-fairness against semi-malicious adversaries)**.** *For any $n \geq 2$, there is an n-party private preference coin toss protocol that is perfectly private SNE-fair against semi-malicious adversaries.*

Proof. We simply modify the public preference duelling protocol by first asking all parties to reveal their private preference. If any parties abort, we ignore them. For the remaining non-aborting parties, we proceed with the dueling protocol as in the public preference setting. Note that since we only consider semi-malicious adversaries, the revealed preferences must be the true preferences.

By Fact 3 (which can be verified to hold in the private preference setting with the same argument) , we only need to resist unanimous coalitions. Hence, all aborting parties must share the same preference as their non-aborting coalition (if any), who do not gain any advantage by the fairness of the public preference protocol. If all non-aborting parties are honest, then correctness of the honest execution also implies that the aborting parties do not gain any advantage. $\square$

Note that Theorem 14 is proved by a protocol that first asks all parties to reveal their private preference and then executes a public preference protocol among the non-aborting parties, and intuitively, this works since the semi-malicious can only reveal their true preferences. However, while this intuition turns out to be true for SNE-fairness and maximin fairness (which we discuss later), it can be subtle for CSP-fairness since the adversary still has the advantage of aborting before revealing his preference. We discuss this next.

CSP-fairness. Recall that in the public preference setting, Corollary 2 says that for $n \geq 3$, there exists an n-party coin toss protocol that achieves perfect CSP-fairness for any almost unanimous preference profile against all fail-stop adversaries that can control up to n parties. In particular, there exists a three-party perfect CSP-fair protocol against fail-stop adversaries who may corrupt all three parties[9]. Interesting, this becomes impossible in the private preference setting.

Theorem 15 (Impossibility of CSP-fairness against fail-stop all-corruption adversary)**.** *No three-party private preference coin-toss protocol can achieve computational CSP-fairness against fail-stop adversaries that can corrupt up to three parties.*

Proof. (sketch) For the sake of contradiction, suppose Π is a three-party private preference coin-toss protocol that achieve the claimed fairness. Let us consider a scenario where P_3 always abort at the beginning, and P_1 and P_2 has preference 0 and 1, respectively. Note that suppose P_1 and P_2 execute the protocol honestly, the outcome need to be unbiased: Suppose the outcome is biased towards b and the private preference of P_3 is also b, then the CSP-fairness is violated.

Thus, in this scenario where P_3 is aborting, honest P_1 and P_2 execute a two-party protocol and produce an unbiased outcome. We can apply Cleve's lower bound argument to show the existence of a fail-stop adversary P_a that can bias the outcome non-negligibly towards b, for some $a \in \{1, 2\}$ and $b \in \{0, 1\}$. Now, suppose the private preference of P_3 is b, and consider an adversary $\mathcal{A}$ that corrupts all three parties and does the following: (i) $\mathcal{A}$ lets P_3 aborts at the beginning, and (ii) $\mathcal{A}$

[9] We focus on the three-party case here since the case of four or more parties are impossible in the private preference setting due to the existence of amply divided preference profiles for four or more parties.

let P_a to perform the fail-stop attack to cause non-negligible bias of the outcome towards b. This violates CSP-fairness since the total utility of the corrupted parties is increased by a non-negligible amount. □

On the positive side, we observe that Corollary 1 extends to the private preference setting.

Theorem 16. *There is an* 3*-party private preference coin toss protocol that achieves perfect CSP-fairness against the family of all semi-malicious adversaries that control at most* 2 *parties.*

Proof. (sketch) We follow the same strategy to first ask each party reveal his preference, and then let the non-aborting parties to execute a fair public preference protocol. Specifically, if no party aborts, then we run the three-party CSP-fair protocol in Corollary 1. If one party aborts and the remaining two parties have the same preference, then they output their preference. If one party aborts and the remaining two parties have different preferences, then they execute the dueling protocol. If two parties abort, then the remaining party simply output his preference. It is not hard to see by inspection that private CSP-fairness holds in all cases. □

Maximin fairness. We end this section with a brief discussion on the maximin fairness for the private preference protocols. Note that the only interesting question is whether Theorem 6, which states the existence of perfect maximin fair coin toss protocol against fail-stop adversaries, extends to the private preference setting. Now, observe that the definition of maximin fairness only concerns the honest party's utility, so an adversary who aborts without revealing his preference cannot hurt maximin fairness. Therefore, the strategy of first asking each party to reveal his preference, and then letting the non-aborting parties to execute a fair public preference protocol works directly for maximin fairness.

Theorem 17 (Possibility of perfect maximin fairness for 3 parties and fail-stop adversaries.)**.** *There exists a 3-party private preference coin toss protocol that achieves perfect maximin fairness against fail-stop adversaries.*

8 Related Work

Related works on strongly fair coin toss [18,27] as well as Blum's notion of weak fair coin toss [16] have been discussed earlier in Section 1. In this section, we discuss additional related work.

Game theory and cryptography. Historically, game theory [36,48] and multi-party computation [27,52,53] were investigated by separate communities. Some recent efforts have investigated the marriage of game theory and cryptography (see the excellent surveys by Katz [37] and by Dodis and Rabin [23]). This line of work has focused on two broad types of questions:

- First, a line of works [1,4–6,32,38,49] investigated how to define game-theoretic notions of security (as opposed to cryptography-style security notions) for multi-party computation tasks such as secret sharing and secure function evaluation. Existing works consider a different notion of utility than us: specifically, these works make (a subset to all of) the following assumptions about players' utility: players prefer to compute the function correctly; further, they prefer to learn secrets, and prefer that other players do not learn secrets. These works then investigate how to design protocols such that rational players will be incentivized to follow the honest protocol.

- Second, a line of work has asked how cryptography can help traditional game theory. Particularly, many classical works in game theory [36, 48] assumes the existence of a trusted mediator — and recent works have shown that under certain conditions, this trusted mediator can be implemented using cryptography [9, 22, 29, 34].

In this paper, we investigate game-theoretic notions of fairness for coin toss protocols. Our notions are novel in comparison with the aforementioned related work. First, to the best of our knowledge, we are the first to apply game theory to coin toss protocols, and asking whether we can circumvent known impossibilities [18] by considering rational players. Second, the fairness notions proposed in this paper are novel and to the best of our knowledge have not been investigated before for multiple parties. Specifically, we consider a natural notion of utility for coin toss protocols, where players have a preference over the outcome of the coin toss. We require that an honest execution produces an unbiased coin (unless all parties prefer the same bit); however if one or more coalition(s) deviate from the honest protocol, the coin toss outcome need not be unbiased (but we want that certain fairness properties must be preserved). All notions of fairness defined in the paper consider *corrupt majority* — since in the case of honest majority, strongly fair coin toss is known to be possible assuming standard cryptography assumptions [27], and the standard strong fairness notion implies all game-theoretic notions considered in this paper. In comparison, most earlier works [1, 4–6, 9, 22, 29, 32, 34, 38, 49] at the intersection of cryptography and game theory consider only the popular Nash equilibrium notion that is concerned about coalitions of size 1. Our fairness definitions are inspired by equilibrium notions in game theory that resist coalitions in various capacities [15, 35].

Other notions of fairness. Our work is inspired by the study of new, financially motivated fairness notions in blockchains and cryptocurrency applications [3,8,13,21,39–41,45]. Several recent works [13, 21, 39, 40] show that to achieve a suitable notion of financial fairness, the protocol may require that parties place collateral on the blockchain to participate, and misbehaving parties can be penalized by taking away their collateral. Among these works, the most closely related to ours are those that investigate lottery-style protocols [8,13,21,39,45]. While earlier works [3,13] require quadratic amount of collateral, more recent works [8,45] showed that it is possible to realize fair lottery in the presence of a blockchain (i.e., a broadcast medium with identifiable abort) requiring no collateral at all, by relying on a folklore tournament-tree approach. Interestingly, although not explicitly noted, all these works on fair lottery over a blockchain [8, 13, 21, 39, 45] adopt a game theoretic notion of fairness, that is, although a deviating coalition can bias the outcome of toss of the n-sided dice, such bias must be towards a direction that harms the perpetrators. In fact, the implicit fairness notion in these papers is equivalent to our notion of maximin fairness and cooperative-strategy-proof (CSP) fairness — for 0-sum games like a lottery, these two notions are equivalent.

Other relaxations of strong fairness have also been considered for coin toss and multi-party computation. For example, several works [2, 7, 10, 11, 14, 17–20, 28, 30, 31, 46] consider a notion of ϵ-fairness, i.e., the adversary can bias the coin by at most a non-negligible ϵ amount. Moran et al. [46] showed that for general R, there is an R-round, 2-party coin toss protocol that satisfies $O(1/R)$-fairness — and this is optimal since Cleve [18] showed that for every R-round 2-party coin toss protocol, there exists an efficient adversary that can bias the honest party's outcome by at least $\Omega(1/R)$.

9 Conclusion

In this paper we proposed several natural, game theoretic notions of fairness for multi-party coin toss protocols. In the case of two parties, all of these notions equate to Blum's notion of weakly fair coin toss [16]; however, for more than 2 parties, these notions differ in strength and lead to different feasibility and infeasibility results. We summarize the strengths of various notions from strongest to weakest (for general n and divided preference profiles).

Maximin $\neq$ **Cooperative-Strategy-Proof (CSP)** $>$ **Cooperative-Coalition-Proof (CCP)** $=$ **Strong Nash Equilibrium (SNE)** $>$ Coalition-Proof $>$ Nash

Among the above notions, we show broad feasibility results for SNE-fairness (which directly implies feasibility for coalition-proof equilibrium and Nash Equilibrium too). For other notions, we give a complete characterization of their feasibilities and infeasibilities — and for all of them we prove infeasibilities for amply divided preference profiles.

Note that among these notions, cooperative-strategy-proof and cooperative-coalition-proof are new notions first proposed in this paper — although we study them in the context of coin toss protocols, they would make sense for general games with transferrable utilities too. Finally, although maximin fairness is incomparable to CSP-fairness in general, the two are equivalent for balanced preference profiles (analogous to zero-sum games).

Acknowledgments

We gratefully acknowledge Iddo Bentov and Andrew Miller: discussions with the two of them provided initial motivation for this work; moreover, Iddo Bentov also contributed ideas that inspired our lower bound proofs. We would like to acknowledge helpful technical discussions with Feng-Hao Liu, Qiang Tang, Gilad Asharov, Ilan Komargodski, Muthu Venkitasubramaniam, and Antigoni Polychroniadou. This work is supported in part by NSF grants CNS-1314857, CNS-1514261, CNS-1544613, CNS-1561209, CNS-1601879, CNS-1617676, an Office of Naval Research Young Investigator Program Award, a Packard Fellowship, a DARPA Safeware grant (subcontractor under IBM), a Sloan Fellowship, Google Faculty Research Awards, a Baidu Research Award, and a VMWare Research Award.

References

[1] I. Abraham, D. Dolev, R. Gonen, and J. Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *PODC*, 2006.

[2] B. Alon and E. Omri. Almost-optimally fair multiparty coin-tossing with nearly three-quarters malicious. In *TCC*, 2016.

[3] M. Andrychowicz, S. Dziembowski, D. Malinowski, and L. Mazurek. Secure Multiparty Computations on Bitcoin. In *S&P*, 2013.

[4] G. Asharov, R. Canetti, and C. Hazay. Towards a game theoretic view of secure computation. In *Eurocrypt*, 2011.

[5] G. Asharov and Y. Lindell. Utility dependence in correct and fair rational secret sharing. In *CRYPTO*, 2009.

[6] G. Asharov and Y. Lindell. Utility dependence in correct and fair rational secret sharing. *Journal of Cryptology*, 24(1), 2011.

[7] B. Awerbuch, M. Blum, B. Chor, S. Goldwasser, and S. Micali. How to implement bracha's o (log n) byzantine agreement algorithm. *Unpublished manuscript*, 1985.

[8] M. Bartoletti and R. Zunino. Constant-deposit multiparty lotteries on bitcoin. In *Financial Cryptography and Data Security*, 2017.

[9] A. Beimel, A. Groce, J. Katz, and I. Orlov. Fair computation with rational players. `https://eprint.iacr.org/2011/396.pdf`, full version of Eurocrypt'12 proceeding version by Groce and Katz, 2011.

[10] A. Beimel, I. Haitner, N. Makriyannis, and E. Omri. Tighter bounds on multi-party coin flipping via augmented weak martingales and differentially private sampling. Technical Report TR17-168, Electronic Colloquium on Computational Complexity, 2017. 7, 2017.

[11] A. Beimel, E. Omri, and I. Orlov. Protocols for multiparty coin toss with a dishonest majority. *Journal of Cryptology*, 28(3):551–600, 2015.

[12] M. Ben-or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *STOC*, 1988.

[13] I. Bentov and R. Kumaresan. How to Use Bitcoin to Design Fair Protocols. In *CRYPTO*, 2014.

[14] I. Berman, I. Haitner, and A. Tentes. Coin flipping of any constant bias implies one-way functions. *JACM*, 65(3):14, 2018.

[15] B. Bernheim, B. Peleg, and M. DWhinston. Coalition-proof nash equilibria i. concepts. *Journal of Economic Theory*, 42(1), 1987.

[16] M. Blum. Coin flipping by telephone. In *CRYPTO*, 1981.

[17] N. Buchbinder, I. Haitner, N. Levi, and E. Tsfadia. Fair coin flipping: Tighter analysis and the many-party case. In *SODA*, 2017.

[18] R. Cleve. Limits on the security of coin flips when half the processors are faulty. In *STOC*, 1986.

[19] D. Dachman-Soled, Y. Lindell, M. Mahmoody, and T. Malkin. On the black-box complexity of optimally-fair coin tossing. In *TCC*, 2011.

[20] D. Dachman-Soled, M. Mahmoody, and T. Malkin. Can optimally-fair coin tossing be based on one-way functions? In *TCC*, 2014.

