A Complete Characterization of Game-Theoretically Fair, Multi-Party Coin Toss*

Ke Wu†[2], Gilad Asharov‡[1], and Elaine Shi§[2]

[1]Department of Computer Science, Bar-Ilan University
[2]Computer Science Department, Carnegie Mellon University

March 5, 2022

Abstract

Cleve's celebrated lower bound (STOC'86) showed that a *de facto* strong fairness notion is impossible in 2-party coin toss, i.e., the corrupt party always has a strategy of biasing the honest party's outcome by a noticeable amount. Nonetheless, Blum's famous coin-tossing protocol (CRYPTO'81) achieves a strictly weaker "game-theoretic" notion of fairness — specifically, it is a 2-party coin toss protocol in which neither party can bias the outcome *towards its own preference*; and thus the honest protocol forms a Nash equilibrium in which neither party would want to deviate. Surprisingly, an n-party analog of Blum's famous coin toss protocol was not studied till recently. The work by Chung et al. (TCC'18) was the first to explore the feasibility of game-theoretically fair n-party coin toss in the presence of corrupt majority. We may assume that each party has a publicly stated preference for either the bit 0 or 1, and if the outcome agrees with the party's preference, it obtains utility 1; else it obtains nothing.

A natural game-theoretic formulation is to require that the honest protocol form a coalition-resistant Nash equilibrium, i.e., no coalition should have incentive to deviate from the honest behavior. Chung et al. phrased this game-theoretic notion as "cooperative-strategy-proofness" or "CSP-fairness" for short. Unfortunately, Chung et al. showed that under $(n-1)$-sized coalitions, it is impossible to design such a CSP-fair coin toss protocol, unless all parties except one prefer the same bit.

In this paper, we show that the impossibility of Chung et al. is in fact not as broad as it may seem. When coalitions are majority but not $n-1$ in size, we can indeed get feasibility results in some meaningful parameter regimes. We give a complete characterization of the regime in which CSP-fair coin toss is possible, by providing a matching upper- and lower-bound. Our complete characterization theorem also shows that the mathematical structure of game-theoretic fairness is starkly different from the *de facto* strong fairness notion in the multi-party computation literature.

*The author ordering is randomized

†kew2@andrew.cmu.edu

‡gilad.asharov@biu.ac.il

§runting@gmail.com

Contents

1 Introduction

Coin toss protocols, first proposed by Blum [Blu81], are at the heart of cryptography and distributed computing. Imagine that Murphy and Mopey simultaneously solve the same long-standing open problem in cryptography, and they both submit a paper with identical results to EUROCRYPT'22. The program committee of EUROCRYPT'22 decide to recommend Murphy and Mopey to merge their papers. Now, Murphy and Mopey want to toss a coin to elect one of them to present the result at EUROCRYPT'22. How can Murphy and Mopey accomplish this task remotely? Clearly, we can use Blum's coin toss protocol. Murphy and Mopey each commit to a random bit, and post the commitment to a public bulletin board (e.g., a blockchain). They then each open their commitments. If the XOR of the two opened bits is 1, Murphy wins; else, Mopey wins. If either player aborts any time during the protocol or does not provide a valid opening for its commitment, it automatically forfeits and the other player wins. Although not explicitly stated in his ground-breaking paper [Blu81], Blum's protocol actually achieves a natural, *game-theoretic* notion of fairness. Since both players want to get elected, we may assume that the winner obtains utility 1, and the loser obtains utility 0. Observe that a rational player who aims to maximize its utility has no incentive to deviate from the honest protocol. Any deviation (including aborting or opening the commitment wrongly) would cause it to lose.

Although this game-theoretic notion of fairness is very natural, it seems to have been overlooked in the subsequent long line of work on multi-party computation (MPC) [Yao82, Yao86, GMW87]. Specifically, the MPC line of work instead switched to considering a *strictly* stronger notion of fairness henceforth called *unbiasability*. Unbiasability requires that an adversary controlling a corrupt coalition cannot bias the outcome of the coin toss whatsoever. Blum's protocol actually does not satisfy this strong, unbiasability notion: a player can indeed bias the outcome in Blum's protocol, although the bias would never be in its own favor. This unbiasability notion has been thoroughly explored in the cryptography literature. It is well-known that in general, if the majority of the players are honest, then unbiasability is indeed attainable [GMW87, BGW88, CCD88, RB89]. On the other hand, the celebrated lower bound of Cleve [Cle86] shows that if half or more of the players are corrupt, unbiasability is impossible — in particular, this lower bound applies to the two-party case where one party can be corrupt.

Despite Cleve's lower bound, the fact that Blum's protocol can achieve meaningful fairness in the two-party case is thought provoking. A natural question arises: *can we achieve game-theoretically fair coin toss in the multi-party setting in the presence of a majority coalition*? Somewhat surprisingly, this question was not explored till the very recent work of Chung et al. [CGL+18].

Imagine that each player has a publicly stated preference for either the bit 0 or 1. If the coin toss outcome agrees with the player's preference, it obtains utility 1; else it obtains nothing. This formulation can have interesting applications. For example, imagine that n parties in a blockchain protocol want to jointly elect a random block proposer among two possible candidates, and users have different preferences among the two depending on which one they are geographically closer to. Another example is where n investors who have invested money into a crowd-funding smart contract want to randomly choose a kick-starter to fund among two candidates, and each player may have a different preference in mind.

In many applications, the preference profiles are public. For example, suppose some blockchain community wants to randomly choose among two governance proposals. Here, the voters are public figures/community leaders whose affiliations, opinions, and past forum posts are known. In general, when the voters' identities/reputations are publicly known and identities do not come for free, voters' preferences are usually public. Another example is games where players must put in stake to play. For example, suppose n players play binary roulette on a blockchain. Here, their preferences are made explicit by their public bets which they cannot lie about.

Chung et al. suggested the following natural formulations of game theoretic fairness for multi-party coin toss, both of which would equate to Blum's notion in the 2-party special case:

- **CSP-fairness:** *Cooperative-strategy-proofness* (or "CSP-fairness" for short) requires that no coalition can *increase* its own expected utility, *no matter how it deviates from the prescribed protocol*. In this way, the honest protocol forms a *coalition-resistant Nash equilibrium*, and no profit-seeking coalition of players would be incentivized to deviate from this equilibrium.

- **Maximin fairness:** Another natural notion is called *maximin fairness*, which requires that no coalition can *harm* any honest party (no matter how the coalition deviates from the prescribed protocol). More precisely, for any (computational) strategy adopted by a coalition of players, the expected utility of any honest party is at most negligibly apart from its utility in an all-honest execution. As motivated by Chung et al. [CGL+18], maximin fairness guarantees that no coalition aiming to monopolize the eco-system by harming and driving away small individual players has incentives to deviate; moreover, no defensive individual aiming to protect itself in the worst-case scenario has incentives to deviate.

Unfortunately, Chung et al. [CGL+18] showed very broad lower bounds which seem to crush our original hope of using game-theoretic fairness to circumvent Cleve's impossibility [Cle86] in the corrupt majority setting. Specifically, Chung et al. proved that unless all parties except a single one all have the same preference, it would be impossible to realize either CSP-fair coin toss or maximin-fair coin toss.

1.1 Our Results and Contributions

It may seem that Chung et al.'s results have put a pessimistic closure to this direction. However, upon more careful examination, their lower bound proofs implicitly assume that all but one parties can be corrupt and form a coalition. It is not immediately clear whether the impossibility would still hold if majority but not $n-1$ parties are corrupt. We therefore revisit the question originally posed by Chung et al., i.e., whether one can rely on game-theoretic fairness to overcome Cleve's impossibility for coin toss protocols in the corrupt majority setting. Specifically, we focus on the following refinement of the question:

Can we achieve game-theoretically fair coin toss under for majority but not necessarily $(n-1)$*-sized coalitions?*

In this paper, we give a complete characterization of the landscape of game-theoretically fair coin toss, including for the CSP-fair and the maximin-fair notions. At a very high level, we show the following results:

- For CSP-fairness, the pessimistic view of Chung et al. [CGL+18] poorly reflects the actual state of affairs. In contrast, we show that under a broad range of parameter regimes, CSP-fairness is possible in the presence of a majority coalition; moreover, we give a complete characterization of the parameter regimes under which CSP-fairness is possible.

- For maximin-fairness, we show that the pessimistic view of Chung et al. indeed applies quite broadly. Roughly speaking, we show that except for the cases when all parties but one prefer the same outcome, or when exactly half of the players are corrupt, maximin-fairness is impossible to attain. We fully characterize maximin fairness as well.

Note that in cases when there is an honest individual with an opposite preference as the coalition, maximin-fairness would directly imply CSP-fairness. This partly explains why maximin-fairness is harder to attain than CSP-fairness.

Our work sheds new light on the intriguing mathematical structure of game-theoretic fairness, which differs fundamentally from the mathematical structure of the *de facto* unbiasability notion that is widely adopted in the cryptography literature. Since coin toss protocols [Blu81] have been the cornerstone of the long line of work on multi-party computation protocols, we hope that our work can inspire future work in the exciting space of "game theory meets multi-party protocols" in general. We now give more formal statements of our results.

CSP fairness. For CSP fairness, we design a new protocol and explore for which range of parameters the upper bound holds. In addition, we generalize the lower bound proof of Chung et al. [CGL+18], and give a precise range of parameters in which impossibility holds. Our upper- and lower-bounds tightly match in their stated parameter regimes. Therefore, our two main results jointly provide a complete characterization of CSP fairness. It is also worth noting that our upper bound holds in the presence of a *malicious* coalition that may deviate from the prescribed protocol arbitrarily to increase its own gain; whereas our lower bound holds for a *fail-stop* coalition whose only possible deviation is to have some of its players abort from the protocol. This makes both the upper- and lower-bound results stronger.

Our results can be summarized with the following theorem statements — below, let n_0 be the number of players that prefer 0 (also called 0-supporters), and let n_1 be the number of players that prefer 1 (also called 1-supporters). Throughout the paper, *without loss of generality, we may assume that* $n_1 \geq n_0 \geq 1$ since the other direction is symmetric. Additionally, we assume $n_0 + n_1 > 2$, since for 2-parties, we can just run Blum's coin toss.

Theorem 1.1 (Upper bound)**.** *Assume the existence of Oblivious Transfer (OT), and without loss of generality, assume that* $n_1 \geq n_0 \geq 1$*, and* $n_0 + n_1 > 2$*. There exists a CSP-fair coin toss protocol which tolerates up to* t*-sized non-uniform p.p.t. coalitions where*

$$t := \begin{cases} n_1 - \lfloor \frac{1}{2} n_0 \rfloor, & \text{if } n_1 \geq \frac{5}{2} n_0; \\ \lfloor \frac{2}{3} n_1 - \frac{1}{6} n_0 \rfloor + \lceil \frac{1}{2} n_0 \rceil + 1 = n_1 + 1, & \text{if } n_1 = n_0 = \text{odd}; \\ \lfloor \frac{2}{3} n_1 - \frac{1}{6} n_0 \rfloor + \lceil \frac{1}{2} n_0 \rceil, & \text{otherwise.} \end{cases} \tag{1}$$

Our upper bound holds even when the coalition may deviate arbitrarily from the prescribed protocol to increase its gain.

Theorem 1.2 (Lower bound)**.** *Without loss of generality, assume that* $n_1 \geq n_0 \geq 1$ *and* $n_0 + n_1 > 2$*. There does not exist a CSP-fair* n*-party coin toss which tolerates coalitions of size* $t + 1$ *or greater where* t *is same as Eq. (1).*

Further, this lower bound holds even for fail-stop coalitions whose only possible deviations are aborting from the honest protocol, and it holds even allowing computational hardness assumptions and restricting the coalition to be computationally bounded.

Previously, the work of [CGL+18] shows possibility only for the case where where $n_0 = 1$ or $n_1 = 1$ and $t = n_0 + n_1 - 1$. Moreover, it showed that it is impossible to tolerate $n_0 + n_1 - 1$ corruptions only for the case where both $n_0, n_1 \geq 2$ (i.e., there are at least two parties among the set of 0-supporters and at least two parties among the set of 1-supporters).

Observe that the optimal resilience parameter t (specified in Eq. (1)) is a function of n_0 and n_1. Intriguingly, its dependence as a function of n_0 and n_1 changes when $n_1 = \frac{5}{2} n_0$. This intriguing *phase transition* partly suggests that *the mathematical structure of game theoretic fairness is starkly* *different the classical notion of unbiasability.* The reason for this phase transition is related to the concrete techniques we adopt to prove our theorems. We will explain why this phase transition occurs as we describe our protocol to help the reader gain intuition (see Remark 2.4 of Section 2.1 for more explanations). Note also that the transition has a continuous boundary, i.e., at exactly $n_1 = \frac{5}{2}n_0$, the two expressions $n_1 - \lfloor\frac{1}{2}n_0\rfloor$ and $\lfloor\frac{2}{3}n_1 - \frac{1}{6}n_0\rfloor + \lceil\frac{1}{2}n_0\rceil$ are equal (to $2n_0$).

Maximin fairness. The work of [CGL+18] shows that maximin fairness is possible against $t \leq n-1$ corruptions only when all but one of the parties are interested in the same outcome. We next show that this is essentially the only interesting setting which does not behave as in the crypto settings. We show that even when allowing a more liberate security threshold, we cannot push the barriers much further than relying on an honest majority. We show the following possibility and its complementary impossibility result:

Theorem 1.3. *Without loss of generality, assume that the number of* 1*-supporters* n_1 *is at least the number of* 0*-supporters,* n_0*, and assume that* $n_0 + n_1 > 2$*. Then:*

- *For* $n_0 \geq 2$*, there does not exist a maximin-fair* n*-party coin toss protocol which tolerates more than* $\lceil\frac{1}{2}(n_0+n_1)\rceil$ *number of* fail-stop *adversaries. Moreover, there exists a (statistically-secure) maximin-fair* n*-party coin toss protocol which tolerates up to* $\lceil\frac{1}{2}(n_0+n_1)\rceil - 1$ *malicious corruptions.*

- *For the special case where* $n_0 = 1$*, we show that there does not exist a maximin-fair* n *party coin toss protocol which tolerates more than* $\lceil\frac{1}{2}n_1\rceil + 1$ *number of (semi-malicious) players. Assuming Oblivious Transfer, there exists a maximin fair-coin tossing protocol tolerating up to* $\lceil\frac{1}{2}n_1\rceil$ *malicious corruptions.*

Public verifiability. Our positive results are achieved in a model that allowed *public verifiability*. In particular, the output of the protocol can be computed from messages that were sent over the broadcast medium (e.g., a public blockchain), and therefore also external *observers*, i.e., parties that do not take part of the computation, can also learn the output. Such public verifiability is often needed in blockchain and decentralized smart contract applications.

2 Technical Overview

This technical overview focuses on the complete characterization of CSP-fairness. In Section 2.1 we discuss the underlying techniques of our upper bound, and in Section 2.2 we present ideas of our lower bound. Intuition for Maximin fairness can be found in the body of the paper (Section 6).

2.1 Upper Bound

Glimpse of Hope. In light of the pessimistic view of Chung et al. [CGL+18], we start with a relatively simple protocol that gives us a glimpse of hope. As a special case, consider the scenario when $n_0 = n_1 = 2$ — recall that for $b \in \{0,1\}$, n_b denotes the number of players that prefer b (also called b-supporters). In this case, the following protocol achieves CSP-fairness against any coalition of at most 2 players. Imagine that we elect one 0-supporter and one 1-supporter arbitrarily as two representatives each preferring 0 and 1, respectively. We now have the two representatives duel with each other using Blum's coin toss, where if the b-supporter aborts then the protocol outputs $1-b$ for $b \in \{0,1\}$. A simple argument proves that this protocol satisfies CSP-fairness:

- If a coalition controls only 1 player, it makes no sense to deviate whether or not the corrupt player is elected representative.
- If the coalition controls 2 players with opposing preferences, then the coalition is indifferent to the outcome and has no incentive to deviate.
- Finally, if the coalition controls 2 players with the same preference, then one of the two will be elected as representative, and the representative should not have incentive to deviate (whereas the non-representative's behavior has no influence to the outcome).

This very simple teaser already shows that Chung et al. [CGL+18]'s impossibility proof does not hold when there is no $(n-1)$-sized coalition. Moreover, it also shows that this notion is weaker than cryptographic fairness, as there is no honest majority and still there is a possibility result. Therefore, the next natural question is to characterize the exact conditions under which we can achieve feasibility.

Warmup Protocol for a Semi-Malicious Coalition

Unfortunately, the approach taken by the above teaser protocol for $n_0 = n_1 = 2$ does not easily generalize to larger choices of n_0 and n_1. We next give a warmup protocol that is somewhat more sophisticated, but it suggests a more general paradigm which inspires our final upper bound result. Chung et al. [CGL+18] gave a protocol against a coalition of size up to n_1 players for $n_0 = 1$, thus we only consider $n_0 \geq 2$ in our construction. For simplicity, we start with the *semi-malicious* model [AJL+12], i.e., the coalition is restricted to the following two types of deviations:

1. It can abort from the protocol in some round, after looking at the honest messages of that round. Moreover, once a player has aborted, it stops participating from that point on.
2. The coalition can choose its random coins to be used in each round after inspecting the honest messages of that round.

Besides these two possible deviations, the coalition would otherwise follow the protocol faithfully.

The $\mathsf{HalfToss}$ sub-protocol. Consider the following sub-protocol called $\mathsf{HalfToss}^b[k]$ where $b \in \{0, 1\}$, and k is a threshold parameter whose purpose will become clear shortly. At a very high level, the sub-protocol chooses a random coin for the group of players that invoke this sub-protocol. Later on, this $\mathsf{HalfToss}^b$ protocol will be executed twice: first among the 0-supporters and all the 1-supporters act as silent observers; and then among the 1-supporters where the 0-supporters act as silent observers. We use $\mathsf{HalfToss}^0$ and $\mathsf{HalfToss}^1$ to distinguish the two instances. Henceforth, let $\mathcal{P}_b \subset [n]$ denote the set of b supporters for $b \in \{0, 1\}$.

The final coin would be the XOR of the coins of the two groups. Thus, the overall idea is similar to the protocol of four parties described in the beginning of the section: Instead of having two representatives (one for the 0-supporters and one for the 1-supporters) and run Blum's protocol among the two representatives, we let the 0-supporters to jointly choose a coin, the 1-supporters to toss their own coin, and then compute the XOR of the two coins. Instead of using a commitment scheme as in Blum protocol, the parties will use secret sharing. Yet, we will carefully define parameters (such as the threshold of the secret sharing scheme) to implement the above idea, and will analyze which parameters lead to optimal resilience.

Protocol 2.1: $\mathsf{HalfToss}^b[k]$ sub-protocol (semi-malicious version)

Sharing phase.

1. Each b-supporter $i \in \mathcal{P}_b$ chooses a random bit $\mathsf{coin}_i \xleftarrow{\$} \{0,1\}$. It then uses $(k+1)$-out-of-n_b Shamir secret sharing[1] to split the coin coin_i into n_b shares, denoted $\{[\mathsf{coin}_i]_j\}_{j \in \mathcal{P}_b}$, respectively. Player i then sends $[\mathsf{coin}_i]_j$ to each player $j \in \mathcal{P}_b$ over a private channel.
2. If a b-supporter has not aborted, post a heartbeat message to the broadcast channel. At this moment, the *active set* $\mathcal{O}_b$ is defined to be the set of all b-supporters that indeed posted a heartbeat to the broadcast channel. Each player $i \in [\mathcal{P}_b]$ computes $s_i := \oplus_{j \in \mathcal{O}_b} [\mathsf{coin}_j]_i$ where $[\mathsf{coin}_j]_i$ is the share player i has received from player j.