[21] K. Delmolino, M. Arnett, A. E. Kosba, A. Miller, and E. Shi. Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab. In *Financial Cryptography and Data Security*, 2016.

[22] Y. Dodis, S. Halevi, and T. Rabin. A cryptographic solution to a game theoretic problem. In *CRYPTO*, 2000.

[23] Y. Dodis and T. Rabin. Cryptography and game theory. In *Algorithmic Game Theory*, 2007.

[24] B. Games. One-night werewolf.

[25] J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Eurocrypt*, 2015.

[26] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 25–32, New York, NY, USA, 1989. ACM.

[27] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC*, 1987.

[28] S. D. Gordon and J. Katz. Partial fairness in secure two-party computation. *J. Cryptology*, 25(1):14–40, 2012.

[29] A. Groce and J. Katz. Fair computation with rational players. In *Eurocrypt*, 2012.

[30] I. Haitner and E. Omri. Coin flipping with constant bias implies one-way functions. *SIAM Journal on Computing*, 43(2):389–409, 2014.

[31] I. Haitner and E. Tsfadia. An almost-optimally fair three-party coin-flipping protocol. *SIAM Journal on Computing*, 46(2):479–542, 2017.

[32] J. Halpern and V. Teague. Rational secret sharing and multiparty computation. In *STOC*, 2004.

[33] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. In *FOCS*, 1989.

[34] S. Izmalkov, S. Micali, and M. Lepinski. Rational secure computation and ideal mechanism design. In *FOCS*, 2005.

[35] R. J.Aumann. Acceptable points in general cooperative n-person games. Contributions to the Theory of Games IV", Princeton Univ. Press, Princeton, N.J., 1959.

[36] R. J.Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1(1), 1974.

[37] J. Katz. Bridging game theory and cryptography: Recent results and future directions. In *TCC*, 2008.

[38] G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In *TCC*, 2008.

[39] A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *S&P*, 2016.

[40] R. Kumaresan and I. Bentov. How to Use Bitcoin to Incentivize Correct Computations. In *CCS*, 2014.

[41] R. Kumaresan and I. Bentov. Amortizing secure computation with penalties. In *CCS*, 2016.

[42] H. Lin and R. Pass. Constant-round nonmalleable commitments from any one-way function. *J. ACM*, 62(1):5:1–5:30, 2015.

[43] H. Lin, R. Pass, and M. Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In *TCC*, 2008.

[44] H. K. Maji, M. Prabhakaran, and A. Sahai. On the computational complexity of coin flipping. In *FOCS*, 2010.

[45] A. Miller and I. Bentov. Zero-collateral lotteries in bitcoin and ethereum. In *EuroS&P Workshops*, 2017.

[46] T. Moran, M. Naor, and G. Segev. An optimally fair coin toss. *J. Cryptol.*, 29(3):491–513, July 2016.

[47] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[48] J. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2), 1951.

[49] S. J. Ong, D. C. Parkes, A. Rosen, and S. P. Vadhan. Fairness with an honest minority and a rational majority. In *TCC*, 2009.

[50] R. Pass, L. Seeman, and A. Shelat. Analysis of the blockchain protocol in asynchronous networks. In *Eurocrypt*, 2017.

[51] R. Pass and E. Shi. Rethinking large-scale consensus. In *CSF*, 2017.

[52] A. C.-C. Yao. Protocols for secure computations. In *FOCS*, 1982.

[53] A. C.-C. Yao. How to generate and exchange secrets. In *FOCS*, 1986.

A Detailed Proofs for Maximin Fairness

A.1 The Case of Two Parties

First, as mentioned, for two parties, maximin fairness equates to Blum's notion of weak fairness [16], assuming that the two parties have opposing preferences. Due to well-known results [33, 44], we immediately rule out the possibility of achieving statistical maximin fairness in this setting.

For semi-malicious adversaries, it is not difficult to see that the following simple protocol satisfies perfect maximin fairness: both parties choose a random coin and announce it, and the outcome is the XOR of the two coins. If either party aborts prior to broadcasting its bit, the outcome is the other party's preference. This protocol can be upgraded to achieve *computational* maximin fairness against malicious adversaries relying on cryptographic commitments [16]: first, Alice commits to a random bit; then Bob announces a random bit; and finally, Alice opens her commitment. Normally the outcome is the XOR of the two bits. If one party aborts, the outcome is the other party's preference. It suffices to use a standard commitment scheme that is perfectly binding and computationally hiding, and such a commitment scheme can be built from one-way functions [16, 26].

Therefore most of the paper focuses on three or more parties which is the more interesting and unexplored case.

A.2 Proof of Claim 1

We present the detailed proof for Claim 1: informally, for a protocol satisfies maximin fairness against arbitrarily many corruptions as long as it satisfies maximin fairness against all coalitions of size $n-1$.

Proof. It suffices to show that if a protocol Π is maximin fair against any coalition of size exactly $n-1$, then Π is maximin fair against any arbitrarily sized coalition (the reverse direction is obvious). We prove it below it for computational or statistical security — for perfect security, the argument can be easily modified by requiring negligible functions to be 0.

Consider any adversary $\mathcal{A}$ that controls a coalition $C \subset [n]$ of size at most $n-1$. Let $\overline{C} := [n] \backslash C$. We would like to show that for any honest party $i \in \overline{C}$, its expected reward cannot be less than $\frac{1}{2} - p(\kappa)$ for any polynomial $p(\cdot)$. We can view the execution with $\mathcal{A}$ alternatively as the following execution: i is the only honest party, and everyone else is corrupt. Specifically, those in C execute the same strategy as $\mathcal{A}$; for any $j \neq i$ and $j \notin C$, imagine that j is a corrupt party following the honest protocol. Now if Π is computationally or statistically maximin fair against such an $(n-1)$-sized coalition, we conclude that i's expected payoff in this randomized execution cannot be negligibly smaller than $\frac{1}{2}$. $\square$

A.3 Equivalence to Group Maximin Fairness

We define an alternative notion called group maximin fairness, and prove that it is equivalent to maximin fairness in the context of coin toss.

Definition 6 (Group maximin fairness)**.** Let $\mathfrak{A}$ be a family of adversaries that corrupt up to $n-1$ parties; and let $\mathcal{P} \in \{0,1\}^n$ be any divided preference profile. We say that an n-party coin toss protocol is is group maximin fair for $\mathcal{P}$ against the family $\mathfrak{A}$, iff for any adversary $\mathcal{A} \in \mathfrak{A}$, there exists some negligible function $\mathsf{negl}(\cdot)$ such that in an execution with the preference profile $\mathcal{P}$ and the adversary $\mathcal{A}$, the expected total reward for the set of all honest parties (denoted H) is at least $\sigma^{\mathcal{P}}(H) - \mathsf{negl}(\kappa)$ where $\sigma^{\mathcal{P}}(H)$ denotes the expected total payoff of the set H in an all-honest execution under the preference profile $\mathcal{P}$.

Just like in Definition 2, now depending on the family $\mathfrak{A}$ of adversaries that we are concerned about, we can define computational, statistical, or perfect notions for group maximin fairness, and for fail-stop, or malicious adversaries respectively. We omit the detailed definitions for conciseness.

Claim 2. *Let $\mathcal{P} \in \{0,1\}^n$ be any divided preference profile. An n-party coin toss protocol Π satisfies computational (or statistical, perfect resp.) maximin fairness for $\mathcal{P}$ against fail-stop (or malicious resp.) adversaries, iff Π satisfies computational (or statistical, perfect resp.) group maximin fairness for $\mathcal{P}$ against fail-stop (or malicious resp.) adversaries.*

Proof. The fact that maximin fairness implies group maximin fairness is obvious. For the reverse direction, observe that group maximin fairness implies maximin fairness against coalitions of size $n-1$ which implies maximin fairness against arbitrarily sized coalition by Claim 1. $\square$

A.4 Impossibility for Malicious Adversaries and Almost Unanimous Preference Profiles

In this section, we present the formal, detailed proof for Theorem 7.

Without loss of generality, we may assume that the three parties P_1, P_2, and P_3 prefer 1, 0, and 1 respectively. We may also assume that with probability 1, each party consumes at most $\ell(\kappa)$ bits of randomness in the protocol even in the presence of any possibly unbounded adversary. Henceforth we use the notation $T_1, T_2, T_3 \in \{0,1\}^{\ell(\kappa)}$ to denote the random bits consumed by parties P_1, P_2, and P_3 respectively.

At a very high level, our proof strategy is the following: we start by assuming that some protocol Π satisfies computational maximin fairness under semi-malicious corruptions, and assuming that the three parties have preferences for $1, 0, 1$ respectively. We then analyze such a protocol, and by the end of the proof we will have constructed an explicit polynomial-time attack against maximin fairness for this protocol. We thus reach a contradiction and conclude that no 3-party protocol can satisfy computational maximin fairness.

First, consider any fixed $T_2 \in \{0,1\}^{\ell(\kappa)}$. We use the notation $f(T_2)$ to denote the expected outcome when P_2 uses the randomness T_2 whereas P_1 and P_3 executed the protocol honestly.

Proof of Lemma 1. We now prove Lemma 1, i.e., almost all random coins of a lone semi-malicious P_2 are created equal. We restate the lemma here for the reader's convenience.

Lemma 2 (Restatement of Lemma 1)**.** *Suppose that the protocol under consideration satisfies computational maximin fairness. Then, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that except for* $\mathsf{negl}(\kappa)$ *fraction of* T_2*'s, it must be that that* $f(T_2) \in [0.5 - 1/p(\kappa), 0.5 + 1/p(\kappa)]$ *for any polynomial function* $p(\cdot)$*. In other words, except for* $\mathsf{negl}(\kappa)$ *fraction of* T_2*'s,* $|f(T_2) - 0.5|$ *is a negligible function in* κ*.*

Proof. First, we prove the following fact.

Fact 4. *For sufficiently large* κ *and any* $T_2 \in \{0,1\}^{\ell(\kappa)}$*,* $f(T_2) \geq 0.5 - 1/p(\kappa)$ *for any polynomial function* $p(\cdot)$*.*

Suppose that the above fact is not true for the sake of reaching a contradiction. Then, there must exist some $T_2^* \in \{0,1\}^{\ell(\kappa)}$ and some polynomial function $p(\cdot)$ such that $f(T_2^*) < 0.5 - 1/p(\kappa)$. Given this, we can construct a non-uniform, semi-malicious adversary that corrupts only P_2 and breaks maximin fairness for the honest parties P_1 and P_3. Specifically, the adversary is given the non-uniform advice string $T_2^* \in \{0,1\}^{\ell(\kappa)}$, and it makes P_2 use T_2^* instead of sampling an honest random string at random; otherwise P_2 follows the protocol faithfully.

Fact 5. *By correctness, under all-honest execution we have that* $\mathbf{E}_{T_2}(f(T_2)) = 1/2$*.*

We now proceed to prove Lemma 1. Suppose that Lemma 1 is not true for the sake of reaching a contradiction. Then, by Fact 4, there must exist some inverse-polynomial $1/p'(\kappa)$ fraction of T_2's such that $f(T_2) > 0.5 + 1/p(\kappa)$ for some polynomial $p(\cdot)$. Again by Fact 4, we have that for any polynomial $q(\cdot)$,

$$\begin{aligned}\mathbf{E}_{T_2}(f(T_2)) &\geq (0.5 - 1/q(\kappa)) \cdot (1 - 1/p'(\kappa)) + (0.5 + 1/p(\kappa)) \cdot 1/p'(\kappa) \\ &= 0.5 + \frac{1}{p \cdot p'} - \frac{1}{q} \cdot (1 - \frac{1}{p'})\end{aligned}$$

The above is greater than 0.5 for a sufficiently large polynomial $q(\cdot)$ — i.e., when $1/q$ is sufficiently small. This contradicts Fact 5. $\square$

Remark 2. Lemma 1 uses the fact that a lonely corrupt P_2 whose randomness can be arbitrarily programmed (but P_2 otherwise follows the honest protocol to the end) cannot bias the outcome of the coin toss. It will become clear later that this is the only place we rely on maximin fairness against *semi-malicious* adversaries in the entire proof of Theorem 7. The remaining proof of Theorem 7 relies only on the fact that the protocol is maximin fair against *fail-stop* adversaries.

Definition of bias. Given an adversary $\mathcal{A}$ that controls a coalition C, let $\overline{C}$ denote the remaining set of honest parties. We say that the adversary $\mathcal{A}$ causes μ-bias towards $b \in \{0, 1\}$ iff in an execution with $\mathcal{A}$,

$$\Pr[\overline{C}\text{'s outcome} = b] - \frac{1}{2} \geq \mu$$

We also sometimes say that $\mathcal{A}$ biases $\overline{C}$ towards b by μ.

We say that the adversary $\mathcal{A}$, controlling C, causes μ-bias in either direction iff

$$\max\left\{\Pr[\overline{C}\text{'s outcome} = 0], \quad \Pr[\overline{C}\text{'s outcome} = 1]\right\} - \frac{1}{2} \geq \mu$$

Lone-wolf and wolf-minion conditions. Now elaborating on Remark 2, the remainder of the proof will rely on the following conditions that are implied by maximin fairness against non-uniform p.p.t.*fail-stop* adversaries. Below although not explicitly noted, we assume that the adversary is non-uniform p.p.t..

- *Lone-wolf condition.* When P_1 (or P_3) is the only fail-stop party, it cannot cause non-negligible (in κ) bias towards either direction. Such an attack is also called a lone-wolf attack.
- *Wolf-minion condition.* When P_1 and P_2 (or P_2 and P_3) form a fail-stop coalition, they cannot cause non-negligible (in κ) bias towards 0. In fact we only care about attacks where P_2 is a silent accomplice (called a *minion* [24]) that never aborts but shares information with P_1 (or P_3); and P_1 (or P_3) may decide to abort depending on its view in the execution (called a *wolf* [24]). Such attacks are called wolf-minion attacks.