Reconstruction phase.

1. Every b-supporter $i \in \mathcal{P}_b$ posts the reconstruction message (i, s_i) to the broadcast channel.
2. If at least $k+1$ number of b-supporters posted a reconstruction message, then reconstruct the final secret s using Shamir secret sharing. Specifically, interpret each reconstruction message of the form (j, s_j) as jointly defining some polynomial f such that $f(j) = s_j$ and the reconstructed secret $s := f(0)$. Output s.
3. Else if fewer than $k+1$ number of b-supporters posted a reconstruction message, output $\perp$.

Properties of the $\mathsf{HalfToss}$ sub-protocol. The $\mathsf{HalfToss}^b[k]$ sub-protocol satisfies the following properties:

- *Binding.* The sharing phase uniquely defines a secret s, such that the reconstruction phase either succeeds and outputs s, or it fails and outputs $\perp$.
- *Knowledge threshold.* If at least $k+1$ number of b-supporters are corrupt, then the coalition can control the outcome of the coin toss. Specifically, during the sharing phase, the coalition will know the coin_i value for every honest i, and thus it can choose the coalition's coin values accordingly to program the outcome to its own liking.

  On the other hand, if at most k number of b-supporters are corrupt, then the coin value s that the sharing phase binds to is uniform and independent of the coalition's view in the sharing phase (i.e., the coalition is completely unaware of this random coin value).
- *Liveness threshold.* If the coalition controls at least $n-k$ number of b-supporters, it can cause the reconstruction to fail and output $\perp$.

  On the other hand, if the coalition controls fewer than $n-k$ number of b-supporters, then the reconstruction phase must succeed.

Our warmup protocol. Our warmup protocol makes use of two instances of the $\mathsf{HalfToss}^b$ sub-protocol among the 0-supporters and 1-supporters, respectively. The two instances are parametrized with the thresholds k_0 and k_1 — we shall first describe the protocol leaving k_0 and k_1 unspecified, we then explain how to choose k_0 and k_1 to get CSP fairness.

Protocol 2.2: Warmup protocol with semi-malicious security

Sharing phase.

1. (0-supporters participate, 1-supporters observe). Run the sharing phase of $\mathsf{HalfToss}^0[k_0]$.
2. (1-supporters participate, 0-supporters observe). Run the sharing phase of $\mathsf{HalfToss}^1[k_1]$.

[1] For concreteness, in $(k+1)$-out-of-n secret sharing, a subset of k parties learn nothing about the secret while each subset of $k+1$ can reconstruct the secret.

Reconstruction phase.

1. (0-supporters participate, 1-supporters observe). Run the reconstruction phase of $\mathsf{HalfToss}^0[k_0]$, and let its outcome be s_0 if reconstruction is successful. In case the reconstruction outputs $\perp$, then let $s_0 := 0$.
2. (1-supporters participate, 0-supporters observe). Run the reconstruction phase of $\mathsf{HalfToss}^1[k_1]$. If the reconstruction phase outputs $\perp$, then output 0 as the final coin value. Else let s_1 be the reconstructed value, and output $s_0 + s_1$ as the final coin value.

Some intuition about the asymmetry. What is most intriguing about the above protocol is the *asymmetry* between the 0-supporters and 1-supporters. In particular, if the reconstruction of the 0-supporters' coin s_0 fails, we set $s_0 = 0$, and as long as the 1-supporters' coins successfully reconstruct, the final outcome will be $s_0 \oplus s_1$. However, if the reconstruction of the 1-supporters' coin s_1 fails, the entire protocol simply outputs 0. At a very high level, one helpful intuition is the following: suppose that there are more 1-supporters than 0-supporters. Then, if the coalition can fail the reconstruction of s_1, it must control so many 1-supporters such that the coalition must prefer 1 (see also condition C3 later). In other words, our protocol makes sure that a coalition who have the capability to potentially abort the reconstruction of s_1 should never have the incentive to actually let it fail. This intuitive condition alone, however, is not enough to make the protocol fully work — below we precisely charcterize the conditions that are necessary to make the protocol work.

Choosing the thresholds k_0 and k_1. Suppose we want to have a CSP-fair protocol for coalitions of size at most t. Let t_0 and t_1 denote the number of corrupted 0-supporters and 1-supporters, respectively. Our idea is to choose the thresholds k_0 and k_1 in light of n_0, n_1, and t, such that the following conditions are satisfied (and recall that we assume without loss of generality that $n_1 \geq n_0$):

(C1) The coalition cannot control both coin values s_0 and s_1. That is, for either $b \in \{0,1\}$, if the coalition controls at least $k_b + 1$ number of b-supporters, then because it is subject to the corruption budget t, the coalition must control at most k_{1-b} number of $(1-b)$-supporters, such that the coin value s_{1-b} is uniform and independent of the coalition's view at the end of the sharing phase.

(C2) If the coalition can control the s_1 coin, i.e., it controls at least $k_1 + 1$ number of 1-supporters, then it cannot hamper the reconstruction of the coin s_0 due to the corruption budget. That is, the coalition must control at most $n_0 - k_0 - 1$ number of 0-supporters.

(C3) If the coalition controls at least $n_1 - k_1$ number of 1-supporters such that it can cause the reconstruction of s_1 to fail, then the coalition must prefer 1 or is indifferent to the outcome. In other words, denoting by t_b the number of corrupted b-supporters and letting $t_1 \geq n_1 - k_1$ then we have two cases: (a) if $n_1 - k_1 \geq n_0$, then this implies that the coalition prefers 1 (since $t_0 \leq n_0 \leq n_1 - k_1 \leq t_1$) and there is no new constraint; otherwise (b) if $n_1 - k_1 < n_0$, then we simply require that $t \leq 2t_1$. This implies that $t_0 \leq t_1$ (and the coalition prefers 1 or is indifferent) since $t = t_0 + t_1$.

If parameters k_0, k_1, t satisfy the following constraints, then they satisfy the above conditions.

Parameter Constraints 2.3 (semi-malicious version)**.**
Assume: $0 \leq k_0 \leq n_0$, $0 \leq k_1 \leq n_1$

(C1) $t \leq k_0 + k_1 + 1$,

(C2) $t \leq k_1 + 1 + n_0 - k_0 - 1 = n_0 + k_1 - k_0$,

(C3) if $n_1 - k_1 < n_0$, then $t \leq 2(n_1 - k_1)$.

Given the above constraints and the parameters n_0, n_1, and t, if a feasible solution for k_0 and k_1 exists, the above warmup protocol (parametrized with the feasible solution k_0 and k_1) would be CSP-fair against t-sized coalitions. The reasoning is as follows.

- First, due to condition (C3), it never makes sense for the coalition to prevent the reconstruction of the s_1 coin (in which case 0 would be the declared output). If the coalition controls enough 1-supporters such that it is capable of failing the reconstruction of s_1, then it either prefers 1 or is indifferent.
- Henceforth we may assume that s_1 is successfully reconstructed. Now, due to condition (C1), there are two cases: 1) either the value of s_1 is uniform and independent of the coalition's view at the end of the sharing phase, or; 2) the coalition can control the value of s_1.

  In the former case, since the coin s_1 is assumed to be successfully reconstructed, the final outcome must be random. It is important that s_1 is reconstructed at the very end, after s_0 is reconstructed. Otherwise, this argument will not hold, since the coalition may examine the reconstructed s_1 value, and then decide whether to abort the reconstruction of s_0. In the latter case, due to conditions (C1) and (C2), it must be that s_0 is uniform and independent of the coalition's view at the end of the sharing phase, and moreover, the coalition cannot hamper the reconstruction of s_0. In this case, the final outcome $s_0 \oplus s_1$ must be random, too.

Optimal resilience for the warmup protocol. Given n_0 and n_1, we may ask what is the optimal resilience for this warmup protocol? Solving for the optimal resilience is equivalent to solving for the maximum t such that there exists a feasible solution for k_0 and k_1 given the above constraints. It turns out that t is maximized under the following choices of k_0 and k_1, depending on n_0 and n_1 where $n_1 \geq n_0 \geq 1$ (see proof in supplementary material C.3):

Case	k_0	k_1	t
If $n_1 \geq \frac{5}{2}n_0$	$\lfloor \frac{n_0}{2} \rfloor$	$n_1 - n_0$	$n_1 - \lfloor \frac{1}{2}n_0 \rfloor$
Otherwise	$\lfloor \frac{n_0}{2} \rfloor$	$\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor$	$\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{n_0}{2} \rceil$

Remark 2.4. *The intuition for the phase transition at $n_1 = \frac{5}{2}n_0$ follows from the implications of the different constraints. In particular, when $n_1 \geq \frac{5}{2}n_0$, then to corrupt a coalition that prefers 0, the adversary does not have to corrupt too many parties, and the conditions are easily satisfied. If the coalition prefers 1, then Condition (C3) does not add any constraint. In that case t is maximized subject to only the constraints corresponding to Condition (C1) and (C2). When $n_1 < \frac{5}{2}n_0$, then it is possible that a coalition corrupting majority parties prefers 0. Therefore, we need to maximize t under the three constraints corresponding to Condition (C1), (C2) and (C3).*

Visualizing the feasible region. In the supplementary material A, we visualize the choice of t as a function of n_0 and n_1, to help understand the intriguing mathematical structure of game-theoretic fairness in multi-party coin toss.

Extension to malicious security. So far, our warmup protocol achieves only semi-malicious security. In Section 4 we also generalize the warmup protocol achieve malicious security. It is worth mentioning that we cannot generically apply a semi-honest/semi-malicious to malicious compiler such as [GMW87, AJL+12] since when faced with corrupt majority, classical compilers completely give up upon anyone aborting, as this is sufficient to guarantee the classical notion of "security with abort". By contrast, our game theoretic setting requires us to deal with aborts more carefully. We also want to avoid using a common reference string, and thus we cannot rely on Non-Interactive Zero-Knowledge proofs (NIZKs) to get public verifiability, like in the compiler of Asharov et al. [AJL+12]. Due to these technicalities, we need additional non-trivial techniques to get the maliciously secure version. We defer to Section 4.1 the maliciously secure protocol and its analysis, leading to Theorem 1.1.

2.2 Lower Bound

Our lower bound techniques are inspired by that of Chung et al. [CGL+18], who proved that there is no CSP-fair n-party coin toss protocol for $n \geq 4$ even against *fail-stop* coalitions, unless all parties except one prefer the same bit (i.e., they assume there are at least two 0-supporters and at least two 1-supporters). In the following, we may assume $n_1 \geq n_0 \geq 2$, since the corner cases where $n_0 = 1$ has already been treated by Chung et al. [CGL+18].

Three-party protocol. The crux of the lower bound of [CGL+18] shows that no three-party coin-toss protocol among $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ can satisfy the following three conditions simultaneously, regardless of the CSP-fairness property. Then, they show how CSP-fair protocol for $n \geq 4$ parties (for some proper values of n_0, n_1 and t) leads to a three party coin-tossing protocol that satisfies these conditions simultaneously. The three conditions are:

(LBC1) *Lone-wolf condition*: a fail-stop adversary controlling $\mathcal{S}_1$ (or $\mathcal{S}_3$) alone adopting any non-uniform p.p.t. strategy cannot bias the output towards *either* direction by a non-negligible amount.

(LBC2) *Wolf-minion condition*: a fail-stop adversary controlling $\mathcal{S}_1$ and $\mathcal{S}_2$ (or $\mathcal{S}_2$ and $\mathcal{S}_3$), adopting any non-uniform p.p.t. strategy, cannot bias the output towards 1 by a non-negligible amount.

(LBC3) T_2-*equity condition*: for all but a negligible fraction of $\mathcal{S}_2$'s random coins T_2, $|f(T_2) - \frac{1}{2}|$ is negligible, where $f(T_2)$ is the expected coin outcome in an honest execution when $\mathcal{S}_2$'s random coins are fixed to T_2.

The following theorem is implicit in the lower bound of [CGL+18], which is proven by considering a sequence of adversaries as the impossibility of Cleve [Cle86], and essentially reducing the three party protocol into a two party coin-tossing protocol. We provide the proof of this theorem in supplementary material D.2.

Theorem 2.5 (Generalized Theorem 21 of Chung et. al. [CGL+18])**.** *There is no coin-tossing protocol* Π *among three parties* $\mathcal{S}_1$, $\mathcal{S}_2$ *and* $\mathcal{S}_3$ *such that* Π *satisfies the above lone-wolf condition (LBC1), the wolf-minion condition (LBC2), and the* T_2*-equity condition (LBC3) simultaneously.*

CSP-fairness. We now show that an n-party protocols satisfying certain conditions can be translated into a three-party protocol that satisfy the above three conditions simultaneously. Analyzing when this transformation is possible would tell us how large a coalition can be tolerated by a CSP-fair coin toss protocol.

Given an n-party protocol Π, the idea is to partition the players into three partitions denoted $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, respectively. We may assume that there is an ordering for the identities of all parties and that the preferences are public. Then:

- $\mathcal{S}_1$ runs the code of the first α_0 number of 0-supporters, and the first α_1 number of 1-supporters.
- $\mathcal{S}_2$ runs the code of the next $(n_0 - 2\alpha_0)$ number of 0-supporters and the next $(n_1 - 2\alpha_1)$ number of 1 supporters.
- $\mathcal{S}_3$ runs the code of the next (last) α_0 number of 0-supporters and the last α_1 number of 1-supporters.

This means that each party $\mathcal{S}_i$ internally emulates the execution of all parties it runs; all messages that are sent between theses parties are dealt internally by $\mathcal{S}_i$ and all messages that are sent between parties that are controlled by different $\mathcal{S}_i$, $\mathcal{S}_j$ are sent as a message from $\mathcal{S}_i$ to $\mathcal{S}_j$ (with a clear labeling that states which message is intended to which internal party).

The idea of the lower bound is to show that as long as there exist α_0, α_1 and t satisfying a set of conditions defined with respect to n_0, n_1, then for any n-party protocol Π achieving CSP-fairness against any non-uniform fail-stop coalition of size t, its corresponding three-party coin-toss protocol must satisfy the above defined properties, simultaneously, in contradiction to Theorem 2.5. This tells us for which values of t with respect to n_0 and n_1 we cannot construct a CSP-fair protocol. The only choice of parameters that is considered in the lower bound of [CGL+18] is when $\alpha_0 = \alpha_1 = \lfloor n_0/2 \rfloor$ and $t = n - 1 = n_0 + n_1 - 1$.

Using our parameterized partitioning, we derive a rather involved system of constraints (Section 5.1) and show that when those constraints are satisfied, then the CSP-fairness of Π implies that the corresponding three-party protocol satisfies lone-wolf, wolf-minion and the T_2-equity conditions (Section 5.2). What is most interesting is that the constraints for the lower bound are *not an immediate negation* of the contraints we encountered for the upper bound, and yet the solution of the feasible space exactly matches what the upper bound can attain. As an example of such a constraint, recall that the lone-wolf condition states that $\mathcal{S}_1$ (resp. $\mathcal{S}_3$) cannot bias the output alone. Since $\mathcal{S}_1$ controls $\alpha_0 + \alpha_1$ parties, we must have that $\alpha_0 + \alpha_1 \leq t$ to make sure that it cannot bias the output alone.

To prove the lower bound, we need to find the minimal t as a function of n_0 and n_1, such that there exist valid α_0, α_1 and t satisfying the system of constraints (Section 5.3). This leads to the proof of Theorem 1.2, which matches our upper bound.

2.3 Related Work

We now review some additional related work.

Game theory meets cryptography. Although game theory [Nas51, J.A74] and multi-party computation [GMW87, Yao82] originated from different academic communities, some recent efforts have investigated the connections of the two areas (e.g., see the excellent surveys by Katz [Kat08] and by Dodis and Rabin [DR07]). At a high level, this line of work focuses on two broad classes of questions.

First, a line of works [HT04, KN08, ADGH06, OPRV09, AL11, ACH11] explored how to define game-theoretic notions of security (as opposed to cryptography-style security notions) for distributed computing tasks such as secret sharing and secure function evaluation. Earlier works in this space considered *a different notion of utility* than our work. Utility functions are often defined with the following assumptions regarding players' perference: players prefer to compute the function correctly; they prefer to learn others' secret data, and prefer that other players do not learn their own secrets. In light of such utility functions, earlier works in this space explored whether we can design protocols such that rational players will be incentivized to follow the honest protocol. Inspired by this line of work, Garay et al. propose a new paradigm called Rational Protocol Design (RPD) [GKM+13], and this paradigm was developed further in several subsequent works [GKTZ15, GTZ15] (we will comment on the relationship of our notion and RPD shortly).

Second, another central question is how cryptography can help traditional game theory. Classical works in game theory [Nas51, J.A74] assumed the existence of a trusted mediator. Therefore, recent works considered how to realize this trusted mediator using cryptography [DHR00, IML05, GK12, BGKO11].

It is well-understood that the notion of Nash equilibrium may predict unstable outcomes since it may rely on empty threats. Our CSP notion adopts the (coalition-resistant) Nash equilibrium paradigm and therefore it does not eliminate the issue of empty threats. In other words, for a CSP-fair protocol, it could be that a player threatens to deviate from the honest protocol (possibly at a harm to itself), making other players reconsider their strategies too. A couple works proposed new notions in the context of computationally bounded agents, aiming to eliminate empty threats. Gradwohl, Livne and Rosen [GLR13] suggested a notion called computational threat-free Nash equilibrium, which can be viewed as a relaxation of the classical notion of subgame perfect equilibrium for computationally bounded agents. This work does not consider coalition resistance. Pass and shelat [PS11] suggest a new notion called renegotiation-safe equilibrium, which they show to be incomparable to Nash equilibrium. Their work captures some notion of coalition resistance in the sense that coalitions do not want to renegotiate to strategies that are themselves resilient to future renegotiations. Our protocol is not a threat-free Nash/renegotiation safe under the same resilience parameter — it is interesting to study what resilience parameters our protocol can tolerate under these notions. In fact, Threat-Free Nash and Renegotiation Safety have not been explored in a coalition setting before. For Renegotiation Safety, there may even be multiple parameters that matter, including the size of the coalition who proposes the renegotiated strategy, and how many non-negotiating players there are. It would also be an interesting future direction to explore the (in)feasiblity of threat-free or renegotiation-safe notions in the context of multi-party coin toss.

Recent efforts. More recently, there has been renewed interest in the connection of game theory and cryptography, partly due to the success of decentralized blockchains. Besides the work of Chung et al. [CGL+18] which provided direct inspiration of our work, the recent work of Chung, Chan, Wen, and Shi [CCWS21] suggested an alternative formulation of game-theoretically fair multi-party coin toss. Specifically, they consider the task of electing a leader among n players, where everyone is competing to get elected. Therefore, if a user gets elected, its utility is 1, else its utility is 0. Their formulation can be viewed as tossing an n-way dice whereas our formulation and that of Chung et al. consider a binary coin. Intriguingly, for the leader election formulation, it is indeed possible to achieve CSP-fairness under any number of corruptions, and thus Chung et al. [CCWS21] focus on understanding the round complexity of such protocols. Chung et al. also explore how to define *approximate* notions of game-theoretic fairness in a distributed protocol context, and they point out that further subtleties exist in defining an *approximate* notion, and thus they suggest new notions called sequential CSP fairness and sequential maximin fairness. These technicalities only pertain to approximate notions with non-negligible slack, and are *not* relevant for us since we consider (1-negligible)-fairness.

Other recent works, also inspired by blockchain applications, consider a financial fairness notion through the use of collateral and penalities [BK14, KMS+16, ADMM16, KB14, KVV16]. In comparison, the protocols in this paper can ensure game theoretic fairness even *without* the use of collateral or penalties if applied in blockchain contexts.

Relationship to RPD. Chung, Chan, Wen, and Shi [CCWS21] also show a connection between their approximate game-theoretic notion and the elegant RPD notion by Garay et al. [GKM+13, GKTZ15, GTZ15]. The same connection also applies to our notion. More specifically, the RPD framework models a meta-game, i.e., a Stackelberg game between the protocol designer and an attacker: the designer first picks a protocol Π, then the attacker can decide which coalition to corrupt and its strategy after examining this protocol Π. They want a solution concept that achieves a subgame perfect equilibrium in this Stackelberg meta-game, but consider classical-style utility functions related to breaking privacy or correctness. Essentially, Chung et al. [CCWS21] showed that the CSP-fairness notion can be an equivalent interpretation in the RPD framework if we alter their utility notion accordingly to match our notion. We refer the readers to Chung et al. [CCWS21] for a detailed statement and proof of this equivalence.

Other related works. Finally, we can also circumvent Cleve's impossibility of strongly fair (i.e., unbiasable) coin toss under corrupt majority by introducing a trusted setup, or introducing non-standard cryptographic assumptions such as Verifiable Delay Functions [BBF18, BBBF18]. In this paper, we focus on *the plain model without trusted setup, without any common reference string (CRS), and standard cryptographic hardness assumptions.*

2.4 Organization

The rest of the paper is organized as follows. Section 3 introduces definitions and notations. In Section 4 we present our upper bound result for CSP fairness. In Section 5 we give the CSP lower bound. The complete characterization of maximin fairness is given in Section 6. The formal proofs are deferred to the supplementary material.

3 Definitions

The model. In an n-party coin toss protocol, n players interact through pairwise private channels as well as a public broadcast channel. We assume that all communication channels are authenticated, i.e., messages always carry the true sender's identity. Without loss of generality, we assume the players are numbered $1, 2, \ldots, n$, respectively. We assume that the network is synchronous and the protocol proceeds in rounds. Each player has a publicly stated preference for either the bit 0 or the bit 1. We call the vector of players' preferences as *the preference profile*, denoted as $\mathcal{P}$. At the end of the protocol, the coin toss outcome is defined as a deterministic, polynomial-time function over *the set of public messages posted to the broadcast channel.* The utility function that we consider is defined as follows:

> **The utility function:** If the outcome agrees with a player's preference, the player obtains utility 1; else it obtains 0.
>
> The utility of a coalition $A \subset [n]$ is the sum of the utilities of all coalition members.

The protocol execution is parametrized with a security parameter λ, and we may assume that n is polynomially bounded in λ. We assume that the coalition A (also called the *adversary*) may perform a *rushing* attack: in any round r, it can wait for honest players (i.e., those not in A) to send messages, and then decide what round-r messages the corrupt players in A want to send.

Correctness. We let $\sigma^* = (\sigma_1^*, \ldots, \sigma_n^*)$ denote the strategy (the code) of the all honest execution. That is, σ_i^* can be viewed as the code that party P_i is supposed to run according to the protocol specifications. We say that the protocol is *correct* if, unless all players have the same preference (in which case we can simply output the preferred bit with probability 1), the coin toss outcome is some fixed $b \in \{0,1\}$ with probability at most $1/2 \pm \mathsf{negl}(\lambda)$ for some negligible function $\mathsf{negl}(\cdot)$.

Notations. For a coalition $A \subset [n]$, we let U_A denote the utility of the coalition. We let $\sigma^* = (\sigma_1^*, \ldots, \sigma_n^*)$ denote the strategy (the code) of the all honest execution. For a coalition $A \subset [n]$, we denote by $U_A(\sigma_A, \sigma_{-A}^*)$ the expected utility of all members in A where the members of A follow some σ_A and the members that are not in A follow the honest strategy σ_{-A}^*. We denote by $U_A(\sigma_A^*, \sigma_{-A}^*)$ the expected utility of all members in A where all parties follow the honest strategy. All executions are considered with respect to some utility function and some public preference profile $\mathcal{P}$.

CSP fairness. Recall that in CSP fairness we require that no coalition can increase its own expected utility no matter how it deviates from the prescribed strategy. This is formalized as follows:

Definition 3.1 (CSP-fairness [CGL+18])**.** *We say that a coin toss protocol σ^* satisfies* cooperative-strategy-proofness *(or CSP-fairness) against any for t-sized coalitions with respect to a preference profile $\mathcal{P}$, iff for all $A \subseteq [n]$ of cardinality at most t, any non-uniform probabilistic polynomial-time (p.p.t.) strategy σ'_A adopted by the coalition A, there is a negligible function* $\mathsf{negl}(\cdot)$*, such that*[2]

$$U_A(\sigma'_A, \sigma_{-A}^*) \le U_A(\sigma_A^*, \sigma_{-A}^*) + \mathsf{negl}(\lambda) \ .$$

Note that in this definition, if the coalition controls the same number of 0-supporters and 1-supporters, then we allow it to bias the output arbitrarily since it has no preference.

Maximin fairness. Maximin fairness requires that no coalition can harm any honest party. This is formalized as follows:

Definition 3.2. *We say that a coin-toss protocol σ^* satisfies* maximin fairness *for t-sized coalitions with respect to a preference profile $\mathcal{P}$, iff for any p.p.t. adversary $\mathcal{A}$ controlling at most t parties, there exists a negligible function $\mathsf{negl}(\cdot)$ such that, in an execution of the protocol involving the adversary $\mathcal{A}$, the expected utility of any honest party i is at least $U_i(\sigma^*) - \mathsf{negl}(\lambda)$, where $U_i(\sigma^*)$ is the expected utility of party i in an honest execution of the protocol with respect to $\mathcal{P}$.*

4 Upper Bound

Our starting point is the warmup protocol for semi-malicious adversary, as presented in Section 2.1, which leads to the following optimal resilience:

Case	k_0	k_1	t
If $n_1 \ge \frac{5}{2}n_0$	$\lfloor \frac{n_0}{2} \rfloor$	$n_1 - n_0$	$n_1 - \lfloor \frac{1}{2}n_0 \rfloor$
Otherwise	$\lfloor \frac{n_0}{2} \rfloor$	$\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor$	$\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{n_0}{2} \rceil$

[2]Like earlier works [PS17, CGL+18, GKM+13, GKTZ15, GTZ15, HT04, KN08, ADGH06, OPRV09, AL11, ACH11], our CSP-fair notion considers the deviation of *a single* coalition. Such a definitional approach is standard and dominant in the game theory literature, and the philosophical motivation is that the honest protocol would then become an *equilibrium* such that no coalition (of a certain size) would be incentivized deviate. In fact, many earlier works (including the standard Nash equilibrium notion) would even consider deviation of a single individual rather than a coalition.

A corner case of $n_0 = n_1 =$ odd. It turns out that the above solution for t is optimal (even for semi-malicious coalitions) in light of our lower bound in Section 5, except for the corner case $n_0 = n_1 = odd$. This is because the above conditions (C1), (C2) and (C3) are slightly too stringent — in cases when the adversary corrupts exactly the same number of 0-supporters and 1-supporters, the coalition is actually indifferent (i.e., have no preference). In such cases, the coalition is allowed to bias the coin towards either direction, and therefore we do not need the above conditions to hold. We defer a detailed analysis of this corner case to supplementary material C.4. Taking this corner case into account, we obtain that the number of corruptions that can be tolerated is:

Case	k_0	k_1	t
If $n_1 \geq \frac{5}{2}n_0$	$\lfloor \frac{n_0}{2} \rfloor$	$n_1 - n_0$	$n_1 - \lfloor \frac{1}{2}n_0 \rfloor$
If $n_1 = n_0 =$ odd	$\lfloor \frac{n_0}{2} \rfloor$	$\lfloor \frac{1}{2}n_1 \rfloor$	$n_1 + 1$
Otherwise	$\lfloor \frac{n_0}{2} \rfloor$	$\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor$	$\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{n_0}{2} \rceil$

Due to our lower bound in Section 5, the above resilience parameter is optimal for CSP fairness, even for semi-malicious corruptions.

4.1 Our Final Protocol for Malicious Coalitions

We now present our final construction ensures CSP-fairness against malicious coalitions that may deviate arbitrarily from the prescribed protocol.

4.1.1 Maliciously Secure $\mathsf{HalfToss}^b$ Sub-Protocol

To lift the warmup protocol to malicious security, the main challenge is how to realize a counterpart of the $\mathsf{HalfToss}^b$ protocol for the malicious corruption model. Recall that in the semi-malicious model, we relied on the players themselves to send heartbeats to identify which players have aborted (see supplementary material B for formal definitions). In this malicious model, we can no longer rely on such self-identification because players can lie. In a corrupt majority model, we also cannot easily take majority vote to determine who remains online and honest.

Our final solution relies on MPC with identifiable abort [GMW87,IOZ14] which can be accomplished assuming the existence of Oblivious Transfer (OT). Recall that in MPC with identifiable abort, either the players successfully evaluate some ideal functionality, or if the protocol aborted, then all honest players receive the identity of an offending player. The idea is that the honest players can now kick out the offending player and retry, until the protocol succeeds in producing output.

Specifically, we will replace our earlier $\mathsf{HalfToss}^b[k]$ sub-protocol with the following maliciously secure counterpart, in which the b-supporters participate and the $(1-b)$-supporters observe.

Protocol 4.1: $\mathsf{HalfToss}^b[k]$ **sub-protocol with malicious security**

Sharing phase.

1. Initially, define the active set $\mathcal{O} := \mathcal{P}_b$. Repeat the following until success:
   (a) The active set $\mathcal{O}$ use MPC with identifiable abort to securely compute the ideal functionality $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ to be described below (Functionality 4.2).
   (b) If the protocol aborts, then every honest player obtains the identity of a corrupt player $j^* \in \mathcal{O}$. Remove j^* from $\mathcal{O}$.

2. At this moment, each player $i \in \mathcal{O}$ has obtained the tuple $(\mathsf{vk}, [s]_i, [r]_i, [\mathsf{com}]_i, \sigma_i, \sigma'_i)$ from $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$.

Vote phase.

1. Each player posts vk to the broadcast channel — henceforth this is also called a vote for vk. Let vk' be the verification key that has gained the most number of votes, breaking ties arbitrarily.
2. If vk' has not gained at least $k+1$ votes, declare that the vote phase failed and return. Else, if $\mathsf{vk}' = \mathsf{vk}$, then player i posts $[\mathsf{com}]_i$ and σ_i to the broadcast channel.
3. Everyone gathers all $([\mathsf{com}]_j, \sigma_j)$ pairs posted to the broadcast channel such that σ_j is a valid signature of $[\mathsf{com}]_j$ under vk'. If there are at least $k+1$ such tuples and all shares $[\mathsf{com}]_j$ reconstruct uniquely to the value com, then record the reconstructed commitment com. Else we say that the vote phase failed.

Reconstruction phase.

1. If the vote phase failed, output the reconstructed value $\perp$. Else, continue with the following.
2. For each player $i \in \mathcal{O}$, if $\mathsf{vk}' = \mathsf{vk}$, then post to the broadcast channel the tuple $([s]_i, [r]_i, \sigma'_i)$.
3. Every player does the following: gather all tuples $([s]_j, [r]_j, \sigma'_j)$ posted to the broadcast channel such that σ'_j is a valid signature for $([s]_j, [r]_j)$ under vk'. If all such $([s]_j, [r]_j)$ tuples reconstruct to a unique value (s, r) and moreover, (s, r) is a valid opening of com, then output the reconstructed value s. Else output $\perp$ as the reconstructed value.

Functionality 4.2: The $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ ideal functionality

1. Sample $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Sig.KeyGen}(1^\lambda)$ where $\mathsf{Sig} := (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$ denotes a signature scheme.
2. Sample $s \xleftarrow{\$} \{0,1\}$, and randomness $r \in \{0,1\}^\lambda$, let $\mathsf{com} := \mathsf{Commit}(s, r)$.
3. Use a $(k+1)$-out-of-$|\mathcal{O}|$ Shamir secret sharing scheme to split the terms (s, r) and com into $|\mathcal{O}|$ shares, denoted $\{[s]_i, [r]_i, [\mathsf{com}]_i\}_{i \in \mathcal{O}}$, respectively. Let $\sigma_i := \mathsf{Sig.Sign}(\mathsf{sk}, [\mathsf{com}]_i)$ and $\sigma'_i := \mathsf{Sig.Sign}(\mathsf{sk}, ([s]_i, [r]_i))$ for $i \in \mathcal{O}$.
4. Each player in $\mathcal{O}$ receives the output $(\mathsf{vk}, [s]_i, [r]_i, [\mathsf{com}]_i, \sigma_i, \sigma'_i)$.

The above maliciously secure $\mathsf{HalfToss}^b[k]$ protocol satisfies the following properties:

- *Binding.* If the vote phase does not fail, then the messages on the broadcast channel in the sharing and vote phases uniquely define a coin $s \neq \perp$ such that reconstruction must either output s or $\perp$.
- *Knowledge threshold.* We now have a computationally secure version of the knowledge threshold property.
  - If at least $k+1$ number of b-supporters are corrupt, then the coalition can bias coin values s that the sharing and vote phases uniquely bind to (assuming that the voting phase did

not fail). Specifically, if the coalition controls $k+1$ number of b-supporters, it can decide whether to abort $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ after seeing the corrupt players' shares $\{[s]_j\}_{j\in A}$ where $A \subset [n]$ denotes the coalition. If it controls $\max(k+1, n_b/2)$ number of b-supporters, it can control the verification key vk' and thus alter the coin s the sharing and vote phases bind to as well.

- If fewer than $k+1$ number of b-supporters are corrupt, then the coalition's view at the end of the voting phase is computationally independent of the coin value s that the sharing and vote phases bind to. More formally, either the vote phase fails, or there exists a p.p.t. simulator Sim such that:
$$(s, \mathsf{view}_A) \approx_c (\mathsf{Uniform}, \mathsf{Sim}(1^\lambda))$$
where s denotes the unique coin value that the sharing phase and vote phases bind to, view_A denotes the coalition's view at the end of the vote phase, $\mathsf{Uniform}$ denotes a random bit sampled from $\{0,1\}$, and $\approx_c$ denotes computational indistinguishability.

- *Liveness threshold.* If the coalition controls at least $\min(n_b - k, n_b/2)$ number of b-supporters, it can cause the reconstruction to output $\perp$. On the other hand, if the coalition controls fewer than $\min(n_b - k, n_b/2)$ number of b-supporters, then the reconstruction phase must succeed.

In comparison with the earlier semi-malicious version, the knowledge threshold and liveness threshold property now become weaker. One relaxation is the computational security relaxation in the knowledge threshold property whereas previously in the semi-malicious version, the property was information theoretic. Another relaxation is that the thresholds for the two properties have changed. Now, the coalition may be able to control the coin value and hamper reconstruction with a smaller threshold.

4.1.2 Final Protocol

Our final protocol is described as follows:

Protocol 4.3: Final protocol with malicious security

Sharing phase.

1. 0-supporters run the sharing phase of $\mathsf{HalfToss}^0[k_0]$.
2. 1-supporters run the sharing phase of $\mathsf{HalfToss}^1[k_1]$.

Vote phase. *(The order of the two instances is important.)*

1. 1-supporters run the vote phase of $\mathsf{HalfToss}^1[k_1]$.
2. 0-supporters run the vote phase of $\mathsf{HalfToss}^0[k_0]$.

Reconstruction phase. *(The order of the two instances is important.)*

1. 0-supporters run the reconstruction phase of $\mathsf{HalfToss}^0[k_0]$, and let its outcome be s_0 if reconstruction is successful. In case the reconstruction outputs $\perp$, then let $s_0 := 0$.
2. 1-supporters run the reconstruction phase of $\mathsf{HalfToss}^1[k_1]$. If the reconstruction phase outputs $\perp$, then output 0 as the final coin value. Else let s_1 be the reconstructed value, and output $s_0 + s_1$ as the final coin value.

In the above, the order of the two instances in the vote and reconstruction phases is important due to a similar reason as in the semi-malicious version.

Setting aside the computational security issue for the time being (which can be formally dealt with using a standard computational reduction argument), in light of the properties for our maliciously secure $\mathsf{HalfToss}^b$ sub-protocol, we can now rewrite the earlier (C1), (C2), (C3) conditions as follows (recall that t_0 and t_1 are number of corrupted 0-supporters and 1-supporters, respectively):

(C1*) The coalition cannot control both s_0 and s_1, i.e., the coin values the sharing and vote phases of $\mathsf{HalfToss}^0[k_0]$ and $\mathsf{HalfToss}^1[k_1]$ bind to (assuming that it did not fail), respectively. This means that if the coalition controls at least $k_b + 1$ number of b-supporters, then it does not have enough corruption budget to control $k_{1-b} + 1$ number of $(1 - b)$-supporters.

(C2*) If the coalition controls the s_1 coin, i.e., it controls at least $k_1 + 1$ number of 1-supporters, then it cannot hamper the reconstruction of the coin s_0 due to the corruption budget. That is, the coalition must control fewer than $\min(n_0 - k_0, n_0/2)$ number of 0-supporters.

(C3*) If the coalition controls at least $\min(n_1 - k_1, n_1/2)$ number of 1-supporters such that it can cause the reconstruction of s_1 to fail, then the coalition must prefer 1 or is indifferent to the outcome — in other words, either $n_0 \le t_1$ or $t \le 2t_1$ ($t_0 \le t_1$ and so $t = t_0 + t_1 \le 2t_1$).

These conditions can be rewritten as the following expressions:

Parameter Constraints 4.4 (malicious version)**.**
Assume: $0 \le k_0 \le n_0,\ 0 \le k_1 \le n_1$

(C1*) $t \le k_0 + k_1 + 1$,

(C2*) $t < k_1 + 1 + \min(n_0 - k_0, n_0/2)$,

(C3*) if $\min(n_1 - k_1, \lceil \frac{n_1}{2} \rceil) < n_0$, then $t \le 2 \cdot \min(n_1 - k_1, \lceil \frac{n_1}{2} \rceil)$.

One can verify that any k_0, k_1, t that satisfy (C1*), (C2*), (C3*) must also satisfy the earlier conditions (C1), (C2) and (C3). This means that the new malicious version of the protocol cannot tolerate more corruptions than the semi-malicious version. Intriguingly, it turns out that there exists a choice of k_0 and k_1 that maximizes t for conditions (C1), (C2) and (C3), such that the same (k_0, k_1, t) also satisfy (C1*), (C2*), and (C3*). This means that our maliciously secure protocol can achieve the same resilience parameter as the semi-malicious version.[3] More specifically, there exists a choice satisfying $k_0 = \lceil (n_0 - 1)/2 \rceil$ and $k_1 \ge \lfloor n_1/2 \rfloor$ such that t is maximized for conditions (C1), (C2) and (C3). One can then verify that that as long as $k_0 = \lceil (n_0 - 1)/2 \rceil$ and $k_1 \ge \lfloor n_1/2 \rfloor$, a feasible solution (k_0, k_1, t) for conditions (C1), (C2) and (C3) would also be a feasible solution for conditions (C1*), (C2*), and (C3*).