In the remainder of the proof, we will show that if Lemma 1 holds, and moreover, if both the lone-wolf condition and the wolf-minion condition hold, then we can construct an explicit, polynomial-time wolf-minion attack that creates non-negligible bias towards 1 — thus reaching a contradiction. Therefore we conclude finally that any n-party coin toss protocol cannot be (even computationally) maximin fair against semi-malicious adversaries.

Assumptions on message schedule. Without loss of generality, we may assume that the protocol's message schedule satisfies the following assumptions[10]. In the presence of any semi-malicious but non-aborting adversary (i.e., no corrupt party ever aborts during protocol execution), then, with probability 1,

- In the first round, P_1 is the only party that sends message;
- In each of rounds $2, 3, \ldots, R(\kappa)$, all parties send messages;
- In the last round $R(\kappa) + 1$, P_3 is the only party that sends message.

If a protocol satisfies the above message schedule requirement, we say that the protocol is *regular*. It is not difficult to see that if there is a non-regular, 3-party coin-toss protocol Π that satisfies maximin fairness, we can always construct a regular, 3-party coin toss protocol Π' that also satisfies maximin fairness and with only $O(1)$ additional rounds than Π: one can always insert a dummy first (or last) round where only P_1 (or P_3) sends message. Further, for every intermediate round, if a party does not want to send a message in that round by the rules of Π, it can always just send a dummy message.

[10]These assumptions are without loss of generality. As it will become clear soon, these assumptions allow us to conform to Cleve's message delivery assumptions when we apply Cleve [18] later in the proof.

A sequence of adversaries. We now define a sequence of polynomial-time adversaries as inspired by Cleve's famous lower bound [18]. However, different from Cleve [18], here we define the sequence of adversaries conditioned on any fixed choice of T_2, and moreover, assuming that the adversary corrupts either (P_1 and P_2), or (P_2 and P_3).

- Adversary $\mathcal{A}_i^b(1^\kappa, T_2)$ corrupts P_1 and P_2 and wants to bias P_3 towards b:
  - $\mathcal{A}_i^b$ uses the randomness T_2 that is provided as input to $\mathcal{A}_i^b$ for party P_2 rather than choosing a random T_2. $\mathcal{A}_i^b$ chooses a random T_1 for party P_1 honestly.
  - $\mathcal{A}_i^b$ now executes the honest protocol on behalf of P_1 and P_2, until the moment right before P_1 is going to broadcast its i-th message.
  - At this moment, $\mathcal{A}_i^b$ computes α_i, that is, imagine that P_3 aborted right after sending its $(i-1)$-th message[11], what would be the outcome of parties P_1 and P_2. Note that α_i is the outcome of P_1 and P_2 if P_3 aborted after sending the $(i-1)$-th message, i.e., P_3's message in round i.
  - If $\alpha_i = b$, then P_1 aborts after sending the i-th message; else P_1 aborts right now without sending the i-th message. In either case P_2 does not abort and plays to the end with P_3.
- Adversary $\mathcal{B}_i^b(1^\kappa, T_2)$: corrupts P_2 and P_3 and wants to bias P_1 towards b:
  - $\mathcal{B}_i^b$ uses the randomness T_2 that is provided as input to $\mathcal{B}_i^b$ for party P_2 rather than choosing a random T_2. $\mathcal{B}_i^b$ chooses a random T_3 for party P_3 honestly.
  - $\mathcal{B}_i^b$ now executes the honest protocol on behalf of P_2 and P_3, until the moment right before P_3 is going to broadcast its i-th message.
  - At this moment, $\mathcal{B}_i^b$ computes β_i, that is, imagine that P_1 aborted right after sending its i-th message, what would be the outcome of parties P_2 and P_3. Note that β_i is the outcome of P_2 and P_3 if P_1 aborted after sending the i-th message.
  - If $\beta_i = b$, then P_3 aborts after sending the i-th message; else P_3 aborts right now without sending the i-th message. In either case, P_2 does not abort and plays to the end with P_1.

By definition, we assume that any $\mathcal{A}_i^b(1^\kappa, T_2)$ or $\mathcal{B}_i^b(1^\kappa, T_2)$ can only be successful in biasing towards b but not $1-b$. Notice that in the above sequence of adversaries

$$\left\{\mathcal{A}_i^b(1^\kappa, T_2), \mathcal{B}_i^b(1^\kappa, T_2) : i \in [R(\kappa)], b \in \{0,1\}\right\},$$

the adversary always corrupts two parties, either (P_1 and P_2) or (P_2 and P_3). Among the two corrupt parties, P_2 always plays the role of a silent accomplice that never aborts, but the other corrupt party P_1 (or P_3) may abort based on certain decision criteria.

Finally, consider the following adversary:

- Adversary $\mathcal{A}_0(1^\kappa, T_2)$: corrupts P_1 and P_2 and wants to bias P_3 towards either direction. $\mathcal{A}_0^b(1^\kappa, T_2)$ programs P_2's randomness to be the T_2 that is provided as input. P_2 would otherwise follow the honest protocol to the end without aborting. However, P_1 aborts prior to sending any message at all.

[11]Note that here we make use of the fact that $\mathcal{A}_i^b$ is a rushing adversary.

A biasing adversary for almost all T_2's. If we consider P_2's randomness to be fixed to an arbitrary T_2, then the protocol execution depends only on the randomness T_1 and T_3. Lemma 1 says that the protocol execution (fixing any T_2) is a coin toss protocol between P_1 and P_3. Specifically, as long as P_1 and P_3 are honest, the expected outcome is negligibly different from 1/2 for any fixed T_2. Cleve [18] show that for any 2-party protocol that 1) generates an unbiased coin under an honest execution and 2) satisfies agreement under an honest execution; there exists an attack that breaks strong fairness. Now, we can apply Cleve's argument for the protocol execution under any fixed T_2.

More formally, fixing T_2 to be an arbitrary value, we can view the 3-party protocol as a "residual 2-party protocol": imagine that T_2 is hard-wired in P_1 and P_3's algorithms. Now, P_1 and P_3 each simulates a copy of P_2 and thus P_1 and P_3 can compute all messages P_2 wants to send. As a result, P_2 no longer needs to send any messages at all, and the only messages sent are by either P_1 or P_3 (and thus we can now remove the party P_2 from the protocol execution).

Earlier we defined the adversaries $\mathcal{A}_i^b(1^\kappa, T_2)$, $\mathcal{B}_i^b(1^\kappa, T_2)$, and $\mathcal{A}_0(1^\kappa, T_2)$ for the 3-party protocol. We can now define the counterparts of these adversaries in the residual 2-party protocol for any fixed T_2: basically $\mathcal{A}_i^b(1^\kappa, T_2)$ and $\mathcal{A}_0(1^\kappa, T_2)$ now only corrupt P_1 and $\mathcal{B}_i^b(1^\kappa, T_2)$ now only corrupts P_3. The following fact is not difficult to see:

Fact 6. *Fix T_2 to be any arbitrary string. Consider a sample path determined by (T_1, T_3) and an adversary playing $\mathcal{A}_i^b(1^\kappa, T_2)$, $\mathcal{B}_i^b(1^\kappa, T_2)$, or $\mathcal{A}_0(1^\kappa, T_2)$ in the 3-party protocol: suppose that the honest party's output is x in the 3-party protocol, then the honest party's output is also x in the residual 2-party protocol for the same sample path and when playing with the same adversary (but now the 2-party counterpart).*

Now Cleve [18]'s proof can be alternatively stated as follows:

Theorem 18 (Cleve [18])**.** *For any fixed T_2, in the residual 2-party protocol,*

1. *either one of adversaries in the set $\left\{\mathcal{A}_i^1(1^\kappa, T_2), \mathcal{B}_i^1(1^\kappa, T_2) : i \in R(\kappa)\right\} \cup \{\mathcal{A}_0(1^\kappa, T_2)\}$ can cause $\frac{1}{2(4R(\kappa)+1)}$-bias towards 1; or*

2. *one of adversaries in the set $\left\{\mathcal{A}_i^0(1^\kappa, T_2), \mathcal{B}_i^0(1^\kappa, T_2) : i \in R(\kappa)\right\} \cup \{\mathcal{A}_0(1^\kappa, T_2)\}$ can cause $\frac{1}{2(4R(\kappa)+1)}$-bias towards 0*

where $R(\kappa)$ denotes the round complexity of the protocol.

Now for almost all T_2's, none of these adversaries cannot cause non-negligible bias towards 0 since otherwise we can construct a successful wolf-minion attack that creates non-negligible bias towards 0. We formalize this in the following lemma.

Lemma 3. *There exists a negligible function $\mathsf{negl}(\cdot)$ such that for all but $\mathsf{negl}(\kappa)$ fraction of T_2's, the following hold:*

1. *At least one adversary in the set $\left\{\mathcal{A}_i^1(1^\kappa, T_2), \mathcal{B}_i^1(1^\kappa, T_2) : i \in R(\kappa)\right\} \cup \{\mathcal{A}_0(1^\kappa, T_2)\}$ must successfully cause $\frac{1}{2(4R(\kappa)+1)}$- bias towards 1.*

2. *No adversary in the set $\left\{\mathcal{A}_i^b(1^\kappa, T_2), \mathcal{B}_i^b(1^\kappa, T_2) : b \in \{0,1\}, i \in R(\kappa)\right\} \cup \{\mathcal{A}_0(1^\kappa, T_2)\}$ causes $1/p(\kappa)$-bias towards 0 for any polynomial $p(\cdot)$.*

Proof. Suppose that the second claim is not true. Then, there must exist some

$$\mathfrak{a}(1^\kappa, \cdot) \in \left\{\mathcal{A}_i^b(1^\kappa, \cdot), \mathcal{B}_i^b(1^\kappa, \cdot) : b \in \{0,1\}, i \in R(\kappa)\right\} \cup \{\mathcal{A}_0(1^\kappa, \cdot)\}$$

such that $\mathfrak{a}(1^\kappa, \cdot)$ can cause $p(\kappa)$-bias towards 0 on $1/q(\kappa)$ fraction of the T_2's for some polynomial $p(\cdot)$ and $q(\cdot)$. We now construct the following fail-stop (non-uniform) adversary denoted $\mathcal{A}^*$ that breaks maximin fairness (and specifically it breaks the wolf-minion condition). The adversary $\mathcal{A}^*$ is given the following (non-uniform) advice: i) which adversary $\mathfrak{a}$ is successful in causing bias towards 0; and ii) the polynomial $p(\cdot)$. Depending on $\mathfrak{a}$, the adversary corrupts either (P_1 and P_2) or (P_2 and P_3).

Now, $\mathcal{A}^*$ picks a random T_2 for party P_2. It will first check whether the chosen T_2 is a good T_2 for $\mathfrak{a}$ (a good T_2 helps $\mathfrak{a}$ create bias towards 0). To do this, $\mathcal{A}^*$ repeats the following $p^2(\kappa)$ times: sample a random T_1 and T_3 and simulate the protocol execution playing with $\mathfrak{a}$. If the outcome is 0 more than $\frac{1}{2} + \frac{1}{2p(\kappa)}$ fraction of the time, determine T_2 to be good; else determine T_2 to be bad. By a simple application of the Chernoff bound, the following claim must hold — henceforth we say that $g(T_2) = \nu$ iff fixing T_2 but sampling T_1 and T_3 at random, and running the protocol with $\mathfrak{a}$, the probability that the outcome is 0 is ν.

Claim 3. *If $g(T_2) \geq \frac{1}{2} + \frac{1}{p(\kappa)}$ then except with negligible in κ probability in the above experiment, T_2 is determined to be good. If $g(T_2) \leq \frac{1}{2}$ then except with negligible in κ probability in the above experiment, T_2 is determined to be bad.*

Now, if T_2 is good, $\mathcal{A}^*$ runs $\mathfrak{a}$; else it follows the honest protocol. It is not difficult to see that $\mathcal{A}^*$ can cause $1/p'(\kappa)$-bias towards 0 for some polynomial $p'(\cdot)$.

So far we have proved that the second claim is true. If so, the first claim also follows from Theorem 18. This concludes the proof of Lemma 3. □

Remark 3. We note that Lemma 3 would be much easier to prove if we relied on maximin fairness against *semi-malicious* adversaries. However, here we present a stronger proof: besides relying on Lemma 1 and Theorem 18, we additionally use only the fact that the protocol is maximin fair against *fail-stop* adversaries (and specifically the wolf-minion condition) — this stronger proof is reusable later in the paper for proving impossibilities for cooperative-strategy-proof fairness (Section 5.3).

We now construct the following *randomizing adversary* $\mathcal{X}(1^\kappa, T_2)$ and $\mathcal{Y}(1^\kappa, T_2)$ for any fixed T_2:

- $\mathcal{X}(1^\kappa, T_2)$ corrupts P_1 and P_2. It picks a random adversary from the set

$$S_A(T_2) := \left\{\mathcal{A}_i^1(1^\kappa, T_2) : i \in [R(\kappa)]\right\} \cup \{\mathcal{A}_0(1^\kappa, T_2)\}$$

  and follows the strategy of this randomly chosen adversary.

- $\mathcal{Y}(1^\kappa, T_2)$ corrupts P_2 and P_3. It picks a random adversary from the set

$$S_B(T_2) := \left\{\mathcal{B}_i^1(1^\kappa, T_2) : i \in [R(\kappa)]\right\}$$

  and follows the strategy of this randomly chosen adversary.