Just like the earlier semi-malicious setting, the above constraints (C1*), (C2*), and (C3*) are in fact slightly too stringent; thus, for the special case $n_0 = n_1 = odd$, the resulting solution of t would have a gap of 1 away from optimal. This gap can be bridged by observing that if the same number of 0-supporters and 1-supporters are corrupt, the coalition would then be indifferent, and it would be fine if the coalition could bias the coin towards either direction. We defer a detailed analysis of the corner case $n_0 = n_1 = odd$ to supplementary material C.4.

Formal proofs. In supplementary material C, we formally prove the following theorem (Theorem 1.1 in the introduction):

[3]Note that since our lower bound holds even for fail-stop adversaries, only when the malicious version matches the resilience of the semi-malicious version can it be tight.

Theorem 4.5 (Upper bound)**.** *Assume the existence of Oblivious Transfer (OT), and without loss of generality, assume that $n_1 \geq n_0 \geq 1$, and $n_0+n_1 > 2$. Protocol 4.3 is CSP-fair coin toss protocol which tolerates up to t-sized non-uniform p.p.t. malicious coalitions where*

$$t := \begin{cases} n_1 - \lfloor \frac{1}{2} n_0 \rfloor, & \text{if } n_1 \geq \frac{5}{2} n_0; \\ \lfloor \frac{2}{3} n_1 - \frac{1}{6} n_0 \rfloor + \lceil \frac{1}{2} n_0 \rceil + 1 = n_1 + 1, & \text{if } n_1 = n_0 = \text{odd}; \\ \lfloor \frac{2}{3} n_1 - \frac{1}{6} n_0 \rfloor + \lceil \frac{1}{2} n_0 \rceil, & \text{otherwise.} \end{cases}$$

5 Lower Bound

5.1 Parameter Constraints

We now show that, if the parameters α_0, α_1 and t satisfy the following constraints, then for any coin toss protocol among n_0 number of 0-supporters and n_1 number of 1-supporters that achieves CSP fairness against a coalition of size up to t,[4] it's corresponding three-party coin toss protocol (after partition with respect to α_0 and α_1 as specified), must satisfy the lone-wolf condition (LBC1), the wolf-minion condition (LBC2), as well as the T_2 equity condition (LBC3) simultaneously.

Parameter Constraints 5.1 (Constraint system for lower bound proof)**.**

Non-negative	*Lone-wolf*	*Wolf-minion*	*T_2-equity*
$0 \leq \alpha_0 \leq \frac{1}{2} n_0$	$\alpha_1 + 1 \leq n_0$	$n_0 - \alpha_0 < n_1 - \alpha_1$	$1 \leq \alpha_0$
$0 \leq \alpha_1 \leq \frac{1}{2} n_1$	$\alpha_0 + 1 \leq n_1$	$n_0 + n_1 - \alpha_0 - \alpha_1 \leq t$	$1 \leq \alpha_1$
	$\alpha_0 + \alpha_1 \leq t$		$3 \leq t$
	$2\alpha_0 + 1 \leq t$		$1 \leq n_0 + n_1 - 2\alpha_0 - 2\alpha_1 \leq t$
	$2\alpha_1 + 1 \leq t$		

In the above set of conditions, the first set (i.e., non-negative) makes sure that the number of 0-supporters and 1-supporters in each partition is non-negative. The next three sets of conditions are required to prove the corresponding three conditions, respectively. We show how the conditions lead to this set of parameter constraints in Section 5.2. Then, given any fixed n_0 and n_1, it suffices to solve for the best partition strategy (i.e., choice of α_0 and α_1) that minimizes t, and this minimal choice of t gives rise to our lower bound in light of Theorem 2.5. We explore that in Section 5.3. It turns out that the minimal t value satisfying the above constraint system coincides with our upper bound, and in particular, with Eq. (1).

5.2 Constraint System Implies the Lone-Wolf, Wolf-Minion, and T_2-Equality Conditions

Below we focus on proving that the three lower bound conditions hold provided the constraint system.

Lemma 5.2 (Generalized lone-wolf lemma)**.** *Let Π be a protocol that is CSP-fair against any non-uniform p.p.t., fail-stop coalition of size t. If α_0, α_1 and t satisfy the non-negative and lone-wolf constraints in Parameter Constraints 5.1, then Π satisfies the lone-wolf condition (LBC1).*

[4]Our main lower bound theorem, i.e., Theorem 1.2, states the impossibility for coalitions of size $t+1$ or greater. For convenience, in this section, we switch the notation to t rather than $t+1$.

Proof. Suppose for the sake of contradiction that the long-wolf condition is violated, i.e., there exists a non-uniform p.p.t. fail-stop adversary $\mathcal{A}$ corrupting only $\mathcal{S}_1$ (the same argument holds for $\mathcal{S}_3$) that can bias the output towards $b \in \{0,1\}$ by a non-negligible amount. We show that then Π is not CSP fair against t fail-stop adversaries. There are two cases:

- If $\alpha_b > \alpha_{1-b}$ then $\mathcal{S}_1$ (resp. $\mathcal{S}_3$) prefers b. The number of parties in $\mathcal{S}_1$ is $\alpha_0 + \alpha_1$. According to the lone-wolf constraints in Parameter Constraints 5.1 we have that $\alpha_0 + \alpha + 1 \le t$ and thus this coalition is supposed to be tolerated.

- If $\alpha_b \le \alpha_{1-b}$, consider the following coalition in the CSP-fair protocol. The coalition corrupts $\mathcal{S}_1$ and in addition $\alpha_{1-b} + 1 - \alpha_b$ number of b-supporters outside $\mathcal{S}_1$. From the lone-wolf constraint in Parameter Constraints 5.1, we have that $n_b \ge \alpha_{1-b} + 1$. This implies that the number of b-supporters outside $\mathcal{S}_1$ is $n_b - \alpha_b \ge \alpha_{1-b} + 1 - \alpha_b$. Then, this coalition consists of α_{1-b} number of $(1-b)$-supporters and $\alpha_{1-b} + 1$ number of b-supporters. From the lone-wolf constraint in Parameter Constraints 5.1 we have that $2\alpha_{1-b} + 1 \le t$. Then, this coalition contains less than t parties and it prefers b. If there exists a fail-stop adversary in the three-party protocol that controls $\mathcal{S}_1$ and can bias towards b, then this coalition in the CSP-protocol can also bias towards b. Note that the additional parties in the coalition that are outside of $\mathcal{S}_1$ act honestly and are used just to change the preference of the coalition, i.e., it is enough to consider the existence of a fail-stop adversary that corrupts only one party in the corresponding three-party protocol.

□

Lemma 5.3 (Generalized wolf-minion lemma). *Let Π be a protocol that is CSP-fair against any non-uniform p.p.t., fail-stop coalition of size t. If α_0, α_1 and t satisfy the non-negative and wolf-minion constraints in Parameter Constraints 5.1, then Π satisfies the wolf-minion condition (LBC2).*

Proof. The non-negative constraints make sure that the number of parties in $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ are non-negative, as $\mathcal{S}_2$ contains $(n_0 - 2\alpha_0)$ number of 0-supporters and $(n_1 - 2\alpha_1)$ number of 1-supporters. If the wolf-minion constrains hold, then the coalition of $\mathcal{S}_1$ and $\mathcal{S}_2$ (or $\mathcal{S}_3$ and $\mathcal{S}_2$) prefers 1 since in total it contains $n_0 - \alpha_0$ number of 0-supporters and $n_1 - \alpha_1$ number of 1-supporters and according to the constraints, $n_1 - \alpha_1 > n_0 - \alpha_0$. Moreover, the number of parties in this coalition is $n_1 + n_0 - \alpha_0 - \alpha_1$, which is at most t according to the condition. Therefore, any fail-stop adversary corrupting $\mathcal{S}_1$ and $\mathcal{S}_2$ (or $\mathcal{S}_3$ and $\mathcal{S}_2$) cannot bias the output towards 1 by a non-negligible amount, according to the CSP fairness of Π against t fail-stop adversaries. This means that the protocol Π satisfies the wolf-minion condition. □

Lemma 5.4 (Generalized T_2-equity lemma). *Let Π be a protocol that is CSP-fair against any non-uniform p.p.t., fail-stop coalition of size t. If α_0, α_1 and t satisfy the non-negative and the T_2-equity constraints in Parameter Constraints 5.1, then protocol Π satisfies the T_2-equity condition (LBC3). That is, for all but a negligible fraction of $\mathcal{S}_2$'s randomness T_2, $|f(T_2) - \frac{1}{2}|$ is negligible.*

Proof. By correctness of the protocol, $\mathbb{E}_{T_2}[f(T_2)] = \frac{1}{2}$. Note that T_2 consists of the randomness of all players in $\mathcal{S}_2$, we can view T_2 as a vector $\{t_Q\}_{Q \in \mathcal{S}_2}$ where t_Q is player Q's randomness. For any fixed party Q in $\mathcal{S}_2$, consider a protocol Π^Q that is same with Π except that Q aborts at the very beginning of the protocol and all other parties behave honestly. Let $g^Q(T_2)$ be the expected output of Π^Q conditioned on $\mathcal{S}_2$'s randomness T_2.

Claim 5.5. *For any $Q \in \mathcal{S}_2$, $|\mathbb{E}_{T_2}[g^Q(T_2)] - \frac{1}{2}|$ is negligible.*

Proof. Suppose for the sake of contradiction that the claim is not true. Then this single aborting party Q can bias the outcome of Π towards $b \in \{0, 1\}$ by a non-negligible amount. This violates the CSP-fairness of the n-party protocol: Consider a coalition that consists of the Q party and two b-supporters. This coalition prefers the coin b, and can bias towards it by having Q abort at the very beginning of the protocol Π. Note that according to T_2-equity constraints in Parameter Constraints 5.1, $\alpha_b \geq 1$, which implies that there are at least two b-supporters outside $\mathcal{S}_2$. Moreover, the size of the coalition is 3, and thus we require that $t \geq 3$. $\square$

Claim 5.6. *For any Q in $\mathcal{S}_2$, for all but a negligible fraction of T_2, $|g^Q(T_2) - f(T_2)|$ is also negligible.*

Proof. Note that for all but a negligible fraction of T_2, $|\mathbb{E}_{T_2}[g^Q(T_2) - f(T_2)]| = |\mathbb{E}_{T_2}[g^Q(T_2)] - \mathbb{E}_{T_2}[f(T_2)]| = |\mathbb{E}_{T_2}[g^Q(T_2)] - \frac{1}{2}|$ is negligible. Suppose that there exists a non-negligible fraction of T_2 such that $f(T_2) - g^Q(T_2)$ is positive and non-negligible, then there must also exists a non-negligible fraction of T_2 such that $g^Q(T_2) - f(T_2)$ is positive and non-negligible. This indicates that for a non-negligible fraction of T_2, Q can bias the output of Π towards 1 (or 0) by a non-negligible amount by aborting at the beginning of the protocol.

Suppose that $\mathcal{S}_2$ prefers 1 (the same argument holds if $\mathcal{S}_2$ prefers 0). Consider an adversary $\mathcal{A}^*$ that receives a polynomial $p(\cdot)$ as an advice where $p(\cdot)$ is chosen such that for a non-negligible fraction of T_2, $g^Q(T_2) - f(T_2) \geq 1/p(\lambda)$. $\mathcal{A}^*$ corrupts $\mathcal{S}_2$ and acts as follows:

- $\mathcal{A}^*$ randomly samples a T_2.
- $\mathcal{A}^*$ repeats the following for $p^2(\lambda)$ times: $\mathcal{A}^*$ samples T_1 and T_3 for $\mathcal{S}_1$ and $\mathcal{S}_3$ and simulates an honest execution with the randomness T_1, T_2, T_3. $\mathcal{A}^*$ also simulates an execution in which Q always aborts at the beginning of the protocol. Then $\mathcal{A}^*$ gets estimates of $\widetilde{g}^Q(T_2)$ and $\widetilde{f}(T_2)$.
- If $\widetilde{g}^Q(T_2) > \widetilde{f}(T_2)$, $\mathcal{A}^*$ instructs Q to abort at the very beginning of the protocol. Otherwise it follows the honest execution.

Note that for any T_2 such that $g^Q(T_2) - f(T_2) \geq \frac{1}{p(\lambda)}$, by the Chernoff bound, except with a negligible probability, it must be that $\widetilde{g}^Q(T_2) > \widetilde{f}(T_2)$. Therefore, $\mathcal{A}^*$ can bias the output of Π towards 1 by a non-negligible amount. This breaks the CSP fairness of Π since, according to the T_2-equity constraint in Parameter Constraints 5.1, $\mathcal{S}_2$, which contains $n_0 + n_1 - 2\alpha_0 - 2\alpha_1$ contains parties which is at most t, and it prefers 1. Therefore, for all but a negligible fraction of T_2, $|g^Q(T_2) - f(T_2)|$ is negligible. $\square$

For any fixed $Q \in \mathcal{S}_2$, for any pair of T_2 and T_2' that only differ in Q's randomness, it must be that $g^Q(T_2) = g^Q(T_2')$. Let ℓ denote the length of T_2, we have:

Claim 5.7. *For any fixed $i \in [\ell]$, for all but a negligible fraction of T_2, $|f(T_2) - f(\widetilde{T}_2^i)|$ is negligible, where $\widetilde{T}_2^i$ is same as T_2 except with the i-th bit flipped.*

Proof of Claim 5.7. Suppose that the i-th bit is contributed by party $Q \in \mathcal{S}_2$. For any polynomial $p(\cdot)$, define bad_1^p to be the event $|f(T_2) - g^Q(T_2)| \geq \frac{1}{p(\lambda)}$, and bad_2^p to be the event $|f(\widetilde{T}_2^i) - g^Q(\widetilde{T}_2^i)| \geq \frac{1}{p(\lambda)}$. Since for all but a negligible fraction of T_2, $|f(T_2) - g^Q(T_2)|$ is negligible, the probability that bad_1^p happens is negligible. The probability that bad_2^p happens is also negligible. Thus by a union bound, the probability that both bad_1^p and bad_2^p do not happen is $1 - \mathsf{negl}(\lambda)$ for some negligible function $\mathsf{negl}(\cdot)$. This indicates that for any polynomial $p(\cdot)$, $|f(T_2) - f(\widetilde{T}_i)| \leq |f(T_2) - g^Q(T_2)| + |f(\widetilde{T}_2^i) - g^Q(\widetilde{T}_2^i)| \leq \frac{2}{p(\lambda)}$ with probability $1 - \mathsf{negl}(\lambda)$. The claim thus follows. $\square$

Claim 5.8. *Pick a random T_2 and a random T_2'. Then except with a negligible probability over the random choice of T_2 and T_2', $|f(T_2) - f(T_2')|$ is negligible.*

Proof. Pick a random T_2 and a random T_2', we define hybrids T^i, $i = 0, \dots, \ell + 1$ as follows:

$$T^i = \{t_1, \dots, t_i, t'_{i+1}, \dots, t'_\ell\},$$

where t_i is the i-th bit of T_2 and t'_i is the i-th bit of T_2'. Then, $T^0 = T_2'$ and $T^\ell = T_2$. For any fixed polynomial $p(\cdot)$, define bad_i^p to be the event that $|f(T^i) - f(T^{i+1})| \geq \frac{1}{p(\lambda)}$. Note that the marginal distribution of T^i is uniform, for any polynomial $p(\cdot)$, the probability that bad_i^p happens is negligible over the choice of T_2 and T_2', according to Claim 5.7. Therefore, for any $p(\cdot)$, by the union bound, the probability that none of bad_i^p happens is $1 - \mathsf{negl}(\lambda)$ for some negligible function $\mathsf{negl}(\cdot)$. Observe that for any fixed polynomial $p(\cdot)$, if none of the events bad_i^p happen, then $|f(T_2) - f(T_2')| \leq \frac{\ell+1}{p(\lambda)}$ by triangle inequality. Hence, for any random T_2 and any random T_2', $|f(T_2) - f(T_2')|$ is negligible except with a negligible probability over the random choices over T_2 and T_2'. □

Together with the fact that $\mathbb{E}_{T_2}[f(T_2)] = \frac{1}{2}$, we have that for all but a negligible fraction of T_2, $|f(T_2) - \frac{1}{2}|$ is negligible. Otherwise if for some polynomial $p(\cdot), q(\cdot)$, there exists $1/p(\lambda)$ fraction of T_2 such that $f(T_2) - \frac{1}{2} \geq 1/q(\lambda)$, then there must exist $1/p'(\lambda)$ fraction of T_2 such that $\frac{1}{2} - f(T_2) \geq 1/q'(\lambda)$ for some polynomial $p'(\cdot), q'(\cdot)$. Then for any random T_2 and T_2', with a non-negligible probability, $|f(T_2) - f(T_2')| \geq 1/q(\lambda) + 1/q'(\lambda)$, which violates the above conclusion.

To conclude, for all but a negligible fraction of T_2, $|f(T_2) - \frac{1}{2}|$ is negligible.

□

5.3 Minimizing t Subject to Constraints

We prove the following Lemma in supplementary material D.1.

Lemma 5.9 (Solving the constraint system and minimizing t)**.** *For Parameter Constraint 5.1, the parameter t is minimized when α_0 and α_1 are chosen as follows, and the corresponding t is:*

Case	α_0	α_1	t
$n_1 \geq \frac{5}{2}n_0$, $n_0 \geq 2$	$\lfloor \frac{1}{2}n_0 \rfloor$	$n_0 - 1$	$n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1$
$2 \leq n_0 < n_1 < \frac{5}{2}n_0$	$\lfloor \frac{1}{2}n_0 \rfloor$	$\lceil \frac{1}{3}n_1 + \frac{1}{6}n_0 \rceil - 1$	$\lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$
$2 \leq n_0 = n_1$	$\lfloor \frac{1}{2}n_0 \rfloor$	$\lfloor \frac{1}{2}n_0 \rfloor - 1$	$2\lceil \frac{1}{2}n_0 \rceil + 1$

Note that for the case $t = 2\lceil \frac{1}{2}n_0 \rceil + 1$, this expression is equal to $\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{1}{2}n_0 \rceil + 1$ when $n_0 = n_1$ is even, and is equal to $n_0 + 2$ when when $n_0 = n_1$ is odd.

6 Complete Characterization of Maximin Fairness

In this section we give a complete characterization of the maximin fairness defined by Chung et al. [CGL+18]. Intuitionally, maximin fairness requires that a corrupted coalition cannot harm the expected reward of any honest party, compared to an all-honest execution. This definition is formalized in Definition 3.2.

6.1 Lower Bound

Unlike CSP-fairness, maximin-fairness is impossible under a broad range of parameters. More specifically, we prove the following theorem, which says that unless $n_0 = 1$ and $n_1 = odd$, for maximin fairness, we cannot tolerate *fail-stop* coalitions of half of the parties or more. The special case $n_0 = 1$ and $n_1 = odd$ is slightly more subtle. Chung et al. [CGL+18] showed that for the special case $n_0 = 1$, it is indeed possible to achieve maximin fairness against all but one *fail-stop* corruptions. We prove that for $n_0 = 1$, we cannot tolerate *semi-malicious* coalitions that are majority in size.

Theorem 6.1 (Lower bound for maximin fairness)**.** *Without loss of generality, assume that $n_1 \geq n_0 \geq 1$ and $n_0 + n_1 > 2$. Then there does not exist a maximmin-fair n-party coin toss protocol that can:*

$$\begin{array}{ll} \textit{tolerate } \textbf{fail-stop} \textit{ coalition of size } t \geq \lceil \frac{1}{2}(n_0 + n_1) \rceil & \textit{for } n_0 \geq 2 \\ \textit{tolerate } \textbf{semi-malicious} \textit{ coalition of size } t \geq \lceil \frac{1}{2} n_1 \rceil + 1 & \textit{for } n_0 = 1 \end{array}$$

Proof sketch. For the case where $n_0 \geq 2$, we show that if there exists a coin toss protocol that achieves maximin-fairness against $\lceil \frac{1}{2}(n_0 + n_1) \rceil$ fail-stop adversaries, then we can construct a two-party protocol that violates Cleve's lower bound [Cle86]. Consider any preference profile that contains at least two 0-supporters and in which $n_1 \geq n_0$. Then, we partition the 0-supporters and 1-supporters as evenly as possible into two partitions, and the two party protocol is simply an emulation of the n-party protocol with respect to this preference profile. Each party internally emulates the execution of all parties it runs in the outer protocol, in a similar manner as in Section 5. Since $n_1 \geq n_0 \geq 2$, each partition must contain at least one 0-supporter and at least one 1-supporter. By maximin fairness, if either partition is controlled by a non uniform p.p.t. adversary $\mathcal{A}$, it should not be able to bias the outcome towards either 0 or 1 by a non-negligible amount — otherwise if $\mathcal{A}$ was able to bias the coin towards $b \in \{0, 1\}$, it would be able to harm an individual b-support in the other partition. Now, if we view the coin toss protocol as a two-party coin toss protocol between the two partitions, the above requirement would contradicts Cleve's impossibility result [Cle86].

For the case where $n_0 = 1$, the proof is similar to that of the CSP-fairness. We partition the players into three partitions: $\mathcal{S}_1$ and $\mathcal{S}_3$ each contains half of 1-supporters and $\mathcal{S}_2$ contains the single 0-supporter. We can show that if a coin toss protocol is maximin-fair against $\lceil \frac{1}{2} n_1 \rceil + 1$ fail-stop adversaries, then it should satisfy the wolf-minion condition, the lone-wolf condition and the T_2-equity condition simultaneously. The full proof is deferred to supplementary material E.1. □

6.2 Upper Bound

As mentioned, except for the special case $n_0 = 1$ and $n_1 = odd$, for maximin fairness, we cannot hope to tolerate half or more fail-stop corruptions. However, if majority are honest, we can simply run honest-majority MPC with guaranteed output delivery [GMW87, RB89].

Therefore, the only non-trivial case is when $n_0 = 1$ and $n_1 = odd$. Chung et al. [CGL+18] showed that for $n_0 = 1$, there exists a coin toss protocol that achieves maximin-fairness against up to $(n - 1)$ *fail-stop* adversaries. Here, we construct a maximin-fair coin toss protocol tolerates exactly half or fewer *malicious* corruptions.

In our protocol, first, the single 0-supporter commits to a random coin, and moreover, the 1-supporters jointly toss a coin s_1 such that the outcome is secret shared among the 1-supporters. Only if $\lceil n_1/2 \rceil$ number of 1-supporters get together, can they learn s_1, influence the value of s_1, or hamper its reconstruction later. Next, the 1-supporters reconstruct the secret-shared coin s_1. If the reconstruction fails, the reconstructed value is set to a canonical value $s_1 := 0$. Finally, the single 0-supporter opens its commitment and let the opening be s_0. If the single 0-supporter aborts any time during the protocol, the outcome is declared to be 1. Else, the outcome is declared to be $s_0 + s_1$. More formally, the protocol is as below.

Protocol 6.2: Protocol for maximin-fairness: special case when $n_0 = 1$ and $n_1 = odd$

1. The single 0-supporter randomly choose $s_0 \xleftarrow{\$} \{0, 1\}$ and compute the commitment $\mathsf{com} = \mathsf{Commit}(s_0, r)$ with some randomness $r \in \{0, 1\}^\lambda$. It then sends the commitment com to the broadcast channel. If the 0-supporter fails to send the commitment, set $s_0 = \perp$.
2. The 1-supporters run an honest-majority MPC with guaranteed output delivery to toss a coin s_1. Each player $i \in \mathcal{P}_1$ (the set of 1-supporters) receives $\widetilde{s}_i$ as the output of the MPC.
3. Every 1-supporter $i \in \mathcal{P}_1$ posts the output $\widetilde{s}_i$ it receives to the broadcast channel. Let s_1 be the majority vote. If no coin gains majority vote, set $s_1 = 0$.
4. The 0-supporter opens its coin s_0. If it fails to open the coin correctly, set $s_0 = \perp$.
5. If $s_0 = \perp$, output 1. Otherwise, output $s_0 \oplus s_1$.

Observe that if the single 0-supporter is honest, then we need to make sure that the coalition cannot bias the coin towards either direction; however, in this case, since the 0-supporter is guaranteed to choose a random coin and open it at the end, this can be ensured. If, on the other hand, the single 0-supporter is corrupt, then we only need to ensure that the coalition cannot bias the coin towards 0. We may therefore assume that the single 0-supporter does not abort because otherwise the outcome is just declared to be 1. Further, in this case, the coalition only has budget to corrupt $\lfloor n_1/2 \rfloor$ number of 1-supporters, which means that we have honest majority in 1-supporters. Therefore, if the 0-supporter does not abort, then the outcome will be a uniformly random coin.

This gives rise to the following theorem, which we prove in supplementary material E.2.

Theorem 6.3 (Upper bound for maximin fairness)**.** *Assume the existence of Oblivious Transfer. Without loss of generality, assume that $n_1 \geq n_0 \geq 1$ and $n_0 + n_1 > 2$. There exists a maximin-fair n-party coin toss protocol among n_0 players who prefer 0 and n_1 players who prefer 1, which tolerates up to t malicious adversaries where*

$$t := \begin{cases} \lceil \frac{1}{2}(n_0 + n_1) \rceil - 1, & \text{if } n_0 \geq 2, \\ \lceil \frac{1}{2} n_1 \rceil, & \text{if } n_0 = 1. \end{cases} \tag{2}$$

Acknowledgments

This work is in part supported by NSF under the award numbers CNS-1601879 and CNS-1561209, a Packard Fellowship, an ONR YIP award, a DARPA SIEVE grant, by the Israel Science Foundation (grant No. 2439/20), and by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 891234.

References

[ACH11] Gilad Asharov, Ran Canetti, and Carmit Hazay. Towards a game theoretic view of secure computation. In *Eurocrypt*, 2011.

[ADGH06] Ittai Abraham, Danny Dolev, Rica Gonen, and Joseph Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *PODC*, 2006.

[ADMM16] Marcin Andrychowicz, Stefan Dziembowski, Daniel Malinowski, and undefinedukasz Mazurek. Secure multiparty computations on bitcoin. *Commun. ACM*, 59(4):76–84, March 2016.

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In *Advances in Cryptology - EUROCRYPT 2012*, pages 483–501, 2012.

[AL11] Gilad Asharov and Yehuda Lindell. Utility dependence in correct and fair rational secret sharing. *Journal of Cryptology*, 24(1), 2011.

[BBBF18] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. In *CRYPTO*, 1 2018.

[BBF18] Dan Boneh, Benedikt Bünz, and Ben Fisch. A survey of two verifiable delay functions. Cryptology ePrint Archive, Report 2018/712, 2018.

[BGKO11] Amos Beimel, Adam Groce, Jonathan Katz, and Ilan Orlov. Fair computation with rational players. `https://eprint.iacr.org/2011/396.pdf`, full version of Eurocrypt'12 version, 2011.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10, 1988.

[BK14] Iddo Bentov and Ranjit Kumaresan. How to use bitcoin to design fair protocols. In *CRYPTO*, pages 421–439, 2014.

[Blu81] Manuel Blum. Coin flipping by telephone. In *CRYPTO*, 1981.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19. ACM, 1988.

[CCWS21] Kai-Min Chung, T-H. Hubert Chan, Ting Wen, and Elaine Shi. Game-theoretic fairness meets multi-party protocols: The case of leader election. In *CRYPTO*, 2021. `https://eprint.iacr.org/2020/1591`.

[CGL+18] Kai-Min Chung, Yue Guo, Wei-Kai Lin, Rafael Pass, and Elaine Shi. Game theoretic notions of fairness in multi-party coin toss. In *TCC*, 2018.

[Cle86] Richard Cleve. Limits on the security of coin flips when half the processors are faulty. In *STOC*, 1986.

[DHR00] Yevgeniy Dodis, Shai Halevi, and Tal Rabin. A cryptographic solution to a game theoretic problem. In *CRYPTO*, 2000.

[DR07] Yevgeniy Dodis and Tal Rabin. Cryptography and game theory. In *AGT*, 2007.

[GK12] Adam Groce and Jonathan Katz. Fair computation with rational players. In *Eurocrypt*, 2012.

[GKM+13] Juan A. Garay, Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Rational protocol design: Cryptography against incentive-driven adversaries. In *FOCS*, 2013.

[GKTZ15] Juan Garay, Jonathan Katz, Björn Tackmann, and Vassilis Zikas. How fair is your protocol? a utility-based approach to protocol optimality. In *PODC*, 2015.

[GLR13] Ronen Gradwohl, Noam Livne, and Alon Rosen. Sequential rationality in cryptographic protocols. *ACM Transactions on Economics and Computation (TEAC)*, 1(1):1–38, 2013.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *ACM symposium on Theory of computing (STOC)*, 1987.

[Gol04] Oded Goldreich. *The Foundations of Cryptography - Volume 2: Basic Applications*. Cambridge University Press, 2004.

[GTZ15] Juan A. Garay, Björn Tackmann, and Vassilis Zikas. Fair distributed computation of reactive functions. In *DISC*, volume 9363, pages 497–512, 2015.

[HT04] Joseph Halpern and Vanessa Teague. Rational secret sharing and multiparty computation. In *STOC*, 2004.

[IML05] Sergei Izmalkov, Silvio Micali, and Matt Lepinski. Rational secure computation and ideal mechanism design. In *FOCS*, 2005.

[IOZ14] Yuval Ishai, Rafail Ostrovsky, and Vassilis Zikas. Secure multi-party computation with identifiable abort. In *CRYPTO*, 2014.

[J.A74] Robert J.Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1(1), 1974.

[Kat08] Jonathan Katz. Bridging game theory and cryptography: Recent results and future directions. In *TCC*, 2008.

[KB14] Ranjit Kumaresan and Iddo Bentov. How to use bitcoin to incentivize correct computations. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 30–41. ACM, 2014.

[KMS+16] Ahmed E. Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pages 839–858. IEEE Computer Society, 2016.

[KN08] Gillat Kol and Moni Naor. Cryptography and game theory: Designing protocols for exchanging information. In *TCC*, 2008.

[KVV16] Ranjit Kumaresan, Vinod Vaikuntanathan, and Prashant Nalini Vasudevan. Improvements to secure computation with penalties. In *ACM CCS*, 2016.

[Nas51] John Nash. Non-cooperative games. *Annals of Mathematics*, 54(2), 1951.

[OPRV09] Shien Jin Ong, David C. Parkes, Alon Rosen, and Salil P. Vadhan. Fairness with an honest minority and a rational majority. In *TCC*, 2009.

[PS11] Rafael Pass and Abhi Shelat. Renegotiation-safe protocols. In *ICS*, pages 61–78. Citeseer, 2011.

[PS17] Rafael Pass and Elaine Shi. Fruitchains: A fair blockchain. In *PODC*, 2017.

[RB89] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 73–85. ACM, 1989.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations. In *FOCS*, 1982.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *FOCS*, 1986.

Supplementary Material

A Visualization of the Resilience Parameter

In Figure 1, we visualize the choice of t as a function of n_0 and n_1, to help understand the intriguing mathematical structure of game-theoretic fairness in multi-party coin toss.

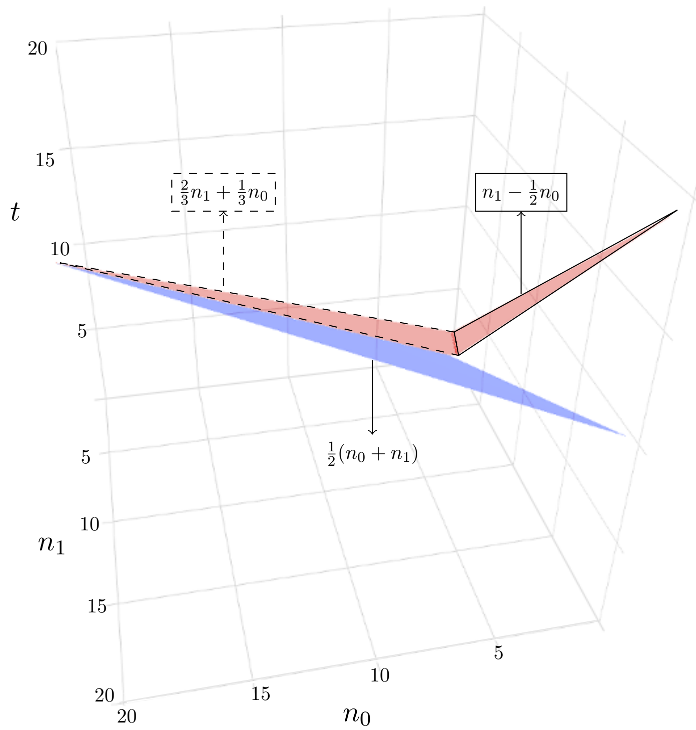


Figure 1: Visualization of the maximum t as a function of n_0 and n_1 in comparison to $\frac{1}{2}(n_0 + n_1)$. For simplicity we ignore the rounding in the plot. The blue plane is $\frac{1}{2}(n_0 + n_1)$, while the red plane with the dashes boundary is $\frac{2}{3}n_1 + \frac{1}{3}n_0$ when $n_1 < \frac{5}{2}n_0$, and the red plane with the solid boundary is $n_1 - \frac{1}{2}n_0$ when $n_1 \geq \frac{5}{2}n_0$.

B Preliminaries: Multi-Party Computation with Identifiable Abort

We define security of protocols that achieve security with identifiable abort. Since we use only functionalities where the parties do not have any inputs, we consider only functionalities with no inputs.

We consider a real world protocol π that securely emulated a functionality $(y_1, \ldots, y_n) = \mathcal{F}(1^\lambda)$. Security is defined via the simulation paradigm, by comparing between the "real world" execution and the "ideal world" as defined next.

The real world execution. Consider a real world protocol for parties $P_1, \ldots, P_n$ in the real model, which consists of specifications of next message functions. Each party P_i is executed with some randomness r_i, and is equipped with n authenticated private channels, where sending a message on the jth channel delivers a message to P_j, and a broadcast channel, in which each message broadcasted is delivered to all parties. The protocol specifies algorithms for computing the next message to deliver as a function of the messages received so far, the private input and the randomness of the party. At the end of the interaction, the party outputs some output y_i.

Let $\mathcal{A}$ be an adversary that initially corrupts some of the parties in $\{P_1, \dots, P_n\}$. We denote the set of corrupted parties by $I \subseteq [n]$. When a party is corrupted, the adversary $\mathcal{A}$ gets its input, and the messages it sends are controlled by the adversary. We let $\text{REAL}_{\pi,\mathcal{A}(z)}(\lambda)$ be a random variable consisting of the view of the adversary and the output of the honest parties, following an execution of π where P_i begins holding the security parameter λ.

The ideal world execution. The parties are $P_1, \dots, P_n$ and the adversary $\mathcal{S}$ controls a subset $I \subset [n]$. The idea execution of the functionality $\mathcal{F}$ proceeds as follows:

- **Inputs:** Each party P_i holds its input the security parameter λ. The adversary $\mathcal{S}$ also receives an auxiliary input z.
- **Trusted party sends outputs:** The trusted party chooses a random r uniformly at random and computes $(y_1, \dots, y_n) = \mathcal{F}(1^\lambda; r)$.

  Let $y_I = \{y_i\}_{i \in I}$. The trusted party sends y_I to the adversary $\mathcal{S}$.
- **The adversary decided whether to abort:** Upon receiving y_I the adversary can reply the trusted party with ok, or it must reply with abort_i for some $i \in I$.
- **Trusted party send outputs to honest parties:** If the adversary sends ok then the trusted party sends to each honest party P_j with $j \notin I$ its output y_j. Otherwise, if it sends abort_i to each honest party P_j.
- **Outputs:** The honest parties output whatever they were sent by the trusted party, the corrupted parties output nothing, and $\mathcal{S}$ outputs an arbitrary function of its view.

We let $\text{IDEAL}_{\mathcal{F},\mathcal{S}(z)}(\lambda)$ be the random variable consisting of the output of the adversary and the output of the honest parties following an execution in the ideal model described above.

Definition B.1. *Let $\mathcal{F}$ be a functionality with no inputs, and let π be a protocol for computing $\mathcal{F}$. The protocol π is said to* securely compute $\mathcal{F}$ with identifiable abort *if for every probabilistic polynomial-time adversary $\mathcal{A}$ in the real model, there exists a probabilistic polynomial-time adversary $\mathcal{S}$ in the ideal model such that*

$$\left\{\text{IDEAL}_{\mathcal{F},\mathcal{S}(z)}(\lambda)\right\}_{z\in\{0,1\}^*,\lambda\in\mathbb{N}} \approx_c \left\{\text{REAL}_{\pi,\mathcal{A}(z)}(\lambda)\right\}_{z\in\{0,1\}^*,\lambda\in\mathbb{N}}$$

The following theorem is based on [GMW87, Gol04, IOZ14]:

Theorem B.2. *Assuming oblivious transfer, for any n-party functionality with no inputs $\mathcal{F}$ there exists a protocol π that securely computes $\mathcal{F}$ with identifiable abort.*

We remark that the theorem holds also for functionalities that do have inputs, but we focus on use security with identifiable abort only for functionalities with no inputs.

C Deferred Proofs for the Upper Bound (Section 4)

C.1 Properties of the $\mathsf{HalfToss}^b$ Protocol (Protocol 4.1)

Lemma C.1 (Properties of the maliciously secure $\mathsf{HalfToss}$ sub-protocol)**.** *Suppose that the non-interactive commitment scheme employed by $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ is perfectly binding and computationally hiding, and that the signature scheme Sig satisfies existential unforgeability under chosen-message attack. Then, our maliciously secure, $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$-hybrid $\mathsf{HalfToss}^b[k]$ sub-protocol (Protocol 4.1) satisfies the following properties:*

- *Binding. If the vote phase does not fail, then the messages on the broadcast channel in the sharing and vote phases uniquely define a coin $s \neq \perp$ such that reconstruction must either output s or $\perp$.*
- *Knowledge threshold. For every non-uniform p.p.t. coalition controlling at most k number of b-supporters, there exists a p.p.t. simulator Sim such that either the vote phase fails, or*

$$(s, \mathsf{view}_A) \approx_c (\mathsf{Uniform}, \mathsf{Sim}(1^\lambda))$$

  where s denotes the unique coin value that the sharing and vote phases bind to, view_A denotes the coalition's view at the end of the vote phase, $\mathsf{Uniform}$ denotes a random bit sampled from $\{0,1\}$, and $\approx_c$ denotes computational indistinguishability.
- *Liveness threshold. If the coalition controls fewer than $\min(n_b - k, n_b/2)$ number of b-supporters, then the reconstruction phase must succeed and output a valid bit.*

Proof. We prove each of the properties one by one.