Lemma 4. *There exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all but* $\mathsf{negl}(\kappa)$ *fraction of T_2's, either $\mathcal{X}(1^\kappa, T_2)$ successfully causes $1/p(\kappa)$-bias towards 1 for some polynomial $p(\cdot)$, or $\mathcal{Y}(1^\kappa, T_2)$ successfully causes $1/p(\kappa)$-bias towards 1 for some polynomial $p(\cdot)$.*

Proof. Follows directly from Lemma 3, the definition of $\mathcal{X}(1^\kappa, T_2)$ and $\mathcal{Y}(1^\kappa, T_2)$, and the fact that $R(\cdot)$ is a polynomial function. $\square$

So far, for almost all T_2's, we have constructed an adversary that corrupts either (P_1 and P_2) or (P_2 and P_3), and biases the remaining honest party towards 1. Such an adversary, however, does not break maximin fairness because 1 is precisely what the remaining honest party prefers! To actually construct an adversary that causes harmful bias to the remaining honest P_1 or P_3, we consider the world when one takes average over the choice of T_2.

Averaging over T_2. Consider the following adversaries $\overline{\mathcal{X}}(1^\kappa)$ and $\overline{\mathcal{Y}}(1^\kappa)$ who chooses a random T_2 rather than receives a specific T_2 as input:

- $\overline{\mathcal{X}}(1^\kappa)$ (or $\overline{\mathcal{Y}}(1^\kappa)$ resp.) picks a random T_2 honestly; it then plays the strategy of $\mathcal{X}(1^\kappa, T_2)$ (or $\mathcal{Y}(1^\kappa)$ resp.), i.e., corrupting the parties $\mathcal{X}(1^\kappa, T_2)$ (or $\mathcal{Y}(1^\kappa)$ resp.) wants to corrupt and follows how $\mathcal{X}(1^\kappa, T_2)$ (or $\mathcal{Y}(1^\kappa)$ resp.) interacts with the remaining honest party.

Claim 4. *Either $\overline{\mathcal{X}}(1^\kappa)$ causes $1/p(\kappa)$-bias towards 1 for some polynomial $p(\cdot)$, or $\overline{\mathcal{Y}}(1^\kappa)$ causes $1/p(\kappa)$-bias towards 1 for some polynomial $p(\cdot)$.*

Proof. Follows directly from Lemma 4 and the definition of $\overline{\mathcal{X}}(1^\kappa)$ and $\overline{\mathcal{Y}}(1^\kappa)$. $\square$

It is not difficult to see that an alternative but equivalent way to view the adversaries $\overline{\mathcal{X}}(1^\kappa)$ and $\overline{\mathcal{Y}}(1^\kappa)$ is the following. First, we define a sequence of $4R(\kappa)+1$ adversaries analogous to those in the set $S(T_2)$ but now, the $4R(\kappa)+1$ adversaries are not given a T_2 as input, but rather, chooses a T_2 at random.

- Adversary $\overline{\mathcal{A}}_i^b(1^\kappa)$ where $i \in [R(\kappa)]$ and $b \in \{0,1\}$: picks a random T_2 and runs $\mathcal{A}_i^b(1^\kappa, T_2)$. Note that $\overline{\mathcal{A}}_i^b(1^\kappa)$ corrupts P_1 and P_2 among whom only P_1 may abort and P_2 is a silent accomplice.
- Adversary $\overline{\mathcal{B}}_i^b(1^\kappa)$ where $i \in [R(\kappa)]$ and $b \in \{0,1\}$: picks a random T_2 and runs $\mathcal{B}_i^b(1^\kappa, T_2)$. Note that $\overline{\mathcal{B}}_i^b(1^\kappa)$ corrupts P_2 and P_3 among whom only P_1 may abort and P_2 is a silent accomplice.
- Adversary $\overline{\mathcal{A}}_0(1^\kappa)$: picks a random T_2 and runs $\mathcal{A}_0(1^\kappa, T_2)$.

Now, we can equivalently view $\overline{\mathcal{X}}(1^\kappa)$ as the adversary that picks a random adversary from the following set $\overline{S}_A$ and executes its strategy:

$$\overline{S}_A := \left\{\overline{\mathcal{A}}_i^1(1^\kappa) : i \in [R(\kappa)]\right\} \cup \{\overline{\mathcal{A}}_0(1^\kappa)\}$$

Similarly, we can equivalently view $\overline{\mathcal{Y}}(1^\kappa)$ as the adversary that picks a random adversary the following set $\overline{S}_B$ and executes its strategy:

$$\overline{S}_B := \left\{\overline{\mathcal{B}}_i^1(1^\kappa) : i \in [R(\kappa)]\right\}$$

Lemma 5. *Either one of the adversaries in the set $\overline{S}_A \backslash \{\overline{\mathcal{A}}_0(1^\kappa)\}$ can cause $1/p(\kappa)$-bias towards 1 for some polynomial function $p(\cdot)$ or one of the adversaries in the set $\overline{S}_B$ can cause $1/p(\kappa)$-bias towards 1 for some polynomial function $p(\cdot)$.*

Proof. Due to Claim 4 and by the above alternative interpretation of $\overline{\mathcal{X}}(1^\kappa)$ and $\overline{\mathcal{Y}}(1^\kappa)$, it suffices to prove that $\overline{\mathcal{A}}_0(1^\kappa)$ cannot cause non-negligible bias towards either direction. To see this, observe that in an execution with $\overline{\mathcal{A}}_0(1^\kappa)$, it is as if P_2 is honest and P_1 is the lone wolf that always aborts prior to sending any message at all. It follows from the lone-wolf condition that $\overline{\mathcal{A}}_0(1^\kappa)$ cannot cause non-negligible bias towards either direction. $\square$

An adversary that causes harmful bias. Finally, we show that if some $\overline{\mathcal{A}}_i^1$ can cause large bias towards 1, then its mirroring adversary $\overline{\mathcal{A}}_i^0$ (i.e., flipping its choice whether to abort prior to or after sending the i-th message) must cause large bias towards 0; and such bias would indeed be harmful towards the remaining honest party P_3. Similarly, if some $\overline{\mathcal{B}}_i^1$ can cause large bias towards 1, then its mirroring adversary $\overline{\mathcal{B}}_i^0$ must cause large bias towards 0 too.

More formally, we prove the following Lemma 6. Note that if Lemma 6 holds, then we can conclude the proof of Theorem 7 since we will have identified an adversary that corrupts either (P_1 and P_2) or (P_2 and P_3), and causes non-negligible and harmful bias for the remaining honest party. This breaks the notion of (computational) maximin fairness — more specifically it violates the wolf-minion condition and results in a contradiction (recall that our proof assumes that the protocol under consideration is maximin fair).

Lemma 6. *For $i \in [R(\kappa)]$, if $\overline{\mathcal{A}}_i^1$ (or $\overline{\mathcal{B}}_i^1$ resp.) can cause $\mu(\kappa)$-bias towards 1, then $\overline{\mathcal{A}}_i^0$ (or $\overline{\mathcal{B}}_i^0$ resp.) can cause $(\mu(\kappa) - \mathsf{negl}(\kappa))$-bias towards 0 for some negligible function $\mathsf{negl}(\cdot)$.*

Proof. We prove the lemma for $\overline{\mathcal{A}}_i^1$ and $\overline{\mathcal{A}}_i^0$ since the argument is symmetric for $\overline{\mathcal{B}}_i^1$ and $\overline{\mathcal{B}}_i^0$.

A sample path is defined by the choice of T_1, T_2, and T_3. Let Q denote the set of sample paths for which $\overline{\mathcal{A}}_i^1$ decides to abort *before* sending the i-th message. Let $\overline{Q}$ denote the set of sample paths for which $\overline{\mathcal{A}}_i^1$ decides to abort *after* sending the i-th message. By definition, $\overline{\mathcal{A}}_i^0$ would abort *after* sending the i-th message on Q and abort *before* sending the i-th message on $\overline{Q}$.

- For $b \in \{0,1\}$, we define a partitioning of Q: let $Q = U_0^{\langle b\rangle} \cup U_1^{\langle b\rangle}$ where $U_0^{\langle b\rangle}$ is the set of sample paths for which P_3's outcome is 0 when playing with $\overline{\mathcal{A}}_i^b$, and U_1 is the set of sample paths for which P_3's outcome is 1 when playing with $\overline{\mathcal{A}}_i^b$.

- Similarly, for $b \in \{0,1\}$, we define a partitioning of $\overline{Q}$: let $\overline{Q} = \overline{U}_0^{\langle b\rangle} \cup \overline{U}_1^{\langle b\rangle}$ where $\overline{U}_0^{\langle b\rangle}$ is the set of sample paths for which P_3's outcome is 0 when playing with $\overline{\mathcal{A}}_i^b$, and $\overline{U}_1$ is the set of sample paths for which P_3's outcome is 1 when playing with $\overline{\mathcal{A}}_i^b$.

Now, consider a hybrid adversary that takes $\overline{\mathcal{A}}_i^1$'s decisions on Q and takes $\overline{\mathcal{A}}_i^0$'s decisions on $\overline{Q}$. Such an adversary effectively always makes P_1 abort *before* sending the i-th message. Since this adversary also chooses T_2 at random, effectively an execution with this hybrid adversary can be viewed as if only a single party P_1 is corrupt and P_2 is honest. Now by the lone-wolf condition, P_3's outcome must not be biased towards either direction (except by negligibly small amounts). We conclude that

$$U_0^{\langle 1\rangle} + \overline{U}_0^{\langle 0\rangle} \approx U_1^{\langle 1\rangle} + \overline{U}_1^{\langle 0\rangle}$$

where $x(\kappa) \approx y(\kappa)$ means that $|x(\kappa) - y(\kappa)| \leq \mathsf{negl}(\kappa)$ for some negligible function $\mathsf{negl}(\cdot)$. By a symmetric argument and considering a P_1 that always aborts *after* sending the i-th message, we have that

$$U_0^{\langle 0\rangle} + \overline{U}_0^{\langle 1\rangle} \approx U_1^{\langle 0\rangle} + \overline{U}_1^{\langle 1\rangle}$$

We thus conclude that

$$(U_0^{\langle 0\rangle} + \overline{U}_0^{\langle 0\rangle}) - (U_1^{\langle 0\rangle} + \overline{U}_1^{\langle 0\rangle}) \approx (U_1^{\langle 1\rangle} + \overline{U}_1^{\langle 1\rangle}) - (U_0^{\langle 1\rangle} + \overline{U}_0^{\langle 1\rangle})$$

The proof concludes by observing that the above means that $\overline{\mathcal{A}}_i^0$'s ability to bias towards 0 is negligibly different from $\overline{\mathcal{A}}_i^1$'s ability to bias towards 1. □

A.5 Impossibility for Amply Divided Preference Profiles

In an amply divided preference profile, at least two parties prefer 0 and two parties prefer 1. We can show strong impossibility in this case, that is, no n-party coin toss protocol can achieve even computational maximin fairness against even fail-stop adversaries. The proof follows by a simple reduction to Cleve's lower bound of strongly fair, 2-party coin toss [18] as we explain below.

Proof of Theorem 5.

Proof. Imagine that we divide all n parties into two disjoint partitions denoted A and B respectively, such that each partition contains at least one party who prefers 0 and one party who prefers 1. Now in an n-party coin toss protocol, imagine the family (denoted $\mathfrak{A}$) of all non-uniform p.p.t. fail-stop adversaries that either control all nodes in A or control all nodes in B. Further, all nodes in the corrupt coalition (either A or B) act in accordance in any round: they either all abort or all continue. By the definition of maximin fairness, no adversary in $\mathfrak{A}$ can cause any non-negligible bias to the outcome of the protocol.

Now, we can construct a 2-party coin-toss protocol where one party simulates all parties inside the partition A and another party simulates all parties inside B. We abuse notation and use A and B to denote these two nodes. Now, every adversary in $\mathfrak{A}$ naturally maps an adversary in the 2-party protocol, where either A or B is a fail-stop party that may (or may not) decide to abort prematurely depending on the prefix of the execution (possibly after having observed the messages the other party wants to send in the same round, i.e., through a rushing attack). We conclude that no non-uniform p.p.t. fail-stop adversary can cause non-negligible bias in this 2-party coin toss protocol — but this violates Cleve's lower bound on strongly fair coin toss [18]. □

B Additional Details for CSP Fairness

Earlier in Section 5.2, we described a protocol that achieves CSP fairness for almost unanimous preference profiles in the presence of malicious adversaries; however, for simplicity we assumed an ideal commitment functionality $\mathcal{F}_{\text{idealcomm}}$. In this section, we elaborate on how to remove the need for $\mathcal{F}_{\text{idealcomm}}$, and instantiate a protocol using special concurrent non-malleable commitments that enjoy public verifiability (which in turn can be instantiated assuming the existence of one-way permutations [42]).

Then, we provide a full proof of Theorem 10, i.e., the impossibility of achieving CSP fairness for amply divided preference profiles.

B.1 Preliminary: Publicly Verifiable Concurrent Non-Malleable Commitment

A publicly verifiable commitment scheme $(\mathsf{C}, \mathsf{R}, \mathsf{V})$ consists of a pair of interacting Turing machines called the committer C and the receiver R respectively, and a deterministic, polynomial-time public audit function denoted V. Suppose the commitment protocol completes successfully and produces some transcript Γ (which includes an ordered sequence of all bits transmitted between C and R), then $\mathsf{V}(\Gamma)$ outputs either a bit $b \in \{0,1\}$ to accept or $\perp$ which indicates rejection. If a bit $b \in \{0,1\}$ is output (and not $\perp$), we call it the *accepting bit*. Henceforth we assume that the protocol has two phases, a commitment phase and an opening phase, and that all algorithms receive a security parameter κ as input. Henceforth we often use the notation $\langle \mathsf{C}^*(z), \mathsf{R}^*(z') \rangle$ to indicate a (possibly randomized) execution between C^* that is invoked with the input z and R^* that is invoked with the input z'.