Binding. In our protocol, the vote phase either outputs a valid com or fails. If the vote phase fails, then reconstruction outputs $\perp$. If the vote phase outputs a valid com, the reconstruction outputs either a valid opening of com or it outputs $\perp$. Therefore, the binding property follows from the perfect binding property of the commitment scheme.

Knowledge threshold. If at most k number of b-supporters are corrupt, then, the vote phase must either fail, or output the vk that everyone receives from the $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ ideal functionality at the end of the sharing phase. Due to the security of the signature scheme, except with negliglible probability, if the vote phase does not fail, it must output the com value chosen by $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ ideal functionality at the end of the sharing phase. This com uniquely determines any non-$\perp$ value that can be reconstructed later.

To show the simulation statement, consider a hybrid experiment which is almost the same as running the sharing and vote phases of the $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$-hybrid $\mathsf{HalfToss}^b[k]$ sub-protocol, except with the following modifications:

- Every $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k_b]$ instance replaces $\mathsf{Commit}(s, r)$ with $\mathsf{Commit}(0, r')$ for some freshly sampled r' instead.
- Further, when it needs to compute $([s]_i, [r]_i)$ for some corrupt b-supporter i, it simply replaces shares with random field elements from the field of the Shamir secret sharing scheme — the replaced shares received by the adversary are identically distributed as honestly computed shares.

Due to the computational hiding property of the commitment scheme, it follows that the $(s_{\text{final}}, \mathsf{view}_A)$ pair in the hybrid experiment is computationally indistinguishable from the same pair in the sharing and vote phases of the $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$-hybrid, where we use s_{final} to denote the coin chosen by the successful instance of $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ that concludes the sharing phase. In the hybrid experiment, observe that view_A does not depend on the s values chosen by the $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ functionality, so we can equivalently imagine that a simulator Sim is sampling view_A. In the hybrid experiment, although the coalition can make $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ abort and retry a few times, it must make this decision without any information about s. Therefore, s_{final} is uniformly distributed.

Liveness threshold. Suppose that the adversary controls fewer than $(n_b - k, n_b/2)$ number of b-supporters. First, there are at least $k+1$ number of honest b-supporters who will vote for the vk output by the concluding $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k]$ instance at the end of the sharing phase. This means that vk' cannot be $\perp$. Should this vk be chosen as the vk' value, these $k+1$ number of honest b-supporters will also open their respective $[\mathsf{com}]_j$ shares during the vote phase, and thus vote phase will succeed. Moreover, during the reconstruction phase, these $k+1$ number of honest b-supporters will correctly open their respective $([s]_j, [r]_j)$ shares attached with a valid signature under $\mathsf{vk}' = \mathsf{vk}$. Thus, final reconstruction will be successful.

Therefore, the only way for the adversary to prevent reconstruction is to cause vk' to be a non-$\perp$ value different from vk. However, if the adversary has fewer than $n_b/2$ number of b-supporters, it cannot succeed in doing so.

□

C.2 Constraints (C1*), (C2*), and (C3*) Imply CSP Fairness

Lemma C.2 (CSP fairness of our final protocol)**.** *Suppose that the parameters k_0, k_1, and t are chosen such that conditions (C1*), (C2*), and (C3*) are satisfied. Then, our final protocol in Section 4.1.2 satisfies CSP fairness against any non-uniform p.p.t. coalition of size at most t.*

Proof. Due to condition (C3*), any coalition that causes the reconstruction of s_1 to output $\perp$ cannot benefit itself. Therefore, it suffices to consider coalition strategies that always let the reconstruction of s_1 to output a valid bit.

It suffices to show that for any non-uniform p.p.t. coalition that lets s_1 successfully reconstruct to a valid bit, the final outcome must be computationally indistinguishable from uniform at random. We now consider the following cases where t_b denotes the number of corrupted b-supporters.

Case 1: $t_1 \leq k_1$. We argue that the final outcome $s_0 + s_1$ output at the end is computationally indistinguishable from random. We consider the following sequence of hybrids. For convenience, for the $\mathsf{HalfToss}^b[k_b]$ sub-protocol, henceforth we call its 6 sequential steps Share^0, Share^1, Vote^1, Vote^0, Recons^0, Recons^1, respectively.

- Real: Execute the $\mathcal{F}^{b,\mathcal{O}}_{\text{sharegen}}[k_b]$-hybrid $\mathsf{HalfToss}^b[k_b]$ sub-protocol for the steps Share^0, Share^1, Vote^1, Vote^0, and Recons^0. Note that at this moment, both s_0 and s_1 are well-defined bits. Output the $s_0 + s_1$ value.

- Hyb: Below, we use $\mathcal{A}$ to denote the non-uniform p.p.t. adversary controlling a coalition $A \subset [n]$. Consider an experiment in which a reduction $\mathcal{R}$ interacts with $\mathcal{A}$ as follows:
  - Share^0: the reduction $\mathcal{R}$ acts on behalf of the honest parties and the $\mathcal{F}^{0,\mathcal{O}}_{\text{sharegen}}[k_0]$ functionality in the $\mathsf{HalfToss}^0[k_0]$ instance and interacts with the adversary $\mathcal{A}$. Let st be the adversary's state at this point.

– Share^1, Vote^1: Sample a coin $s_1 \xleftarrow{\$} \{0,1\}$ uniformly at random. Run the simulator $\mathsf{Sim}^{\mathcal{A}(\mathsf{st})}$ of Lemma C.1. Reset $\mathcal{A}$'s state to the outcome of the simulator.

– Recons^0: $\mathcal{R}$ continues to act on behalf of the honest parties in the $\mathsf{HalfToss}^0[k_0]$ instance and interact with the adversary $\mathcal{A}$. This steps defines an $s_0 \in \{0,1\}$.

– Output $s_0 + s_1$.

Due to the knowledge threshold property of Lemma C.1, if $t_1 \leq k_1$, Real is computationally indistinguishable from Hyb. Observe that in Hyb, the outcome is uniform at random.

Remark C.3. *Interestingly, note that had we reversed the order of* Vote^1 *and* Vote^0 *or reversed the order of* Recons^0 *and* Recons^1 *in the final protocol, the above claim and proof would not hold.*

Case 2: $t_1 \geq k_1 + 1$**.** Due to condition (C2*), the s_0 reconstruction must output a valid bit. Therefore, the final coin value s_0+s_1 is determined at the end of the voting phase. Due to condition (C1*), it must be that $t_0 \leq k_0$. Now, the $\mathsf{HalfToss}^0[k_0]$ instance must satisfy the knowledge threshold property of Lemma C.1. Therefore, we can use a proof almost the same as the proof of Case 1 (but executing only the steps Share^0, Share^1, Vote^1, and Vote^0 which fully determines $s_0 + s_1$), to show that the final coin $s_0 + s_1$ is computationally indistinguishable from random. □

C.3 Maximizing t Subject to the Constraint System

Lemma C.4 (Solving the constraint system and maximizing t)**.** *Assuming $n_1 \geq n_0 \geq 1$. For the constraint system specified by (C1), (C2), and (C3), t is maximized when k_0 and k_1 are chosen as follows:*

- *if $n_1 \geq \frac{5}{2}n_0$: in this case t is maximized when $k_0 = \lfloor \frac{n_0}{2} \rfloor$ and $k_1 = n_1 - n_0$, and the maximum t is $t = n_1 - \lfloor \frac{1}{2}n_0 \rfloor$.*
- *if $n_1 < \frac{5}{2}n_0$: in this case t is maximized when $k_0 = \lfloor \frac{n_0}{2} \rfloor$ and $k_1 = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor$, and the maximum t is $t = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{n_0}{2} \rceil$.*

Proof. For completeness, we write again the constraints that are implied by (C1), (C2), and (C3): (Parameter Constraints 2.3), while recall that we assume that $0 \leq k_0 \leq n_0$, $0 \leq k_1 \leq n_1$:

(C1): $t \leq k_0 + k_1 + 1$,
(C2): $t \leq k_1 + 1 + n_0 - k_0 - 1 = n_0 + k_1 - k_0$,
(C3): if $n_1 - k_1 < n_0$, then $t \leq 2(n_1 - k_1)$.

Note that k_0 only appears in the conditions (C1) and (C2). For any fixed k_1, the feasible region of t and k_0 is depicted in Figure 2.

Therefore, for any fixed k_1, we need to pick k_0 such that $k_0 + k_1 + 1 = n_0 + k_1 - k_0$ to maximize t. After rounding we have that $k_0 = \lfloor \frac{1}{2}n_0 \rfloor$.

Plugging $k_0 = \lfloor \frac{1}{2}n_0 \rfloor$ back, the problem now boils down to finding k_1 that maximizes t such that

- $t \leq k_1 + \lceil \frac{1}{2}n_0 \rceil$.
- if $k_1 > n_1 - n_0$ then $t \leq 2(n_1 - k_1)$.

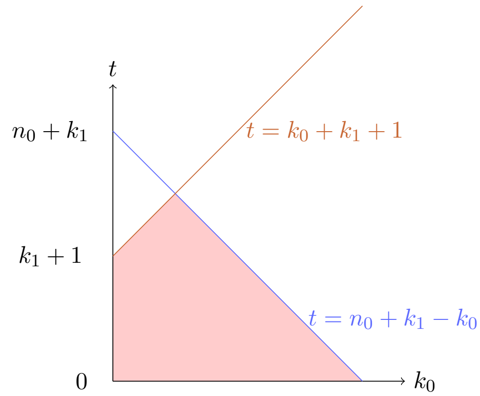
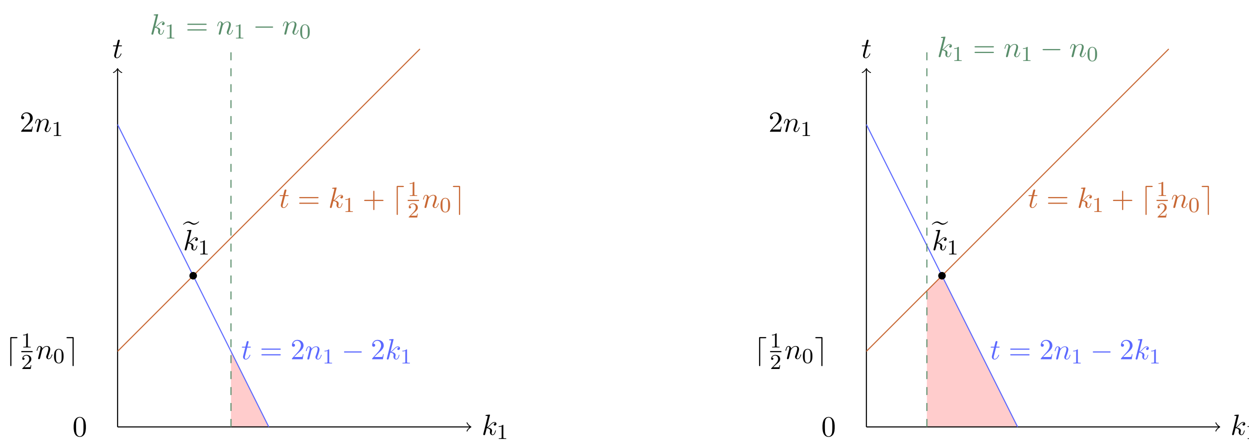


Figure 2: Feasible region (red) defined by $t \le k_0 + k_1 + 1$ and $t \le n_0 + k_1 - k_0$.



(a) The case: $k_1 > n_1 - n_0$.

(b) The case: $k_1 > n_1 - n_0$.

Figure 3: Feasible region (red) defined by $t \le k_1 + \lceil \frac{1}{2} n_0 \rceil$, $t \le 2n_1 - 2k_1$.

There are two cases to consider:

If $\mathbf{n_1 \ge \frac{5}{2} n_0}$: We have two sub-cases here:

1. If $k_1 \le n_1 - n_0$, then we only need to maximize $t \le k_1 + \lceil \frac{1}{2} n_0 \rceil$ given that $k_1 \le n_1 - n_0$. It is clear that picking $k_1 = n_1 - n_0$ maximizes t. In this case $t = k_1 + \lceil \frac{1}{2} n_0 \rceil = n_1 - \lfloor \frac{1}{2} n_0 \rfloor$.

2. If $k_1 > n_1 - n_0$, then the feasible region is depicted in Figure 3a. Note that the two lines $t = k_1 + \lceil \frac{1}{2} n_0 \rceil$ and $t = 2n_1 - 2k_1$ intersect at $\widetilde{k}_1 = \frac{2}{3} n_1 - \frac{1}{3} \lceil \frac{1}{2} n_0 \rceil$. Since $n_1 \ge \frac{5}{2} n_0$, $n_1 - n_0 \ge \widetilde{k}_1$, and t is maximized when picking $k_1 = n_1 - n_0 + 1$. In this case $t = 2n_0 - 2$.

When $n_1 \ge \frac{5}{2} n_0$, we have that $n_1 - \lfloor \frac{1}{2} n_0 \rfloor \ge 2n_0 - 2$. Therefore, t is maximized in the first sub-case among the two sub-cases we just considered, that is, we pick $k_0 = \lfloor \frac{n_0}{2} \rfloor$, $k_1 = n_1 - n_0$, and then the maximum t is $n_1 - \lfloor \frac{1}{2} n_0 \rfloor$.

If $\mathbf{n_1 < \frac{5}{2} n_0}$: We have two sub-cases here.

1. If $k_1 \le n_1 - n_0$, t is maximized when $k_1 = n_1 - n_0$. In this case $t = n_1 - \lfloor \frac{1}{2} n_0 \rfloor$.

2. If $k_1 > n_1 - n_0$, then the feasible region is depicted in Figure 3b. Since $n_1 < \frac{5}{2}n_0$, then $n_1 - n_0 < \widetilde{k}_1$, and t is maximized when picking $k_1 = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor$. In this case the maximum $t = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{1}{2}n_0 \rceil$.

When $n_1 < \frac{5}{2}n_0$, we have that $\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{1}{2}n_0 \rceil \geq n_1 - \lfloor \frac{1}{2}n_0 \rfloor$. In this case, t is maximized in the second sub-cases among the two we have just considered, that is, when $k_0 = \lfloor \frac{n_0}{2} \rfloor$, $k_1 = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor$, and the maximum t is $t = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{1}{2}n_0 \rceil$.

□

Corollary C.5. *Assuming $n_1 \geq n_0 \geq 1$. For the constraint system specified by (C1*), (C2*), and (C3*), t is maximized when k_0 and k_1 are chosen as in Lemma C.4*

- *if $n_1 \geq \frac{5}{2}n_0$: in this case t is maximized when $k_0 = \lfloor \frac{n_0}{2} \rfloor$ and $k_1 = n_1 - n_0$, and the maximum t is $t = n_1 - \lfloor \frac{1}{2}n_0 \rfloor$.*
- *if $n_1 < \frac{5}{2}n_0$: in this case t is maximized when $k_0 = \lfloor \frac{n_0}{2} \rfloor$ and $k_1 = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor$, and the maximum t is $t = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{n_0}{2} \rceil$.*

Proof. As we mentioned in Section 4.1.1, if the optimal solution for (C1), (C2), and (C3) satisfies that $k_0 = \lfloor \frac{1}{2}n_0 \rfloor = \lceil \frac{1}{2}(n_0 - 1) \rceil$ and $k_1 \geq \lfloor \frac{1}{2}n_1 \rfloor$, then this solution is also optimal for the constraint system specified by (C1*), (C2*), and (C3*). The optimal solution stated in Lemma C.4 does satisfy these properties. □

C.4 Tolerating One More Corruption when $n_0 = n_1 = odd$

When $n_0 = n_1 = odd$, our algorithm can actually tolerate one more corruption. That is, when $n_1 = n_0 = odd$, we set $k_0 = \lfloor \frac{1}{2}n_0 \rfloor$ and $k_1 = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor = \lfloor \frac{1}{2}n_1 \rfloor$. The final protocol described in Section 4.1.2 is CSP fair against a coalition of size up to $t = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{1}{2}n_0 \rceil + 1 = n_0 + 1$.

We use t_0 and t_1 to denote the number of corrupted 0-supporters and 1-supporters, respectively. Then we have the following cases:

If $t_0 = k_0 + 1 = \lceil \frac{1}{2}n_0 \rceil$: In this case, $t_1 = t - t_0 = \lceil \frac{1}{2}n_0 \rceil = t_0$. This indicates that the coalition has no preference and they can bias the output arbitrarily.

If $t_0 \leq k_0 = \lfloor \frac{1}{2}n_0 \rfloor$: Then the coalition cannot control coin s_0, nor can it hamper the reconstruction of s_0. Moreover, if the coalition can fail the reconstruction of s_1, then it must corrupt more 1-supporters than 0-supporters. This means that the conditions (C1*), (C2*) and (C3*) are all satisfied. According to Lemma C.2, the protocol achieves CSP fairness.

If $t_0 > k_0 + 1 = \lceil \frac{1}{2}n_0 \rceil$: In this case, $t_1 < \lceil \frac{1}{2}n_0 \rceil$. The coalition cannot control coin s_1, nor can it hamper the reconstruction of s_1. Therefore, the conditions (C1*), (C2*) and (C3*) are all satisfied. According to Lemma C.2, the protocol achieves CSP fairness.

To summarize, when $n_0 = n_1 = odd$, we can tolerate $t = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{1}{2}n_0 \rceil + 1 = n_0 + 1$ number of corruptions by picking $k_0 = \lfloor \frac{1}{2}n_0 \rfloor$ and $k_1 = \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor = \lfloor \frac{1}{2}n_1 \rfloor$.

D Deferred Proofs for the Lower Bound (Section 5)

D.1 Proof of Lemma 5.9

Lemma D.1 (Lemma 5.9, restated: Solving the constraint system and minimizing t)**.** *For Parameter Constraint 5.1, the parameter t is minimized when α_0 and α_1 are chosen as follows, and the corresponding t is:*

Case	α_0	α_1	t
$n_1 \geq \frac{5}{2}n_0$, $n_0 \geq 2$	$\lfloor \frac{1}{2}n_0 \rfloor$	$n_0 - 1$	$n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1$
$2 \leq n_0 < n_1 < \frac{5}{2}n_0$	$\lfloor \frac{1}{2}n_0 \rfloor$	$\lceil \frac{1}{3}n_1 + \frac{1}{6}n_0 \rceil - 1$	$\lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$
$2 \leq n_0 = n_1$	$\lfloor \frac{1}{2}n_0 \rfloor$	$\lfloor \frac{1}{2}n_0 \rfloor - 1$	$2\lceil \frac{1}{2}n_0 \rceil + 1$

Note that for the case $t = 2\lceil \frac{1}{2}n_0 \rceil + 1$, this expression is equal to $\lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + \lceil \frac{1}{2}n_0 \rceil + 1$ when $n_0 = n_1$ is even, and is equal to $n_0 + 2$ when when $n_0 = n_1$ is odd.

Proof. We first prove the case where $n_1 > n_0 \geq 2$. We start by reviewing all constraints as in Parameter Constraints 5.1:

	Non-negative	*Lone-wolf*	*Wolf-minion*	*T_2-equity*
(1)	$0 \leq \alpha_0 \leq \frac{1}{2}n_0$	$\alpha_1 + 1 \leq n_0$	$n_0 - \alpha_0 < n_1 - \alpha_1$	$1 \leq \alpha_0$
(2)	$0 \leq \alpha_1 \leq \frac{1}{2}n_1$	$\alpha_0 + 1 \leq n_1$	$n_0 + n_1 - \alpha_0 - \alpha_1 \leq t$	$1 \leq \alpha_1$
(3)		$\alpha_0 + \alpha_1 \leq t$		$3 \leq t$
(4)		$2\alpha_0 + 1 \leq t$		$1 \leq n_0 + n_1 - 2\alpha_0 - 2\alpha_1 \leq t$
(5)		$2\alpha_1 + 1 \leq t$		

We start by cleaning up the constraints since some of the constraints can be implied from other constraints. We obtain the following set of constraints, and we will next showed why they all imply the previous set of constraints:

Parameter Constraints D.2 (Simplified Constraint System for Lower Bound).

1. $1 \leq \alpha_0 \leq \frac{1}{2}n_0$;
2. $1 \leq \alpha_1 \leq \min(\frac{1}{2}n_1, n_0 - 1, \frac{1}{2}(t-1))$;
3. $n_1 - \alpha_1 > n_0 - \alpha_0$;
4. $t \geq n_0 + n_1 - \alpha_0 - \alpha_1$.

1. The first constraint in Parameter Constraints D.2 implies constraint (1) of non-negative condition and constraint (1) of T_2-equity condition. It also implies constraint (2) of lone-wolf condition: recall that we assume $2 \leq n_0 < n_1$. Thus, $\alpha_0 \leq \frac{1}{2}n_0 < n_0 \leq n_1 - 1$.
2. The second constraint in Parameter Constraints D.2 implies constraint (2) of non-negative condition, constraint (1) and (5) of lone-wolf condition, and constraint (2) of T_2-equity condition.
3. The third constraint in Parameter Constraints D.2 is exactly constraint (1) in wolf-minion condition.
4. The forth constraint in Parameter Constraints D.2 is exactly constraint (2) in wolf-minion condition.
5. The combination of the first, second, and forth constraints in Parameter Constraints D.2 implies the third constraint of lone-wolf. From the first constraint, we have that $\alpha_0 \leq \frac{1}{2}n_0$ and so $2\alpha_0 \leq n_0$. Similarly, from the second constraint we get that $2\alpha_1 \leq n_1$. Putting into the forth constraint:
$$t \geq n_0 + n_1 - \alpha_0 - \alpha_1 \geq \alpha_0 + \alpha_1 \ ,$$
which is exactly the third constraint of lone-wolf.

6. the forth constraint of lone-wolf is implied by the other constraints (which are implied by Parameter Constraints D.2). Specifically, $t \geq n_0 + n_1 - \alpha_0 - \alpha_1$. We know that $\alpha_0 + \alpha_1 \leq t$ (the third constraint of lone-wolf), and thus $2t \geq n_0 + n_1$. Since $n_1 \geq \alpha_0 + 1$ (second constraint of lone-wolf), we obtain that $2t \geq n_0 + \alpha_0 + 1$. Moreover, we know that $\alpha_0 \leq n_0$, and thus $2t \geq 2\alpha_0 + 1$.

Moreover, since $\alpha_0 \leq \frac{1}{2}n_0, \alpha_1 \leq \frac{1}{2}n_1$, we only need to consider the constraint $n_1 + n_0 - 2\alpha_0 - 2\alpha_1 \geq 1$ (the forth constraint of T_2-equity) when both $\alpha_0 = \frac{1}{2}n_0$ and $\alpha_1 = \frac{1}{2}n_1$. Also, we only need to consider $t \geq 3$ after we find the minimum t and check whether the minimized value satisfies that $t \geq 3$ (the third condition of T_2-equity.

The feasible region of α_0 and α_1 in Parameter Constraints D.2 is depicted in Figure 4. Note that red lines are moving lines—namely, there intersection with α_1 and α_0 axes are changed with respect to different values of t. In any case, if there is a feasible solution, then the minimum t is obtained at the black dot which is the intersection of the green vertical line and the red horizontal lines. In that case, $\alpha_0 = \lfloor\frac{1}{2}n_0\rfloor$ and $\alpha_1 = \min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1))$.

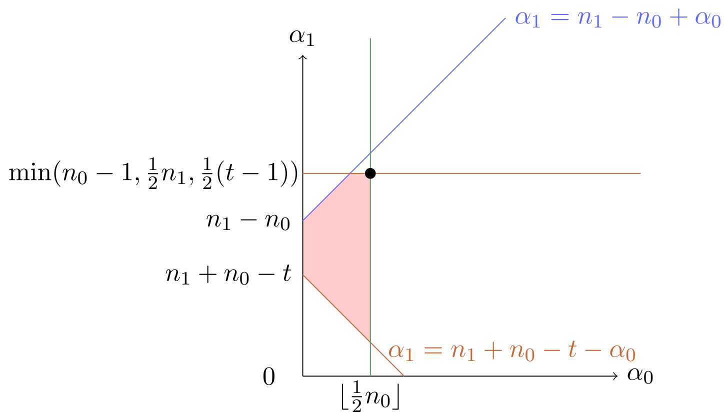


Figure 4: The feasible region defined by simplified constraint system. In this diagram, each of the constraints in Parameter Constrains D.2 is depicted with a corresponding line. For instance, Constraint 4 is depicted with the red decreasing line, and the feasible region must be above it.

We now find the minimal t given α_0, α_1. We have the following three cases:

If $\mathbf{n_0 < n_1 < \frac{5}{2}n_0}$: In this case,

- If $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = n_0 - 1$: Putting $\alpha_0 = \lfloor\frac{1}{2}n_0\rfloor$ and $\alpha_1 = n_0 - 1$ into Constraint 4 in the simplified parameter constraints, $t \geq n_1 + n_0 - \alpha_0 - \alpha_1$, we obtain that $t \geq n_1 + n_0 - \lfloor\frac{1}{2}n_0\rfloor - n_0 + 1 = n_1 - \lfloor\frac{1}{2}n_0\rfloor + 1$.

  Moreover, since $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = n_0 - 1$, we can conclude that $n_0 - 1 \leq \frac{1}{2}(t-1)$ and so $t \geq 2n_0 - 1$. Putting together we have that the minimum $t = \max(2n_0 - 1, n_1 - \lfloor\frac{1}{2}n_0\rfloor + 1)$.

- If $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = \frac{1}{2}n_1$, then we actually have $\alpha_1 = \lfloor\frac{1}{2}n_1\rfloor$ (we add the floor to guarantee that α_1 is an integer). Recall that we omitted the constraint $n_0 + n_1 - 2\alpha_0 - 2\alpha_1 \geq 1$ and mentioned that it should be considered only when $\alpha_0 = \frac{1}{2}n_0$ and $\alpha_1 = \frac{1}{2}n_1$. When $\alpha_0 = \lfloor\frac{1}{2}n_0\rfloor$ and $\alpha_1 = \lfloor\frac{1}{2}n_1\rfloor$, it holds that $n_0 + n_1 - 2\alpha_0 - 2\alpha_1 < 1$ only when both n_0 and n_1 are even. In this case, we take a step back and pick $\alpha_1 = \lfloor\frac{1}{2}n_1\rfloor - 1$.

  In any case, $t \geq n_1 + n_0 - \lfloor\frac{1}{2}n_1\rfloor - \lfloor\frac{1}{2}n_0\rfloor + 1$, and $t \geq n_1 + 1$ since $\frac{1}{2}(t-1) \geq \frac{1}{2}n_1$. Therefore, the minimum possible is obtained when $t = n_1 + 1$.

- If $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = \frac{1}{2}(t-1)$: Then we have $t \geq n_1 + n_0 - \frac{1}{2}n_0 - \frac{1}{2}(t-1)$. This means that $t \geq \frac{2}{3}n_1 + \frac{1}{3}n_0 + \frac{1}{3}$ and $\frac{1}{2}(t-1) \geq \frac{1}{3}n_1 + \frac{1}{6}n_0 + \frac{1}{6}$. After rounding we have $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor, \alpha_1 = \lceil \frac{1}{3}n_1 + \frac{1}{6}n_0 \rceil - 1$, and the minimum $t = n_1 + n_0 - \lfloor \frac{1}{2}n_0 \rfloor - \lceil \frac{1}{3}n_1 + \frac{1}{6}n_0 \rceil + 1 = \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$.

We obtained three different possible values of t. The minimal k is then obtained as (recall, $n_1 < \frac{5}{2}n_0$):

$$\min(\lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1, \max(2n_0 - 1, n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1), n_1 + 1) = \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1.$$

Let $\Delta = \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$. To see that:

- $\Delta \leq n_1 + 1$, note that

$$\Delta = \begin{cases} \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 + \frac{1}{2}n_0 \rfloor + 1 \leq n_1 + 1, & \text{if } n_0 \text{ is even} \\ \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 + \frac{1}{2}n_0 + \frac{1}{2} \rfloor + 1 \leq \lceil \frac{2}{3}n_1 + \frac{1}{3}n_0 \rceil + 1 \leq n_1 + 1, & \text{if } n_0 \text{ is odd} \end{cases}$$

- $\Delta \leq \max(2n_0 - 1, n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1)$, we first note that $\max(2n_0 - 1, n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1) = n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1$ only when $\lfloor \frac{5}{2}n_0 \rfloor - 2 < n_1$. Hence, for $n_1 \leq \lfloor \frac{5}{2}n_0 \rfloor - 2$:

$$\Delta = \begin{cases} \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 + \frac{1}{2}n_0 \rfloor + 1 \leq \lfloor \frac{5}{3}n_0 - \frac{4}{3} + \frac{1}{3}n_0 \rfloor + 1 = 2n_0 - 1, & \text{if } n_0 \text{ is even} \\ \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 + \frac{1}{2}n_0 + \frac{1}{2} \rfloor + 1 \leq \lfloor \frac{2}{3}(\frac{5}{2}n_0 - \frac{5}{2}) + \frac{1}{3}n_0 + \frac{1}{2} \rfloor + 1 = 2n_0 - 1, & \text{if } n_0 \text{ is odd} \end{cases}$$

  On the other hand, for $\lfloor \frac{5}{2}n_0 \rfloor - 2 < n_1 < \frac{5}{2}n_0$, i.e., for $n_1 = \lfloor \frac{5}{2}n_0 \rfloor$ (if n_0 is odd) or $\lfloor \frac{5}{2}n_0 \rfloor - 1$, $\max(2n_0 - 1, n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1) = n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1$. When $n_1 = \lfloor \frac{5}{2}n_0 \rfloor$,

$$\lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1 \leq \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{5}{3}n_0 - \frac{1}{6}n_0 \rfloor + 1 = 2n_0 + 1 \leq n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1.$$

  Similarly, when $n_1 = \lfloor \frac{5}{2}n_0 \rfloor - 1$,

$$\lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1 \leq \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{5}{3}n_0 - \frac{2}{3} - \frac{1}{6}n_0 \rfloor + 1 = 2n_0 \leq n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1.$$

  Therefore, when $n_1 < \frac{5}{2}n_0$, t is minimized when picking $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor$ and $\alpha_1 = \lceil \frac{1}{3}n_0 + \frac{1}{6}n_0 \rceil - 1$. Under this parameter choice t is minimized as $t = \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$, and this minimum value satisfies that $t \geq 3$ when $n_1 > n_0 \geq 2$.

To conclude, the case of $2 \leq n_0 < n_1 < \frac{5}{2}n_0$ boils down to picking $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor$, $\alpha_1 = \lceil \frac{1}{3}n_1 + \frac{1}{6}n_0 \rceil - 1$, and then t is minimized for $t = \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$.

If $\mathbf{n_1 \geq \frac{5}{2}n_0}$: In this case,

- If $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = n_0 - 1$: Putting $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor$ and $\alpha_1 = n_0 - 1$ into Constraint 4 in the simplified parameter constraints, $t \geq n_1 + n_0 - \alpha_0 - \alpha_1$, we obtain that $t \geq n_1 + n_0 - \lfloor \frac{1}{2}n_0 \rfloor - n_0 + 1 = n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1$.

  Moreover, since $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = n_0 - 1$, we can conclude that $n_0 - 1 \leq \frac{1}{2}(t-1)$ and so $t \geq 2n_0 - 1$. Putting together we have that the minimum $t = \max(2n_0 - 1, n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1) = n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1$ since $n_1 \geq \frac{5}{2}n_0$.

- If $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = \frac{1}{2}n_1$: This will not happen since $\frac{1}{2}n_1 \geq \frac{5}{4}n_0 > n_0 - 1$.
- If $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1)) = \frac{1}{2}(t-1)$: Then we have $t \geq n_1 + n_0 - \frac{1}{2}n_0 - \frac{1}{2}(t-1)$. This means that $t \geq \frac{2}{3}n_1 + \frac{1}{3}n_0 + \frac{1}{3}$ and $\frac{1}{2}(t-1) \geq \frac{1}{3}n_1 + \frac{1}{6}n_0 + \frac{1}{6}$. After rounding we have $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor, \alpha_1 = \lceil \frac{1}{3}n_1 + \frac{1}{6}n_0 \rceil - 1$, and the minimum $t = n_1 + n_0 - \lfloor \frac{1}{2}n_0 \rfloor - \lceil \frac{1}{3}n_1 + \frac{1}{6}n_0 \rceil + 1 = \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$.

  However, $\lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1 \geq 2n_0 - 1$. That is, the minimum possible $t \geq 2n_0 - 1$, indicating that $\frac{1}{2}(t-1) \geq n_0 - 1$. Therefore, $\frac{1}{2}(t-1)$ cannot be the minimum value $\min(n_0 - 1, \frac{1}{2}n_1, \frac{1}{2}(t-1))$. In this case there is no feasible solution.

Therefore, when $n_1 \geq \frac{5}{2}n_0$, t is minimized when picking $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor$ and $\alpha_1 = n_0 - 1$. Under this parameter choice t is minimized as $t = n_1 - \lfloor \frac{1}{2}n_0 \rfloor + 1$, and this minimum value satisfies that $t \geq 3$ when $n_1 > n_0 \geq 2$.

The case where $n_1 = n_0$. In this case, we start with Parameter Constraints D.2, and apply $n_0 = n_1$:

1. $1 \leq \alpha_0 \leq \frac{1}{2}n_0$;
2. $1 \leq \alpha_1 \leq \min(\frac{1}{2}n_1, n_0 - 1, \frac{1}{2}(t-1)) = \min(\frac{1}{2}n_0, n_0 - 1, \frac{1}{2}(t-1)) = \min(\frac{1}{2}n_0, \frac{1}{2}(t-1))$ for $n_0 \geq 2$;
3. $n_1 - \alpha_1 > n_0 - \alpha_0$, which implies that $\alpha_0 > \alpha_1$ when $n_0 = n_1$;
4. $t \geq n_0 + n_1 - \alpha_0 - \alpha_1 = 2n_0 - \alpha_0 - \alpha_1$.

Recall that in Parameter Constraints D.2, we used the assumption that $n_0 < n_1$ only to show that the second constraint of lone-wolf is implied by the system, and therefore, apparently, this constraint is not implied when $n_0 = n_1$. However, when $n_0 = n_1$, this constraints boils down to requiring that $\alpha_0 + 1 \leq n_0$, which is implied by our simplified constraints since:

$$\alpha_0 < \alpha_1 \leq n_1 = n_0$$

and so $\alpha_0 + 1 \leq n_0$.

To conclude, we obtain the following simplified constraints:

Parameter Constraints D.3 (Simplified Constraint System for Lower Bound when $n_1 = n_0$).

- $1 \leq \alpha_1 < \alpha_0 \leq \min(\frac{1}{2}n_0, \frac{1}{2}(t-1))$;
- $t \geq 2n_0 - \alpha_0 - \alpha_1$.

Similarly, t is minimized when $\alpha_0 = \min(\frac{1}{2}n_0, \frac{1}{2}(t-1))$, and $\alpha_1 = \alpha_0 - 1$. We have the following cases:

- If $\frac{1}{2}n_0 \leq \frac{1}{2}(t-1)$, then we pick $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor$, $\alpha_1 = \lfloor \frac{1}{2}n_0 \rfloor - 1$. Then $t \geq 2n_0 - \alpha_0 - \alpha_1 = 2\lceil \frac{1}{2}n_0 \rceil + 1$. Moreover, since $\frac{1}{2}n_0 \leq \frac{1}{2}(t-1)$, $t \geq n_0 + 1$. Therefore, the minimum $t = \max(2\lceil \frac{1}{2}n_0 \rceil + 1, n_0 + 1) = 2\lceil \frac{1}{2}n_0 \rceil + 1$.
- If $\frac{1}{2}n_0 > \frac{1}{2}(t-1)$. Letting $\alpha_0 = \frac{1}{2}t - 1$ and $\alpha_1 = \alpha_0 - 1$, In this case $t \geq 2n_0 - \frac{1}{2}(t-1) - \frac{1}{2}(t-1) + 1$, indicating that $t \geq n_0 + 1$. However, this indicates that $\frac{1}{2}n_0 \leq \frac{1}{2}(t-1)$, which contradicts with our assumption. Therefore, there is no feasible solution in this case.

To conclude, when $n_0 = n_1$, t is minimized when $\alpha_0 = \lfloor \frac{1}{2}n_0 \rfloor$, $\alpha_1 = \lfloor \frac{1}{2}n_0 \rfloor - 1$, and the minimum $t = 2\lceil \frac{1}{2}n_0 \rceil + 1$. Note that for even $n_0 = n_1$, $t = 2\lceil \frac{1}{2}n_0 \rceil + 1 = \lceil \frac{1}{2}n_0 \rceil + \lfloor \frac{2}{3}n_1 - \frac{1}{6}n_0 \rfloor + 1$; for odd $n_0 = n_1$, $t = 2\lceil \frac{1}{2}n_0 \rceil + 1 = n_0 + 2$. □

D.2 Proof of Theorem 2.5

For completeness, we give an explicit proof of Theorem 2.5, which is implicit in the work by Chung et. al. [CGL+18].

Theorem D.4 (Theorem 2.5, restated (Generalized Theorem 21 of Chung et. al. [CGL+18]))**.** *There is no protocol Π among three super nodes $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ such that Π satisfies the above wolf-minion condition, the lone-wolf condition and the T_2-equity condition simultaneously.*

Proof. For the sake of contradiction, let Π be an $R = R(\lambda, n_0, n_1)$-round protocol among three super nodes $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$. Moreover, the protocol Π satisfies the lone-wolf condition, the wolf-minion condition and the T_2-equity condition.

Without loss of generality, we assume that the message schedule of the protocol proceeds in R rounds and satisfies the following assumptions:

- In the first round, only $\mathcal{S}_1$ sends messages;
- In round $2, \ldots, R-1$, $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ all send messages;
- In round R, only $\mathcal{S}_3$ sends messages.

It is easy to see that any protocol among three super nodes can be transformed into a three-party protocol that satisfy the above message schedule conditions with only $O(1)$ additional rounds: one can always send a filler message if a party does not want to send a message in that round.