Perfectly correct. We require a strong notion of correctness, that is, for either $b \in \{0, 1\}$, and for any $\kappa \in \mathbb{N}$, if C is honest and receives the input bit b, then for any (possibly unbounded) non-aborting R^*, with probability 1, the execution $\langle \mathsf{C}(1^\kappa, b), \mathsf{R}^*(1^\kappa)\rangle$ will successfully complete with the accepting bit b. Note that here we assume that if the malicious receiver R^* sends malformed messages outside the appropriate range, it is treated the same way as aborting. This notion of correctness implies that an honest sender can always complete the protocol correctly opening its bit, and an arbitrarily malicious (non-aborting) receiver cannot cause it to be stuck.

Perfectly binding. We can denote the transcript as $\Gamma := (\Gamma_0, \Gamma_1) \in \{0, 1\}^{\ell(\kappa)}$ where the former term denotes the transcript by the end of the commitment phase and the latter term denotes the transcript of the opening phase, and $\ell(\cdot)$ is a fixed polynomial function in κ that denotes the maximum length of the transcript in each phase. We require that for any $\kappa \in \mathbb{N}$, any $\Gamma_0, \Gamma_1, \Gamma_1 \in \{0, 1\}^{\ell(\kappa)}$, if $\mathsf{V}(1^\kappa, \Gamma_0, \Gamma_1) = b$ and $\mathsf{V}(1^\kappa, \Gamma_0, \Gamma_1') = b'$ where $b, b' \in \{0, 1\}$, then it must be that $b = b'$. In other words, the transcript by the end of the commitment phase determines at most one bit that can be successfully opened (even when both the committer and receiver are corrupt and unbounded).

Computationally hiding. Henceforth let $p(1^\kappa, v)$ denote the probability that R^* outputs 1 at the end of the commitment phase in the execution $\langle \mathsf{C}(1^\kappa, v), \mathsf{R}^*\rangle$ where C runs the honest committer. We say that a commitment scheme is computationally hiding, iff for every non-uniform p.p.t. R^*, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\kappa \in \mathbb{N}$, for every $v_1, v_2 \in \{0, 1\}^\kappa$, it must hold that $|p(1^\kappa, v_1) - p(1^\kappa, v_2)| \leq \mathsf{negl}(\kappa)$.

Concurrent non-malleability. We use the definition of concurrent non-malleability by Lin et al. [43] — note that this notion implies computationally hiding. To define concurrent non-malleability, we will consider a man-in-the-middle adversary $\mathcal{A}$ that participates in m left interactions and m right interactions: on the left it interacts with an honest committer who runs the commitment phase of the protocol and commits to values $v_1, \ldots, v_m$ using identities $\mathsf{id}_1, \ldots, \mathsf{id}_m$; on the right $\mathcal{A}$ interacts with an honest receiver attempting to commit to a sequence of values $v_1', \ldots, v_m'$, again using identities of its choice $\mathsf{id}_1', \ldots, \mathsf{id}_m'$. If any of the right commitments are invalid its value is set to $\perp$. For any $i \in [m]$, if $\mathsf{id}_i' = \mathsf{id}_j$ for some $j \in [m]$, v_i' is set to $\perp$. Now, let $\mathsf{mitm}^{\mathcal{A}}(1^\kappa, v_1, \ldots, v_m, z)$ denote a random variable that describes the values $v_1', \ldots, v_m'$ and the view of $\mathcal{A}$ in the above experiment — note that $v_1', \ldots, v_m'$ are well-defined if the commitment is perfectly binding.

Definition 7 (Concurrent non-malleability)**.** A commitment scheme is said to be concurrent non-malleable (w.r.t. commitment) if for every polynomial $p(\cdot)$ and every non-uniform p.p.t. adversary $\mathcal{A}$ that participates in at most $m = p(\kappa)$ concurrent executions, there exists a polynomial-time simulator $\mathcal{S}$ such that the following ensembles are computationally indistinguishable:

$$\left\{\mathsf{mitm}^{\mathcal{A}}(1^\kappa, v_1, \ldots, v_m, z)\right\}_{v_1,\ldots,v_m \in \{0,1\}^\kappa, \kappa\in\mathbb{N}, z\in\{0,1\}^*} \quad \text{and} \quad \left\{\mathcal{S}(1^\kappa, z)\right\}_{v_1,\ldots,v_m \in \{0,1\}^\kappa, \kappa\in\mathbb{N}, z\in\{0,1\}^*}$$

Theorem 19 (Publicly verifiable concurrent non-malleable commitment [42])**.** *Assume that one-way permutations exist. Then there exists a publicly verifiable commitment scheme that is perfectly correct, perfectly binding, and concurrent non-malleable.*

Proof. Lin and Pass [42] construct a concurrent non-malleable commitment scheme — it is not difficult to see that their scheme satisfies public verifiability and perfect correctness. □

B.2 CSP Fairness For Almost Unanimous Preference Profiles

Let $(\mathsf{C}, \mathsf{R}, \mathsf{V})$ be a publicly verifiable commitment scheme that is perfectly correct, perfectly binding, computationally hiding, and concurrent non-malleable as defined in Section B.1. Without loss of generality, suppose that there are $n \geq 3$ players, and only one of them is a 0-supporter, the remaining are 1-supporters. Now consider the following protocol where we assume that all commitment instances have unique session identifiers:

1. Phase 0: Everyone player $i \in [n]$ a random bit $b_i \in \{0, 1\}$, and it invokes a separate commitment instance for every $j \neq i$, and commits to b_i to player j.
2. Phase 1: The single 0-supporter opens all of its commitments.
3. Phase 2: Every 1-supporter opens all of its commitments.
4. Phase 3: If for some party, in all of its commitment instances (where it acts as committer) the receiver aborted prematurely, this party (who must be honest) simply broadcasts the bit it wanted to commit to — and this bit is considered as a valid opening for this party.
5. Outcome: At the end of the execution, if any player has aborted, it is *disqualified*. For any player that is not disqualified yet, consider all of its commitment instances (where it acts as the committer) in which the corresponding receiver did not abort: unless all these instances opened to the same bit the player is *disqualified* as well.

   The outcome of the protocol is defined as follows: if any 1-supporter is disqualified, the outcome is defined to be 0; else, the output is the XOR of the unique bit that has been opened for all players that have not been disqualified. Note that for any player that has not been disqualified, its opening is uniquely defined: either in some instances (where the player is committer) the receiver did not abort and all instances open to the same bit; or in all instances the receiver aborted in which case the opening is the bit the party announces in Phase 3.

Theorem 20 (Computational CSP fairness against malicious adversaries)**.** *Assume that one-way permutations exist, then for any $n \geq 3$, there exists an n-party protocol that achieves computational CSP fairness for any almost unanimous preference profile $\mathcal{P} \in \{0, 1\}^n$ against malicious coalitions of size up to $n - 1$.*

Proof. Consider the protocol described above. Correctness follows trivially. We now prove CSP fairness. We number the parties from 0 to $n - 1$, and w.l.o.g. let 0 be the single 0-supporter. Consider the following cases:

1. *The single 0-supporter is corrupt and all 1-supporters are honest*: Due to the perfect correctness of the commitment scheme, the coin toss outcome depends only on the 0-supporter's actions before the beginning of Phase 2, and does not depend on its actions after Phase 2. Let $o_1, \ldots, o_{n-1}$ denote the $n - 1$ values it opened before the begin of Phase 2 (recall that any instance in which the committer aborted prior to opening, we define the opening to be $\perp$). Let $b_1, \ldots, b_{n-1}$ denote all honest parties' random coin tosses that they commit to in Phase 1. Then, the outcome of the execution can be thought of as $f(o_1, \ldots, o_{n-1}) \oplus b_1 \oplus b_2, \ldots, \oplus b_{n-1}$ where f is a function that we do not care. Had $(o_1, \ldots, o_{n-1})$ been picked from a joint distribution that is independent of $b_1 \oplus b_2, \ldots, \oplus b_{n-1}$, then the expected outcome obviously would be $\frac{1}{2}$ (where probability is taken over the randomness of the simulator and honest parties). The proof concludes by observing that due to the concurrent non-malleability and the perfect binding properties of the commitment

scheme, for any fixed $b_1, b_2, \ldots, b_{n-1}$ input to the honest committers, $(o_1, \ldots, o_{n-1})$ is computationally indistinguishable from the output of a simulator that is unaware of $b_1, b_2, \ldots, b_{n-1}$[12].

2. *A single 0-supporter and a single 1-supporter are corrupt*: For this case the definition is trivially satisfied since the corrupt coalition is indifferent to the outcome.

3. *The single 0-supporter is honest and one or more 1-supporters are corrupt.* Let C denote the corrupt coalition, let $\mathbf{b}_C := \{b_i\}_{i \in C}$ denote the vector of bits committed to by the corrupt coalition by the end of Phase 1 — specifically each b_i where $i \in C$ is defined as follows:

   - Consider all instances where i is the committer and the receiver did not abort by Phase 1.
   - If in all these instances the committer did not abort, and the committed values are the same bit $b \in \{0, 1\}$ and not $\perp$, then $b_i := b$; else b_i is defined to be $\perp$ (note that we implicitly use perfectly binding here).

   Let $\mathbf{b}_{-C}$ denote the bits the honest parties want to commit to.

   Now, conditioned on $\mathbf{b}_C$ and $\mathbf{b}_{-C}$ which are well-defined by the end of Phase 1 due to perfect binding, with probability 1, the outcome cannot be greater than $\oplus_{i \in \{0,\ldots,n-1\}} b_i$ where if $b_i = \perp$ it is treated as 0 in the XOR computation; this arises due to the following: 1) all honest committers can successfully open due to perfect correctness (if in all instances where the player is committer the corresponding receiver aborted, the opening is the bit announced in Phase 3); and 2) if for some $i \in C$, $b_i \neq \perp$, then if i aborts without decommitting after Phase 1, the outcome is 0.

   By the concurrent non-malleability of the commitment scheme, for any fixed $\mathbf{b}_{-C}$ input to the honest parties's commitments, $\mathbf{b}_C$ is computationally indistinguishable from what a simulator unaware of $\mathbf{b}_{-C}$ would have output. Now had $\mathbf{b}_C$ been output by this simulator, $\mathbf{b}_C$ would be independent of $\mathbf{b}_{-C}$ and thus $\oplus_{i \in \{0,\ldots,n-1\}} b_i$ would have expected value $\frac{1}{2}$ (recall that we treat $\perp$ as 0 in the XOR), where probability is taken over the randomness of the honest parties as well as that of the simulator.

4. *The single 0-supporter and at least two 1-supporters are corrupt.* In this case it must be that $n \geq 4$. Consider all commitment instances where the single 0-supporter is the committer and let $(o_1, \ldots, o_{n-1})$ denote the $n-1$ values opened before the begin of Phase 2 (recall that any instance in which the committer aborted prior to opening, we define the opening to be $\perp$). Let o be $\perp$ if not all of $(o_1, \ldots, o_{n-1})$ are the same or if some of them are $\perp$; else let o be the consistent bit defined by $(o_1, \ldots, o_{n-1})$. Let S denote the set of 1-supporters that are corrupt. Let $\mathbf{b}_S$ be defined similarly as the earlier definition of $\mathbf{b}_C$ (see Case 3) but now for the set S. Let H denote the set of honest players, and let $\mathbf{b}_H$ denote the vector of bits honest players want to commit to.

   Now, conditioned on $\mathbf{b}_S$, $\mathbf{b}_H$, and $(o_1, \ldots, o_{n-1})$, with probability 1, the outcome cannot be greater than the XOR of all of $\mathbf{b}_S$, $\mathbf{b}_H$, and o where $\perp$ is treated as 0 in the XOR. Due to the concurrent non-malleability and the perfect binding properties of the commitment scheme, for any fixed $\mathbf{b}_H$ input to the honest parties's commitments, $(o_1, \ldots, o_{n-1}, \mathbf{b}_S)$ must be computationally indistinguishable from what a simulator would have output not knowing $\mathbf{b}_H$. Now imagine $(o_1, \ldots, o_{n-1}, \mathbf{b}_S)$ were indeed output by such a simulator, then it must be that $(o_1, \ldots, o_{n-1}, \mathbf{b}_S)$ is independent of $\mathbf{b}_H$. The proof follows by observing that the XOR of all of $\mathbf{b}_S$, $\mathbf{b}_H$, and o

[12]Without loss of generality, we may assume that the opening phase has only a single message where the committer broadcasts its committed bit and randomness. Thus the adversary must make a decision whether to abort in Phase 1 based on its view in Phase 0.

has expectation $\frac{1}{2}$, where probability is taken over randomness of the simulator and of honest parties.

$\square$

B.3 Impossibility for Amply Divided Preference Profiles

We restate Theorem 10 for the reader's convenience.

Theorem 21 (Restatement of Theorem 10: impossibility of CSP-fairness for $n \geq 4$)**.** *Let $n \geq 4$, and let $\mathcal{P} \in \{0,1\}^n$ be any amply divided preference profile. Then, no n-party coin-toss protocol can achieve even computational CSP-fairness for $\mathcal{P}$, against even fail-stop adversaries.*

Proof. For $n = 4$, any divided preference profile $\mathcal{P} \in \{0,1\}^4$ must be balanced, and in this case CSP-fairness is equivalent to maximin fairness. Thus the case of $n = 4$ is implied by Theorem 5. Similarly, for any $n \geq 6$ and for any balanced preference profile $\mathcal{P} \in \{0,1\}^6$, the theorem also follows from Theorem 5. Henceforth we focus on the case when $n \geq 5$ and $\mathcal{P} \in \{0,1\}^n$ is amply divided but unbalanced. Without loss of generality, we will assume that more players prefer 1 than 0.

Recall that we refer to a party that prefers 1 as a 1-supporter and we refer to one that prefers 0 as a 0-supporter.

Viewing it as a 3-party protocol. Our overall proof strategy is similar to that of Theorem 7 for maximin-fairness. We will divide the parties into three disjoint partitions called P_1, P_2, and P_3 respectively. The division algorithm guarantees the following:

1. For either $b \in \{0,1\}$, the number of b-supporters in P_1 is the same as the number of b-supporters in P_3;

2. The number of 0-supporters in P_1 is the same as the number of 1-supporters in P_1 and the same holds for P_3; and

3. P_2 has at most one 0-supporter, and at least one 1-supporter. Further P_2 has more 1-supporters than 0-supporters.

In other words, we are evenly dividing the 0-supporters among P_1 and P_3. In case the total number of 0-supporters is odd, the leftover 0-supporter is placed in P_2. We then place as many 1-supporters in each of P_1 and P_3 as the number of 0-supporters in either of these partitions. All the remaining 1-supporters get placed in P_2. Since by our assumption, there are more 1-supporters than there are 0-supporters, there is at least one 1-supporter in P_2; and further, the 1-supporters strictly outnumber 0-supporters in P_2.

We now regard all the parties in the same partition as a supernode, such that in any round they act in accordance: either jointly abort or jointly continue. In this way, we can view the protocol as a 3-party protocol between P_1, P_2, and P_3.

Lone-wolf and wolf-minion conditions. In this 3-party protocol, we make the following observation.

Claim 5 (Lone-wolf and wolf-minion conditions)**.** *In this 3-party protocol, lone-wolf condition and the wolf-minion condition (see the proof of Theorem 7 for definitions) both hold.*

Proof. The wolf-minion condition follows directly from the definition of CSP-fairness against fail-stop adversaries. We now prove that the lone-wolf condition also holds. We prove it for the case when P_1 is the lone wolf since the argument is symmetric for P_3. Suppose that the lone-wolf condition is not true, i.e., P_1 is the lone wolf and can cause non-negligible bias towards some $b \in \{0, 1\}$. Recall that outside P_1 there is at least one b-supporter left henceforth denoted P^*. Now, consider the fail-stop coalition containing the parties in P_1 and P^* — this coalition prefers b. Moreover, this coalition can cause non-negligible bias towards b by having parties in P_1 act P_1's strategy and having P^* act honestly. This violates CSP-fairness. □

Honest execution is unbiased even when conditioning on T_2. Now, consider an honest execution but conditioned on the fact that P_2's randomness is T_2. Let $f(T_2)$ denote the expected outcome of an honest execution, conditioned on the fact that P_2's randomness is T_2. Our next goal is to prove the following lemma which is akin to Lemma 1 in the proof of Theorem 7 — however, in comparison with Lemma 1, here we allow a negligible fraction of T_2's to not satisfy the stated condition.

Lemma 7 (Honest execution is unbiased even when conditioning on T_2)**.** *There exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all but* $\mathsf{negl}(\kappa)$ *fraction of* T_2*'s* $|f(T_2) - \frac{1}{2}|$ *is a negligible function in* κ.

If we could prove the above lemma, the remainder of the proof would follow in exactly the same manner as that of Theorem 7. Recall that the proof of Theorem 7 goes as follows: assume that Lemma 1 holds and moreover, both the lone-wolf and the wolf-minion conditions hold, now we can construct a polynomial-time attack that violates the wolf-minion condition which creates a contradiction.

In our proof of Theorem 10, we shall follow exactly the same proof strategy as Theorem 7, except that now Lemma 1 is replaced with the slightly weaker version of Lemma 7 which deducts a negligible fraction of bad T_2's. It is not difficult to see that even with this slightly weakened version, all the remainder of the proof of Theorem 7 would nonetheless hold (simply by deducting the negligible fraction of bad T_2's from consideration whenever appropriate).

The crux therefore is to prove Lemma 7 which is the slightly weakened form of Lemma 1. The challenge is the following: earlier in Theorem 7, we proved Lemma 1 by relying on maximin fairness against *semi-malicious* adversaries. Here, however, we only have that the protocol is CSP-fair against *fail-stop* adversaries (*c.f.* note that Lemma 1 does not hold for maximin fairness against *fail-stop* adversaries).

Proof of Lemma 7. We now prove Lemma 7. By correctness of a coin toss protocol, we trivially have that $\mathbf{E}_{T_2}[f(T_2)] = \frac{1}{2}$. Now, we would like to prove that in fact $f(T_2)$ is negligibly different from $\frac{1}{2}$ almost everywhere, i.e., for all but a negligible fraction of T_2's.

Proof intuition. It seems challenging to directly compare $f(T_2)$ with $f(T_2')$ for arbitrary $T_2 \neq T_2'$. Recall that T_2 consists of the randomness from m parties where m is the size of the partition P_2. We can thus regard $T_2 := \{t_\mathfrak{p}\}_{\mathfrak{p} \in P_2}$ as a vector where each coordinate $t_\mathfrak{p}$ is one party's contribution towards the randomness. To overcome the above challenge, our high-level idea is the following:

1. First, we will compare $f(T_2)$ with the expected outcome in the following execution conditioning on T_2. Suppose that there is a single fail-stop party in P_2 henceforth called the lone wolf and denoted $\mathfrak{w} \in P_2$. Further, the wolf always aborts upfront prior to any one speaks. We will show that for almost all T_2's, $f(T_2)$ must be almost the same as the expected outcome in the latter execution with the wolf also conditioning on T_2.

2. Next, we will show that for almost all choices of T_2 and T_2' that differ only in the wolf's randomness (henceforth we say that T_2 and T_2' are neighboring), it must be that $f(T_2)$ and $f(T_2')$ are negligibly apart. To see this, we can compare $f(T_2)$ or $f(T_2')$ to the expected outcome when fixing P_2's randomness to T_2 or T_2', but when the wolf aborts upfront. If the wolf aborts upfront, however, then the wolf's randomness does not matter to the expected outcome — thus the expected outcome should depend only on the part of T_2 or T_2' excluding the wolf's randomness.

Now, with the above arguments, we shall conclude that $f(T_2)$ is negligibly different from $\frac{1}{2}$ for almost all T_2's.

Formal proof. Below we formalize the above intuition. Let $\mathfrak{w} \in P_2$ be any fixed party in P_2. Let $\mathcal{A}(\mathfrak{w})$ be an adversary that controls $\mathfrak{w}$ alone and always makes $\mathfrak{w}$ abort upfront prior to anyone speaks.

For any fixed wolf $\mathfrak{w} \in P_2$, let $g^{\mathfrak{w}}(T_2)$ denote the expected outcome of an execution when playing with $\mathcal{A}(\mathfrak{w})$, and conditioned on the fact that P_2's randomness is T_2.

Lemma 8 (A lone wolf in P_2 cannot cause noticeable bias conditioned on T_2.)**.** *For any fixed $\mathfrak{w} \in P_2$, there exists a negligible function $\mathsf{negl}(\cdot)$, such that for all but $\mathsf{negl}(\kappa)$ fraction of T_2's, it holds that $|f(T_2) - g^{\mathfrak{w}}(T_2)|$ is negligibly small in κ.*

Proof. Suppose that the lemma is not true, i.e., for some fixed $\mathfrak{w} \in P_2$, there are polynomials $p(\cdot)$ and $q(\cdot)$ such that for $1/q(\kappa)$ fraction of T_2's, $|f(T_2) - g^{\mathfrak{w}}(T_2)| \geq 1/p(\kappa)$. Now, observe the following claim.

Claim 6. *Fix any $\mathfrak{w} \in P_2$, it holds that $|\mathbf{E}_{T_2}[g^{\mathfrak{w}}(T_2)] - \frac{1}{2}|$ is negligible in κ.*

Proof. It suffices to prove that in execution where $\mathfrak{w}$ always aborts upfront, the expected outcome is negligibly different from $\frac{1}{2}$. If not, then $\mathfrak{w}$ must be able to cause non-negligible bias either towards 0 or 1. If it can create non-negligible bias towards $b \in \{0, 1\}$, then obviously a fail-stop coalition containing $\mathfrak{w}$ (who always aborts upfront) and two additional b-supporters (who simply follow the honest algorithm) can cause non-negligible bias towards b — this violates CSP-fairness. Notice that the two additional b-supporters must exist (e.g., one from P_1 and one from P_3) due to our partition creation algorithm. □

Given the above claim, and recalling that by honest execution $\mathbf{E}_{T_2}[f(T_2)] = \frac{1}{2}$, we have that $|\mathbf{E}_{T_2}[f(T_2)] - \mathbf{E}_{T_2}[g^{\mathfrak{w}}(T_2)]| = |\mathbf{E}_{T_2}[f(T_2) - g^{\mathfrak{w}}(T_2)]|$ is negligible small in κ for any fixed $\mathfrak{w}$. Thus if there are non-negligible fraction of T_2's such that $f(T_2) - g^{\mathfrak{w}}(T_2) \geq 1/p'(\kappa)$ for some polynomial $p'(\cdot)$, i.e. the wolf biases noticeably towards 0 on non-negligible fraction of T_2's, then there must be non-negligible fraction of T_2's such that $g^{\mathfrak{w}}(T_2) - f(T_2) \geq 1/p''(\kappa)$ for some polynomial $p''(\cdot)$, i.e. the wolf biases noticeably towards 1 on non-negligible fraction of T_2's; and vice versa.

Recall that 1-supporters strictly outnumber 0-supporters in P_2. We can now construct a polynomial-time fail-stop adversary $\mathcal{A}^*$ that controls the coalition P_2 and can cause non-negligible bias towards 1. $\mathcal{A}^*$ receives a non-uniform advice string containing some polynomial $p(\cdot)$; $p(\cdot)$ is chosen such that there are non-negligible fraction of T_2's for which $g^{\mathfrak{w}}(T_2) - f(T_2) \geq 1/p(\kappa)$ for sufficiently large κ — henceforth if a T_2 satisfies this condition we say that the T_2 is *good* for $\mathcal{A}^*$. $\mathcal{A}^*$ now performs the following experiment:

- First, $\mathcal{A}^*$ samples a random T_2 for P_2.
- Next, for $p^2(\kappa)$ times, $\mathcal{A}^*$ samples T_1 and T_3 at random for P_1 and P_3 respectively and emulates an execution playing with the wolf, and conditioned on all parties' randomness being fixed to T_1, T_2, T_3. By averaging the outcomes of these executions, it obtains an estimate $\widetilde{g}^{\mathfrak{w}}(T_2)$.

Similarly, for $p^2(\kappa)$ times, $\mathcal{A}^*$ samples T_1 and T_3 at random and emulates an honest execution conditioned on T_1, T_2, T_3. By averaging the outcomes of these executions, it obtains an estimate $\widetilde{f}(T_2)$.

- Now, if $\widetilde{g}^{\mathfrak{w}}(T_2) > \widetilde{f}(T_2)$, $\mathcal{A}^*$ does not make the wolf abort upfront; else it makes the wolf abort upfront.

Observe that by a simple application of the Chernoff bound, the following claim holds.

Claim 7. *For any fixed T_2 that is good for $\mathcal{A}^*$, except with negligible in κ probability in the above experiment, it must hold that $\widetilde{g}^{\mathfrak{w}}(T_2) > \widetilde{f}(T_2)$.*

Now, on the non-negligible fraction of good T_2's, the adversary $\mathcal{A}^*$ can cause $1/p(\kappa)$-bias towards 1 relative to $f(T_2)$; and for all other T_2's the expected outcome is simply $f(T_2)$. It is not difficult to see that averaging over T_2, $\mathcal{A}^*$ can cause non-negligible bias towards 1.

This concludes the proof of Lemma 8. $\square$

Recall that T_2 is formed by concatenating m players' random strings where m is the size of the set P_2. Henceforth we view $T_2 := \{t_{\mathfrak{p}}\}_{\mathfrak{p} \in P_2}$ as a vector where $t_{\mathfrak{p}}$ is the contribution of player $\mathfrak{p} \in P_2$. Now, if T_2 and T_2' differ in exactly one $t_{\mathfrak{p}}$, we say that T_2 and T_2' are *neighboring* w.r.t. $\mathfrak{p} \in P_2$.

Fact 7. *For any $\mathfrak{w} \in P_2$, for any T_2 and T_2' that are neighboring w.r.t. $\mathfrak{w}$, it must hold that $g^{\mathfrak{w}}(T_2) = g^{\mathfrak{w}}(T_2')$.*

Proof. Follows in a straightforward fashion by observing the following: if the wolf $\mathfrak{w}$ aborts upfront, then the expected outcome conditioning on T_2 depends only on the part of T_2 excluding $\mathfrak{w}$'s contribution. $\square$

We use the notation $T_2' := T_2^{-i}$ to denote the fact that T_2' is almost identical as T_2 except with the i-th bit flipped (note that the i-th bit is part of some player's randomness since each player may consume multiple random bits).

Lemma 9. *For any $i \in [|T_2|]$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all but $\mathsf{negl}(\kappa)$ fraction of T_2's, $|f(T_2) - f(T_2')|$ is negligibly small in κ where $T_2' := T_2^{-i}$.*

Proof. Suppose that the i-th bit is contributed by party $\mathfrak{w} \in P_2$. Pick a random T_2. Now, for any fixed polynomial $p(\cdot)$, define bad event bad_1^p to be when T_2 satisfies $|f(T_2) - g^{\mathfrak{w}}(T_2)| \geq 1/p(\kappa)$. For any fixed polynomial $p(\cdot)$, define bad event bad_2^p to be when $T_2' := T_2^{-i}$ satisfies $|f(T_2') - g^{\mathfrak{w}}(T_2)| \geq 1/p(\kappa)$. Now for any polynomial $p(\cdot)$, the probability that bad_1^p happens is negligibly small in κ by Lemma 8. Similarly, for any polynomial $p(\cdot)$, the probability that bad_2^p happens is negligibly small in κ. By union bound, for any polynomial $p(\cdot)$, the probability that neither bad_1^p nor bad_2^p happens is $1 - \mathsf{negl}(\kappa)$ for some negligible function $\mathsf{negl}(\cdot)$. The lemma follows by observing that for any fixed polynomial $p(\cdot)$, if neither bad_1^p nor bad_2^p happens, then $|f(T_2) - f(T_2')| \leq 2/p(\kappa)$. $\square$

Lemma 10. *Pick a random T_2 and a random T_2', then except with negligible in κ probability in the choice of T_2 and T_2', it holds that $|f(T_2) - f(T_2')|$ is negligibly small in κ.*

Proof. Pick a random T_2 and a random T_2', each of which contains $\ell = \mathsf{poly}(\kappa)$ bits. Now, for $i = 0, 1, \ldots, \ell$, let T^i be the first i bits from T_2' concatenated with the last $\ell - i$ bits of T_2. Clearly $T^0 = T_2$ and $T^\ell = T_2'$.

Now, for $i = 0, 1, \ldots, \ell - 1$ and for some polynomial $p(\cdot)$, let bad event bad_i^p be such that $|f(T^i) - f(T^{i+1})| \geq 1/p(\kappa)$. By Lemma 9, for any polynomial $p(\cdot)$, and any $i \in \{0, 1, \ldots, \ell - 1\}$, bad_i^p happens with negligible probability (over the choice of T_2 and T_2') — to see this, observe that the marginal distribution of T^i is uniformly at random; and further either $T^{i+1} = T^i$ or T^{i+1} is almost identical to T^i but with the $(i+1)$-th bit flipped.

By union bound, the probability that none of the bad_i^p events happen is $1 - \mathsf{negl}(\kappa)$ for some negligible function $\mathsf{negl}(\cdot)$. Now for any fixed polynomial $p(\cdot)$, if none of the bad_i^p events happen for any $i \in \{0, 1, \ldots, \ell-1\}$, then by triangle inequality we have that $|f(T^i) - f(T^{i+1})| \leq (\ell+1)/p(\kappa)$. $\square$

We now continue to prove Lemma 7. Suppose the lemma is not true — since we know that $\mathbf{E}_{T_2}[f(T_2)] = \frac{1}{2}$, it must be that there exist $1/p(\kappa)$ fraction of T_2's such that $f(T_2) \geq \frac{1}{2} + 1/q(\kappa)$ and moreover there exist $1/p'(\kappa)$ fraction of T_2's such that $f(T_2) \leq \frac{1}{2} - 1/q'(\kappa)$ for some polynomials p, p', q, q'. However, suppose that this is true, then if we pick T_2 and T_2' at random, then with non-negligible probability we have that $|f(T_2) - f(T_2')| \geq 1/q''(\kappa)$ for some polynomial q'' — and this violates Lemma 10. $\square$

C Cooperative-Coalition-Proof Fairness

In this section, we consider another notion of fairness, called cooperative-coalition-proof (CCP) fairness, that is inspired by game theory. Imagine the following coalition formation process. Suppose that before protocol execution starts, the parties can freely discuss with each other and decide whether and how to form coalitions, as well as the coalition's strategy. Suppose that a set of parties, denoted $S \subseteq [n]$, decide that forming a coalition and playing a dishonest (cooperative) strategy M will improve the coalition's overall wealth (i.e., expected total payoff). Such a coalition, however, may not be stable (also referred to as *self-enforcing* henceforth): for example, suppose that a sub-coalition $S' \subset S$ decide that if $S \backslash S'$ plays the strategy M and everyone else plays honest, then S' is better off (in terms of overall wealth) playing another dishonest strategy $M' \neq M$. Now if M' is a self-enforcing strategy itself, then the sub-coalition S' will be incentivized to deviate from the coalition S's strategy M. Now, CCP fairness requires that we defend, not against every cooperative coalition strategy that can improve the coalition's overall wealth (as in CSP-fairness), but only against those that are *self-enforcing*, i.e., no *self-enforcing* sub-coalition can be better off by further deviations (note that the definition of self-enforcing is recursive). Therefore, CCP fairness is a relaxation of the earlier notion CSP fairness (see Section 5.1).

After defining CCP fairness, we will prove that it is in fact equivalent to Strong Nash in the context of coin toss protocols (Theorem 22). Note that the equivalence does not hold for general games — for general games, these two notions are incomparable.

C.1 Generalized Execution Model for More Complex Coalition Structures

To formally define the notion of CCP fairness, we need to model complex coalition behavior that reflects self-interestedness, and not just that all corrupt nodes always form a single coalition (as in the standard cryptography literature). For example, we need to consider the possibility that sub-coalitions may arise from within a coalition and the sub-coalition may decide to deviate from the predetermined coalition strategy.

Thus we need to define a more general execution model that allows us to characterize such more complex coalition dynamics. Our modeling techniques can be viewed as an independent contribution of this paper, and can be used to study new, game-theoretic notions of security in the context of more general cryptographic protocols (and not just coin-toss protocols).

The execution model describe below can be viewed as a generalization of that in Section 2.1 (where it was assumed that all corrupt parties form a single coalition). Like in Section 2.1, we still adopt a round-based execution model where parties exchange messages through a public broadcast channel (with identifiable abort). To model more sophisticated coalition behavior, we introduce the notion of back-channeling that characterizes how corrupt nodes within a coalition may share information with each.

Back-channelling. We assume that corrupt nodes may share arbitrary information with other corrupt nodes (in the same coalition or other coalitions) through a *back channel*, i.e., a out-of-band private communication channel. We assume that such back channel information sharing can *take place at any time, even in the middle of a round.* Further, (possibly multiple rounds of) back-channelling can take place before protocol start, e.g., for a coalition of players to decide on a strategy.

Rushing attack. Corrupt parties in any coalition can perform a *rushing* attack, i.e., any corrupt party can decide that in some round r, it wants to hear messages from a subset of other parties first before it chooses its own message to send in the same round r. In a rushing attack, a corrupt party can not only wait for an honest party to send message, it can also wait for a corrupt party in the same or a different coalition. For example, a coalition may jointly decide to take a strategy where some parties in the coalition wait for others to speak before they speak in the same round — and such rushing within the same coalition may help stabilize the coalition and prevent sub-coalitions from deviating.

Since each corrupt party can be blocked waiting for other parties to speak, a deadlocking situation may occur: e.g., when party i is blocked waiting for party j to send message which in turn is waiting for party i to send message (and the cycle can be longer too). Henceforth in this paper, we assume that whenever such circular dependencies happen that cause a deadlock, all parties involved in the circular dependency effectively abort from the protocol in the round r in which the circular dependency takes place — and such aborting may be detected at the beginning of round $r + 1$.

Henceforth we will assume the honest algorithm of any well-defined protocol must satisfy the following:

1. does not send any messages over back channels and ignores all messages received over a back channel;

2. in each round, sends messages at the beginning of the round without performing any rushing attack.

We define the following families of adversarial algorithms w.r.t. to some honest protocol Π:

- *Fail-stop.* An algorithm A (executed by a party) is said to be a fail-stop adversary w.r.t. the honest protocol Π iff A would otherwise follow the honest protocol Π except that it can 1) send and receive information over back channels; 2) perform rushing attacks; and 3) decide to abort prematurely based on the party's view in the protocol (including back-channel messages) so far.

- *Semi-malicious.* An algorithm A (executed by a party) is said to be a semi-malicious adversary w.r.t. the honest protocol Π iff A would otherwise follow the honest protocol Π except that it can deviate in all the ways that a fail-stop adversary could, and moreover, prior to protocol execution, A can participate in (possibly multiple rounds of) back-channel discussion and choose the party's random string arbitrarily based on its view in the back-channel discussion.

- *Malicious.* An algorithm A (executed by a party) is said to be a malicious adversary w.r.t. Π iff it can deviate arbitrarily from Π, send arbitrary protocol messages, and perform all the attacks that a semi-malicious adversary could.

Remark 4 (Special case: a single corrupt coalition)**.** The earlier model in Section 2.1 assumed that all corrupt nodes form a single coalition — this can be viewed as a special case of the more generalized model. Specifically, due to the way we define back-channelling, when all corrupt nodes form a single coalition, we can equivalently think of all corrupt nodes as being controlled by a single adversary $\mathcal{A}$. All corrupt nodes share their views in the protocol with $\mathcal{A}$. The adversary $\mathcal{A}$ can now instruct corrupt nodes how to behave based on the joint views of all corrupt nodes in the protocol. For the special case of a single coalition, during a rushing attack, corrupt parties wait only for honest parties to speak before they speak in the same round.

C.2 Protocol-Induced Games and Sub-games

To define CCP fairness, we need to consider games and sub-games that are induced by a protocol's execution. Here, we view a protocol's execution as a game. Recall that we adopt the generalized execution model described in Section C.1 that allows us to characterize complex coalition structures.

Protocol-induced game. A protocol-induced game among n players is a tuple

$$G := ([n], \mathcal{M}, u)$$

where $[n]$ denotes the set of players, $\mathcal{M}$ is a family of possibly randomized algorithms that denotes every player's strategy space, and u is a utility function where $u_i(M_1, \ldots, M_n)$ determines player i's expected utility when all players play the strategy vector $(M_1, \ldots, M_n) \in \mathcal{M}^n$. Recall that each M_i where $i \in [n]$ not only can interact with other parties through the broadcast channel, but can also communicate with any chosen subset of other nodes through an out-of-band back channel.

Under this definition, we can view any n-player coin toss protocol as a game. In particular, the utility function u is well-defined given the parties' preference profile $\mathcal{P} \in \{0,1\}^n$, and given that the outcome of the protocol can be deterministically computed from the protocol's transcript.

Sub-games. Let $G := ([n], \mathcal{M}, u)$ denote a protocol-induced game among n players. Let $S \subset [n]$ be a non-empty coalition of players, let $-S := [n] \backslash S$ denote the complement of S. Suppose that we fix the strategies of $-S$ to $M^{-S} \in \mathcal{M}^{|-S|}$, and let the set S freely choose their strategies from a family of (possibly randomized) algorithms $\mathcal{M}'$. We henceforth refer to this as a sub-game induced on S by the strategies M^{-S} of the set $-S$ of players. We use G^S_M to denote this induced sub-game among the players in S: if $i \in S$ decides to play the strategy $Q_i(1^\kappa) \in \mathcal{M}'$, then the utility of player $i \in S$ in the sub-game G^S_M is denoted $u_i(Q^S, M^{-S})$. Thus, we can denote G^S_M as

$$G^S_M := (S, \mathcal{M}', \{u_i(\cdot, M^{-S})\}_{i \in S})$$

Note that any sub-game is also a (protocol-induced) game.

C.3 Cooperative-Coalition-Proof Fairness

As mentioned, cooperative-coalition-proof (CCP) fairness is a relaxation of cooperative-strategy-proof fairness. In a cooperative-coalition-proof fair (CCP-fair) protocol, the honest protocol only needs to defend against any cooperative strategy that is *self-enforcing*. A cooperative strategy among a coalition C is said to be self-enforcing, iff no *self-enforcing* cooperative sub-coalition can be better off by further deviation (note the recursive nature of the definition). Our CCP fairness notion is inspired by the *coalition-proof* notion that is standard in game theory [15]. Cooperative-coalition-proof differs from *coalition-proof* in the following sense: the latter notion defends against self-enforcing coalition strategies that seek to improve the payoff of *every* player in the coalition; whereas the former defends against self-enforcing *cooperative* coalition strategies that seek to improve the *overall wealth* of the coalition.

In the formal definition below, we consider protocol-induced games and sub-games.

Definition 8 (Cooperative-coalition-proof)**.** In a single-player game $G := ([1], \mathcal{M}, u_1)$, a (possibly randomized) algorithm $M^*(1^\kappa) \in \mathcal{M}$ is said to be is a cooperative-coalition-proof Nash Equilibrium (CCPNE) iff there is some negligible function $\mathsf{negl}(\cdot)$ such that for every algorithm $M(1^\kappa) \in \mathcal{M}$, it holds that $u_1(M^*) \geq u_1(M) - \mathsf{negl}(\kappa)$.

Now, let $n > 1$ and assume that CCPNE has been defined for all games with fewer than n players. For any n-player game $G := ([n], \mathcal{M}, u)$, let $M(1^\kappa) := (M_1(1^\kappa), \ldots M_n(1^\kappa)) \in \mathcal{M}^n$ denote some strategy profile for all players. We define *self-enforcing* and CCPNE for an n-player game as follows.

1. We say that $M(1^\kappa)$ is self-enforcing iff for any *proper* subset $S \subset [n]$, M^S is a CCPNE in the induced sub-game G^S_M.

2. We say that $M(1^\kappa)$ is a CCPNE if it is self-enforcing and there exists a negligible function $\mathsf{negl}(\cdot)$, such that for every *self-enforcing* strategy $M'(1^\kappa) := (M'_1(1^\kappa), \ldots M'_n(1^\kappa)) \in \mathcal{M}^n$, it must be that $\sum_{i=1\in[n]} u_i(M) \geq \sum_{i=1\in[n]} u_i(M') - \mathsf{negl}(\kappa)$.

Definition 9 (Cooperative-coalition-proof fairness)**.** Let Π denote an n-party coin-toss protocol, and let $\mathcal{M}$ denote a family of adversarial algorithms (executed by one party) that includes the honest algorithm defined by Π. Let $\mathcal{P} \in \{0,1\}^n$ denote a divided preference profile. The protocol Π is said to be cooperative-coalition-proof fair (or CCP-fair for short) for $\mathcal{P}$ w.r.t. the family $\mathcal{M}$ iff the strategy profile where all players follow the honest algorithm is a CCPNE in the protocol-induced game $G := ([n], \mathcal{M}, u^{\mathcal{P}})$, where $u^{\mathcal{P}}$ is the utility function for all players given the preference profile $\mathcal{P}$.

Specifically, if $\mathcal{M}$ is the class of all non-uniform p.p.t. fail-stop (or semi-malicious, malicious resp.) adversarial algorithms w.r.t. to the honest protocol, we say that the protocol is *computationally* CCP-fair against fail-stop (or semi-malicious, malicious resp.) adversaries. If $\mathcal{M}$ is the class of all possibly unbounded fail-stop (or semi-malicious, malicious resp.) adversarial algorithms w.r.t. the honest protocol, we say that the protocol is *statistically* CCP-fair against fail-stop (or semi-malicious, malicious resp.) adversaries. If the protocol satisfies CCP-fairness with a choice of 0 for the negligible function, then the protocol is said to be *perfectly* CCP-fair.

C.4 Equivalence of CCP and SNE for Coin Toss Protocols

In this section, we show that our notion of CCP-fairness is equivalent to Strong Nash in the context of coin-toss protocols. To prove this, we must first adapt the earlier SNE-fairness definition to our generalized model of execution defined in Section C.1. Recall that here we model a corrupt coalition by the vector of their strategies (rather than a single adversary $\mathcal{A}$), and we allow sharing of information between coalition members through back channels.

Definition 10 (SNE in protocol-induced games)**.** Let $G := ([n], \mathcal{M}, u)$ be a protocol-induced game. We say that the strategy profile $M \in \mathcal{M}^n$ is a Strong Nash Equilibrium (SNE), iff for any sub-coalition $S \subseteq [n]$ containing up to n parties, for any coalition strategy $Q^S \in \mathcal{M}^{|S|}$, there exists a negligible function $\mathsf{negl}(\cdot)$ and some $i \in S$ such that $u_i(Q^S, M^{-S}) \leq u_i(M) + \mathsf{negl}(\kappa)$.

In other words, SNE requires that in the equilibrium strategy, if any sub-coalition decides to deviate, at least one coalition member will not have noticeable gain.

Definition 11 (SNE-fairness in the generalized execution model)**.** Let Π denote an n-party coin toss protocol, and let $\mathcal{M}$ be a family of adversaries (executed by one party) that includes the honest algorithm defined by Π. Let $\mathcal{P} \in \{0,1\}^n$ be any divided preference profile. We say that Π is SNE-fair for $\mathcal{P}$ and w.r.t. $\mathcal{M}$, iff the strategy profile where all parties play the honest protocol Π is an SNE in the Π-induced game $G := ([n], \mathcal{M}, u^{\mathcal{P}})$ where $u^{\mathcal{P}}$ is the utility function for all players under the preference profile $\mathcal{P}$.

The following fact says that in any (sub-)game induced by a coin toss protocol, an SNE strategy only needs to defend against deviations by unanimous sub-coalitions (i.e., coalitions where members have consistent preference).

Definition 12 (Resists deviation from coalition)**.** Let $G := ([n], \mathcal{M}, u)$ be a (sub-)game induced by a multi-party coin toss protocol. We say that a strategy profile $M \in \mathcal{M}^n$ *resist deviation from some coalition* $S \subseteq [n]$ containing up to n parties, iff for any coalition strategy $Q^S \in \mathcal{M}^{|S|}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $u_S(Q^S, M^{-S}) \leq u_S(M) + \mathsf{negl}(\kappa)$, where $u_S(M') := \sum_{i \in S} u_i(M')$ denotes the total payoff of the coalition S.

Claim 8. *Let $G := ([n], \mathcal{M}, u)$ be a (sub-)game induced by a multi-party coin toss protocol. Then, $M \in \mathcal{M}^n$ is an SNE iff M resists deviation from any unanimous coalition.*

Proof. First, the following fact is straightforward:

Fact 8. *For any unanimous coalition S, $u_S(M_0) \leq u_S(M_1) + \mathsf{negl}'(\kappa)$ iff there exists $i \in S$, $u_i(M_0) \leq u_i(M_1) + \mathsf{negl}'(\kappa)$ for some other negligible function $\mathsf{negl}'(\cdot)$.*

- (SNE $\Longrightarrow$ resists unanimous coalition): If a strategy M is an SNE, by definition of SNE, for any unanimous coalition S and strategy $Q \in \mathcal{M}^{|S|}$, there exists a negligible function $\mathsf{negl}(\cdot)$ and some $i \in S$ such that $u_i(Q^S, M^{-S}) \leq u_i(M) + \mathsf{negl}(\kappa)$. For a unanimous coalition S, by Fact 8, we have that $u_S(Q^S, M^{-S}) \leq u_S(M) + \mathsf{negl}'(\kappa)$ for some negligible function $\mathsf{negl}'(\cdot)$.

- (Resists unanimous coalition $\Longrightarrow$ SNE): Suppose that for any unanimous coalition S and strategy $Q \in \mathcal{M}^{|S|}$, $u_S(Q^S, M^{-S}) \leq u_S(M) + \mathsf{negl}(\kappa)$ for some negligible function $\mathsf{negl}(\cdot)$. By Fact 8, for any unanimous coalition S and strategy $Q \in \mathcal{M}^{|S|}$, there exists some negligible function $\mathsf{negl}(\cdot)$ and $i \in S$ such that $u_i(Q^S, M^{-S}) \leq u_i(M) + \mathsf{negl}(\kappa)$

  Now, consider a divided coalition $S \subseteq [n]$. If the parties in S that prefer $b \in \{0,1\}$ have expected reward more than $u_i(M)$ by deviating, then those who prefer $1-b$ in S have expected reward less than $u_i(M)$. This concludes the proof.

□

We can now prove the equivalence of CCPNE and SNE in the context of coin toss protocols and induced (sub-)games.

Theorem 22 (Equivalence of SNE and CCP for multi-party coin toss)**.** *Let Π denote an n-party coin toss protocol, and let $\mathcal{M}$ be a family of adversarial algorithms (executed by one party) including the honest strategy defined by the protocol Π. Let $\mathcal{P} \in \{0,1\}^n$ be any divided preference profile. Then, Π is CCP-fair for $\mathcal{P}$ and w.r.t. $\mathcal{M}$ iff Π is SNE-fair for $\mathcal{P}$ and w.r.t. $\mathcal{M}$.*

Proof. It suffices to prove that for any game induced by a coin-toss protocol or any sub-game among n players, let $G := ([n], \mathcal{M}, u)$ denote this game or sub-game — then, a strategy profile $M \in \mathcal{M}^n$ is a CCPNE iff M is an SNE.

For $n = 1$, the claim holds trivially by definitions of CCPNE and SNE. Now, suppose that the claim holds for any $\widetilde{n} < n$, we now prove it inductively for the case of n.

We first prove a useful lemma.

Lemma 11. *Suppose that the induction hypothesis holds for any $\widetilde{n} < n$. Then, in n-party game induced by a coin toss protocol, $M \in \mathcal{M}^n$ is is self-enforcing iff M resists deviation from any unanimous sub-coalition of size at most $n - 1$.*

Proof. M is self-enforcing iff for any proper sub-coalition $S \subset [n]$, M^S is a CCPNE in G^S_M. By induction hypothesis, this means that M is self-enforcing iff for any proper sub-coalition $S \subset [n]$, M^S is an SNE in G^S_M. The lemma now follows by Claim 8. $\square$

We now proceed with the induction step.

1. CCPNE $\Longrightarrow$ SNE: suppose that in an n-player game or sub-game (denoted G) induced by a coin-toss protocol, some strategy profile M is a CCPNE, we would like to show that M is an SNE too. To show this, it suffices to show that M resists deviation from any unanimous coalition.

   By definition of CCPNE, we know that M is self-enforcing. By Lemma 11, M resists deviation from any proper unanimous coalition.

   It remains to prove that if all parties prefer the same bit b, then M resists the deviation by the global coalition (i.e., the coalition that includes everyone). Let $\widetilde{M} \in \mathcal{M}^n$ be the strategy such that $u_{[n]}(\widetilde{M}) \geq u_{[n]}(M')$ for all $M' \in \mathcal{M}^n$. It suffices to show that $\widetilde{M}$ must be self-enforcing — if so, it cannot be that $u_{[n]}(\widetilde{M}) > u_{[n]}(M) + p(\cdot)$ since otherwise it would violate the fact that M is a CCPNE.

   To show that $\widetilde{M} \in \mathcal{M}^n$ is self-enforcing, by Lemma 11, it suffices to show that $\widetilde{M}$ resists deviation from any proper sub-coalition $S \subset [n]$ — this is guaranteed by the construction of $\widetilde{M}$ and all parties prefer the same bit.

2. SNE $\Longrightarrow$ CCPNE: Suppose that M is an SNE, and we would like to show that M is a CCPNE. By Claim 8, M resists deviation from any unanimous coalition. By Lemma 11, M is self-enforcing. It suffices to show that no self-enforcing M' improves the global coalition's utility by more than a negligible amount. There are two cases.

   - Case 1: if everyone in $[n]$ has the same preference, the conclusion follows from the fact that M is SNE.
   - Case 2: if not everyone in $[n]$ has the same preference, then suppose for the sake of contradiction there is a self-enforcing that outperforms M by more than a negligible amount in terms of global payoff (i.e., the sum of everyone's payoff) — henceforth let M' be the self-enforcing strategy that maximizes the global payoff. If the global coalition obtains the same payoff for both outcomes 0 and 1, then the claim follows directly since the global coalition is indifferent. Otherwise, we may without loss of generality assume that the global coalition obtains higher payoff when the outcome is 1. Now, consider a hybrid strategy M_{hyb} where the 0-supporters in $[n]$ play their respective strategies defined by M, and the 1-supporters in $[n]$ play their respective strategies defined by M'. Since M' is the best self-enforcing global strategy, it must be that M_{hyb} achieves a global payoff that is not negligibly smaller than the global payoff of M' — if so, then the 0-supporters are better off deviating and M' would not be self-enforcing by Lemma 11. Therefore, we conclude that in M, the 1-supporters will be better off deviating to their strategies in M_{hyb}. This means that M cannot be SNE.

$\square$

D Another Natural Payoff Function

So far we have focused on a natural payoff function where winners obtain unit payoff and losers obtain nothing — henceforth we refer to this payoff function as *unit payoff* (see Section 3.1). In this section, we consider another natural payoff function henceforth referred to as *zero-sum payoff*. Imagine the following game: every player expresses a public preference for an outcome that is either 0 or 1. Additionally, every player puts down a bet of 1 Ether — thus the total pot is n Ethers where n denotes the number of players If the coin flip turns out to be b, the b-supporters evenly split the pot. More formally, suppose there are n_b b-supporters for $b \in \{0, 1\}$ such that $n_0 + n_1 = n$: if the outcome of the coin toss is b, then, each b-supporter obtains a pay-off of $\frac{n-n_b}{n_b}$; and the $(1-b)$-supporters each has a payoff of -1. All of our game-theoretic fairness notions (including maximin, group maximin, CSP, SNE, and CCP) are well-defined for this new pay-off function too.

We now discuss the (in)feasibility of achieving game-theoretic notions of fairness for this new payoff function.

D.1 Maximin, Group Maximin, SNE, and CCP-Fairness

For almost all game-theoretic notions we considered with the exception of CSP-fairness, it turns out that fairness under *unit* payoff is equivalent to fairness under *zero-sum* payoff (assuming that we consider coalitions of up to $n-1$ parties for maximin and group maximin fairness, and coalitions of up to n parties for SNE and CCP fairness). More concretely, we show that the following theorem:

Theorem 23. *Let $X \in \{$maximin, group maximin, SNE, CCP$\}$. Let $\mathcal{P} \in \{0, 1\}^n$ be any preference profile and let $n \geq 2$. An n-party coin-toss protocol Π satisfies computational (or statistical, perfect resp.) X-fairness against fail-stop (or malicious resp.) for the earlier unit payoff function, iff Π satisfies computational (or statistical, perfect resp.) X-fairness against fail-stop (or malicious resp.) for the zero-sum payoff.*

Thus for all aforementioned notions excluding CSP-fairness, all the feasibility and infeasibility results described earlier are directly applicable to the zero-sum utility.

We now prove the above Theorem 23.

Proof of Theorem 23. We now prove Theorem 23.

- **Maximin and group maximin fairness.** It is not difficult to see that Claim 2 holds for the new 0-sum payoff too. Therefore, a coin-toss protocol Π is *not* maximin-fair under the unit utility iff there exists an adversary controlling $n-1$ parties and some polynomial p such that the coin toss's outcome is b with probability $0.5 - 1/p$ where b is the remaining honest party's preference. Moreover, the same holds for the 0-sum utility too.

- **SNE and CCP.** For SNE fairness, we only care about coalitions with unanimous preferences. For both payoff functions, a protocol is *not* SNE-fair for $\mathcal{P}$ iff for some $b \in \{0, 1\}$, an adversary controlling all the b-supporters can cause the outcome to be b with higher than $0.5 + 1/p$ probability for some polynomial function p.

  For CCP fairness, it is not difficult to see that the equivalence between SNE and CCP fairness (i.e., Theorem 22) holds for the new zero-sum payoff function too.

D.2 CSP Fairness

CSP fairness for the *zero-sum* payoff is *not* equivalent to CSP fairness for the earlier *unit* payoff. However, since the new payoff function is always zero-sum, it is straightforward to see that under the new payoff function, CSP fairness and group maximin fairness are equivalent. More formally, the following fact is straightforward:

Fact 9 (Equivalence of group maximin fairness and CSP fairness for the zero-sum utility)**.** *Let $\mathfrak{A}$ denote a family of adversaries that corrupt up to $n-1$ parties and let $\mathcal{P} \in \{0,1\}^n$ denote any balanced profile. Then, under the zero-sum payoff, an n-party coin toss protocol Π is group maximin fair for $\mathcal{P}$ against the family $\mathfrak{A}$ iff Π is CSP-fair for $\mathcal{P}$ against the family $\mathfrak{A}$.*

Therefore, under the new zero-sum payoff, the feasibility and infeasibility results for CSP fairness equate to the feasibility and infeasibility of maximin fairness under the previous unit payoff (see Section 4).