Now we define a sequence of adversaries as in [Cle86], but conditioned on any fixed choice of $\mathcal{S}_2$'s randomness T_2:

- $\mathcal{A}_i^b(1^\lambda, T_2)$ corrupts $\mathcal{S}_1$ and $\mathcal{S}_2$ and wants to bias the output towards $b \in \{0, 1\}$: $\mathcal{A}_i^b$ uses T_2 as the randomness for party $\mathcal{S}_2$. It chooses the randomness for party $\mathcal{S}_1$ honestly. Then it executes the protocol honestly till the moment right before $\mathcal{S}_1$ is going to broadcast its i-th message.

  Then it computes α_i, the output of $\mathcal{S}_1$ and $\mathcal{S}_2$ imagining that $\mathcal{S}_3$ aborts right after sending its $(i-1)$-th message, i.e., $\mathcal{S}_3$'s message in i-th round.

  If $\alpha_i = b$, then $\mathcal{S}_1$ aborts after sending the i-th message. Otherwise $\mathcal{S}_1$ aborts without sending the i-th message.

- $\mathcal{B}_i^b(1^\lambda, T_2)$ corrupts $\mathcal{S}_3$ and $\mathcal{S}_2$ and wants to bias the output of $\mathcal{S}_1$ towards $b \in \{0, 1\}$: $\mathcal{B}_i^b$ uses T_2 as the randomness for party $\mathcal{S}_2$. It chooses the randomness for party $\mathcal{S}_3$ honestly. Then it executes the protocol honestly till the moment right before $\mathcal{S}_3$ is going to broadcast its i-th message.

  Then it computes β_i, the output of $\mathcal{S}_3$ and $\mathcal{S}_2$ imagining that $\mathcal{S}_1$ aborts right after sending its $(i-1)$-th message.

  If $\beta_i = b$, then $\mathcal{S}_3$ aborts after sending the i-th message. Otherwise $\mathcal{S}_3$ aborts without sending the i-th message.

- $\mathcal{A}_0(1^\lambda, T_2)$ corrupts $\mathcal{S}_1$ and $\mathcal{S}_2$ and wants to bias the output of $\mathcal{S}_3$ towards either direction. It runs $\mathcal{S}_2$ with randomness T_2 and has $\mathcal{S}_1$ abort at the very beginning of the protocol.

By definition, in the above sequences of adversaries $\{\mathcal{A}_i^b(1^\lambda, T_2), \mathcal{B}_i^b(1^\lambda, T_2) \text{ for } i \in [R], b \in \{0, 1\}\}$ and $\mathcal{A}_0(1^\lambda, T_2)$, each adversary corrupts two parties, either $\mathcal{S}_1$ and $\mathcal{S}_2$, or $\mathcal{S}_3$ and $\mathcal{S}_2$. And they all run $\mathcal{S}_2$ as a silent corrupted party that never aborts.

Fixing any T_2, the three-party protocol Π with R rounds can be viewed as a residual two-party protocol between $\mathcal{S}_1$ and $\mathcal{S}_3$ with R rounds. In this residual protocol, T_2 is hardwired in $\mathcal{S}_1$ and $\mathcal{S}_3$'s programs so they can simulate the behavior of $\mathcal{S}_2$. We denote the two-party residual protocol as Π_{res}. The above sequence of adversaries in Π can thus be viewed as a sequence of adversaries in the residual two-party protocol Π_{res}. Now $\{\mathcal{A}_i^b(1^\lambda, T_2)\}_{i\in[R],b\in\{0,1\}}$ and $\mathcal{A}_0(1^\lambda, T_2)$ corrupts $\mathcal{S}_1$ and that $\{\mathcal{B}_i^b(1^\lambda, T_2)\}_{i\in[R],b\in\{0,1\}}$ corrupts $\mathcal{S}_3$.

According to the T_2-equity condition, for all but a negligible fraction of T_2, the expected outcome of an honest execution of Π_{res} is negligibly different from $\frac{1}{2}$. Since Cleve [Cle86] shows that one of the above adversaries can bias the outcome by a non-negligible amount if in an honest execution of the 2-party protocol, the output is negligibly apart from an unbiased coin and there is an agreement of the outcome, we have

Lemma D.5. *For any fixed T_2, in the residual two party protocol Π_{res}, at least one of the following happens:*

1. *either one of $\{\mathcal{A}_i^0(1^\lambda, T_2)\}_{i\in[R]}, \{\mathcal{B}_i^0(1^\lambda, T_2)\}_{i\in[R]}, \mathcal{A}_0(1^\lambda, T_2)$ can bias the outcome of Π_{res} towards 0 by $\frac{1}{2(4R+1)}$;*
2. *or one of $\{\mathcal{A}_i^1(1^\lambda, T_2)\}_{i\in[R]}, \{\mathcal{B}_i^1(1^\lambda, T_2)\}_{i\in[R]}$ can bias the outcome of Π_{res} towards 1 by $\frac{1}{2(4R+1)}$.*

For all but a negligible fraction of T_2, none of these adversaries in Π_{res} can cause a non-negligible bias towards 1. Otherwise we can construct an adversary $\mathcal{A}^*$ that breaks the wolf-minion condition of Π. Formally,

Lemma D.6. *For all but a negligible fraction of T_2,*

1. *at least one of $\{\mathcal{A}_i^0(1^\lambda, T_2)\}_{i\in[R]}, \{\mathcal{B}_i^0(1^\lambda, T_2)\}_{i\in[R]}, \mathcal{A}_0(1^\lambda, T_2)$ can bias the outcome of Π_{res} towards 0 by $\frac{1}{2(4R+1)}$;*
2. *none of $\{\mathcal{A}_i^b(1^\lambda, T_2)\}_{i\in[R]}, \{\mathcal{B}_i^b(1^\lambda, T_2)\}_{i\in[R]}$ bias the outcome of Π_{res} towards 1 by a non-negligible amount for $b \in \{0,1\}$.*

Proof of Lemma D.6. Suppose that the second claim is not true. Then there exist some polynomials $p(\cdot)$ and $q(\cdot)$ such that, for $1/p(\lambda)$ fraction of T_2, either one of $\{\mathcal{A}_i^b(1^\lambda, \cdot)\}_{i\in[R]}$ or one of $\{\mathcal{B}_i^b(1^\lambda, \cdot)\}_{i\in[R]}$ must be able to bias the outcome of Π_{res} towards 1 by $1/q(\lambda)$ amount. Assume that it is one of $\{\mathcal{A}_i^b(1^\lambda, \cdot)\}_{i\in[R]}$ that can bias the output (the same argument works for $\{\mathcal{B}_i^b(1^\lambda, \cdot)\}_{i\in[R]}$). Consider a fail-stop adversary $\widetilde{\mathcal{A}}$ in the three-party protocol Π that acts as follows.

$\widetilde{\mathcal{A}}$ takes $q(\cdot)$ as an advice and corrupts $\mathcal{S}_2$ and $\mathcal{S}_1$. It randomly chooses a T_2 for $\mathcal{S}_2$ and checks whether this is a "good" T_2 as the following. For each $i \in [R]$, $\widetilde{\mathcal{A}}$ repeats the following for $q^2(\lambda)$ times: it samples a random T_1 and T_3 and simulates an execution of the protocol Π_{res} involving $\mathcal{A}_i^b(1^\lambda, T_2)$. If there exists an $i \in R$ such that the outcome is 1 for more than $\frac{1}{2} + \frac{1}{2q(\lambda)}$ fraction of the time, then we say T_2 is "good" for $\mathcal{A}_i^b(1^\lambda, \cdot)$. If T_2 is "good" for $\mathcal{A}_i^b(1^\lambda, \cdot)$, $\widetilde{\mathcal{A}}$ follows the strategy of $\mathcal{A}_i^b(1^\lambda, \cdot)$; otherwise it follows the honest execution of the protocol.

By the Chernoff bound, except with a negligible probability, for any T_2 such that there exists one $\mathcal{A}_i^b(1^\lambda, \cdot)$ that can bias the outcome of Π_{res} towards 1 by $\frac{1}{q(\lambda)}$ amount, T_2 will be determined as "good" for $\mathcal{A}_i^b(1^\lambda, \cdot)$. This means that $\widetilde{\mathcal{A}}$ can cause a non-negligible bias towards 1, which breaks the wolf-minion condition. Therefore for all but a negligible fraction of T_2, none of $\{\mathcal{A}_i^1(1^\lambda, T_2)\}_{i\in[R]}$ can bias the outcome of Π_{res} towards 1 by a non-negligible amount. By a similar argument, for all but a negligible fraction of T_2, none of $\{\mathcal{B}_i^1(1^\lambda, T_2)\}_{i\in[R]}$ can bias the outcome of Π_{res} towards 1 by a non-negligible amount.

Combining with Lemma D.5, at least one of $\{\mathcal{A}_i^0(1^\lambda, T_2)\}_{i\in[R]}, \{\mathcal{B}_i^0(1^\lambda, T_2)\}_{i\in[R]}, \mathcal{A}_0(1^\lambda, T_2)$ must be able to bias the outcome of Π_{res} towards 0 by $\frac{1}{2(4R+1)}$. □

Now define adversaries $\bar{\mathcal{A}}_i^b(1^\lambda)$, $\bar{\mathcal{B}}_i^b(1^\lambda)$ and $\bar{\mathcal{A}}_0(1^\lambda)$ in Π as follows:

- $\bar{\mathcal{A}}_i^b(1^\lambda)$ corrupts $\mathcal{S}_1$ and $\mathcal{S}_2$. It randomly picks a T_2 and follows the strategy of $\mathcal{A}_i^b(1^\lambda, T_2)$, for $b \in \{0,1\}, i \in [R]$;
- $\bar{\mathcal{B}}_i^b(1^\lambda)$ corrupts $\mathcal{S}_3$ and $\mathcal{S}_2$. It randomly picks a T_2 and follows the strategy of $\mathcal{B}_i^b(1^\lambda, T_2)$, for $b \in \{0,1\}, i \in [R]$.
- $\bar{\mathcal{A}}_0(1^\lambda)$ corrupts corrupts $\mathcal{S}_1$ and $\mathcal{S}_2$. It randomly picks a T_2 and follows the strategy of $\mathcal{A}_0(1^\lambda, T_2)$.

By Lemma D.6 and the definition above, for almost all T_2, at least one of $\{\bar{\mathcal{A}}_i^0(1^\lambda)\}_{i\in[R]}$, $\{\bar{\mathcal{B}}_i^0(1^\lambda)\}_{i\in[R]}$, $\bar{\mathcal{A}}_0(1^\lambda, T_2)$ can bias the output towards 0 by a non-negligible amount. However, in an execution where Π interacting with $\bar{\mathcal{A}}_0(1^\lambda)$, it is same as in an execution of Π where $\mathcal{S}_2$ is honest and $\mathcal{S}_1$ always aborts at the beginning of the protocol. According to the lone-wolf condition, $\bar{\mathcal{A}}_0(1^\lambda)$ cannot bias the output of Π towards 0 by a non-negligible amount. Therefore, at least one of $\{\bar{\mathcal{A}}_i^0(1^\lambda)\}_{i\in[R]}, \{\bar{\mathcal{B}}_i^0(1^\lambda)\}_{i\in[R]}$ can bias the outcome of Π_{res} towards 0 by a non-negligible amount.

Now we show that if $\bar{\mathcal{A}}_i^0(1^\lambda)$ can bias the outcome of Π towards 0 by a non-negligible amount, then $\bar{\mathcal{A}}_i^1(1^\lambda)$ is also able to bias the outcome of Π towards 1 by a non-negligible amount. If this is true, then one of $\{\bar{\mathcal{A}}_i^1(1^\lambda)\}_i, \{\bar{\mathcal{B}}_i^1(1^\lambda)\}_i$ can bias the outcome of Π towards 1 by a non-negligible amount. Consider the following two fail-stop adversaries $\bar{\mathcal{X}}(1^\lambda)$ and $\bar{\mathcal{Y}}(1^\lambda)$:

$\bar{\mathcal{X}}(1^\lambda)$ randomly pick an i from $[R]$ and run $\bar{\mathcal{A}}_i^1(1^\lambda)$; $\bar{\mathcal{Y}}(1^\lambda)$ randomly pick an i from $[R]$ and run $\bar{\mathcal{B}}_i^1(1^\lambda)$.

Then either $\bar{\mathcal{X}}(1^\lambda)$ can cause a non-negligible bias towards 1 or $\bar{\mathcal{Y}}(1^\lambda)$ can cause a non-negligible bias towards 1 in Π. This breaks the wolf-minion condition and the theorem thus follows. So what remains to be shown is that

Lemma D.7. *If $\bar{\mathcal{A}}_i^0((1^\lambda))$ can cause μ-bias towards 0, then $\bar{\mathcal{A}}_i^1((1^\lambda))$ can cause at least $(\mu - \mathsf{negl}(\lambda))$-bias towards 1.*

Proof of Lemma D.7. The randomness of the three parties T_1, T_2 and T_3 together define a sample path. Let S denote the set of sample paths for which $\bar{\mathcal{A}}_i^1(1^\lambda)$ decides to abort *before* sending the i-th message. And $\bar{S}$ denote the set of sample paths for which $\bar{\mathcal{A}}_i^1(1^\lambda)$ decides to abort *after* sending the i-th message. Then by definition of the adversaries, $\bar{\mathcal{A}}_i^0$ will abort *after* sending the i-th message on S and abort *before* sending the i-th message on $\bar{S}$.

Now we define a partition of S and $\bar{S}$. Let $U_0^{\langle b\rangle}$ be the set of sample paths in S on which $\mathcal{S}_3$'s output is 0 when playing with $\bar{\mathcal{A}}_i^b(1^\lambda)$, and $U_1^{\langle b\rangle}$ be the set of sample paths in S on which $\mathcal{S}_3$'s output is 1 when playing with $\bar{\mathcal{A}}_i^b(1^\lambda)$. Then $S = U_0^{\langle b\rangle} \cup U_1^{\langle b\rangle}$. Similarly, let $\bar{U}_0^{\langle b\rangle}$ be the set of sample paths in $\bar{S}$ on which $\mathcal{S}_3$'s output is 0 when playing with $\bar{\mathcal{A}}_i^b(1^\lambda)$, and $\bar{U}_1^{\langle b\rangle}$ be the set of sample paths in $\bar{S}$ on which $\mathcal{S}_3$'s output is 1 when playing with $\bar{\mathcal{A}}_i^b$. Then $\bar{S} = \bar{U}_0^{\langle b\rangle} \cup \bar{U}_1^{\langle b\rangle}$.

Now consider a hybrid adversary, that takes $\bar{\mathcal{A}}_i^1$'s decisions on S and $\bar{\mathcal{A}}_i^0$'s decisions on $\bar{S}$, i.e., it always makes $\mathcal{S}_1$ abort before sending the i-th message. Since this adversary chooses T_2 honestly, an execution with this hybrid adversary is same as an execution in which $\mathcal{S}_1$ is the only corrupted party and always aborts before sending the i-th message. Then by the lone-wolf condition, $\mathcal{S}_3$'s outcome should not be biased towards either direction, except by a negligible amount. Therefore we must have

$$|U_0^{\langle 1\rangle}| + |\bar{U}_0^{\langle 0\rangle}| - (|U_1^{\langle 1\rangle}| + |\bar{U}_1^{\langle 0\rangle}|) \leq \mathsf{negl}(\lambda)$$

By a symmetric argument, consider a hybrid adversary that always makes $\mathcal{S}_1$ abort after sending the i-th message, we have that

$$|U_0^{\langle 0\rangle}| + |\bar{U}_0^{\langle 1\rangle}| - (|U_1^{\langle 0\rangle}| + |\bar{U}_1^{\langle 1\rangle}|) \leq \mathsf{negl}(\lambda)$$

We can conclude that

$$[|U_0^{\langle 0\rangle}| + |\bar{U}_0^{\langle 0\rangle}| - (|U_1^{\langle 0\rangle}| + |\bar{U}_1^{\langle 0\rangle}|)] - [|U_1^{\langle 1\rangle}| + |\bar{U}_1^{\langle 1\rangle}| - (|U_0^{\langle 1\rangle}| + |\bar{U}_0^{\langle 1\rangle}|)] \leq \mathsf{negl}(\lambda)$$

This indicates that $\bar{\mathcal{A}}_i^0$'s ability to bias the output towards 0 is negligibly different from $\bar{\mathcal{A}}_i^1$'s ability to bias the output towards 1, which concludes the lemma. □

□

E Deferred Proofs for Maximin Fairness (Section 6)

E.1 Formal Proof of Lower Bound

We provide a full proof of the following Theorem:

Theorem E.1 (Theorem 6.1, restated: Lower bound for maximin fairness)**.** *Without loss of generality, assume that $n_1 \geq n_0 \geq 1$ and $n_0 + n_1 > 2$. Then there does not exist a maximmin-fair n-party coin toss protocol which tolerates:*

For $n_0 \geq 2$ tolerating $t \geq \lceil \frac{1}{2}(n_0 + n_1) \rceil$ number of **fail-stop** *is impossible*
For $n_0 = 1$ tolerating $t \geq \lceil \frac{1}{2} n_1 \rceil + 1$ number of **semi-malicious** *is impossible*

Proof. We show the two cases.

Case I: $n_1 \geq n_0 \geq 2$: Suppose that there exists a protocol Π that achieves maximin fairness among n_0 number of 0-supporters and n_1 number of 1-supporters against $t = \lceil \frac{1}{2}(n_0 + n_1) \rceil$ fail-stop adversaries. Consider the following partition. $\mathcal{S}_1$ contains $\lceil \frac{1}{2} n_0 \rceil$ number of 0-supporters and $\lfloor \frac{1}{2} n_1 \rfloor$ number of 1-supporters. $\mathcal{S}_2$ contains $\lfloor \frac{1}{2} n_0 \rfloor$ number of 0-supporters and $\lceil \frac{1}{2} n_1 \rceil$ number of 1-supporters. Then Π can be viewed as a two-party coin toss protocol between $\mathcal{S}_1$ and $\mathcal{S}_2$. Moreover, in an all-honest execution, the expected output is $\frac{1}{2}$ due to the correctness.

Since $n_1 \geq n_0 \geq 2$, there is at least one 0-supporter and one 1-supporter in $\mathcal{S}_2$. Consider a fail-stop adversary $\mathcal{A}$ corrupting $\mathcal{S}_1$, which consists of $\lceil \frac{1}{2} n_0 \rceil + \lfloor \frac{1}{2} n_1 \rfloor \leq t$ number of players. Then $\mathcal{A}$ cannot bias the output of Π towards $b \in \{0, 1\}$ by a non-negligible amount. Otherwise it reduces the utility of the honest $(1-b)$-supporters in $\mathcal{S}_2$ by a non-negligible amount, which breaks the maximin fairness of Π. Similarly, $\mathcal{S}_2$ cannot bias the output of Π towards either direction by a non-negligible amount. However, this contradicts with Cleve's lower bound.

Case II: $n_0 = 1$: Chung et al. [CGL+18] proved the impossibility of having a maximin-fair protocol against semi-malicious coalitions of size up to $n-1$. Our proof is similar to Chung et al., but we generalize their proof and characterize the number of corruptions needed more carefully. Suppose that there exists a protocol Π that achieves maximin fairness among one 0-supporter and n_1 number of 1-supporters against $\lceil \frac{1}{2} n_1 \rceil$ semi-malicious adversaries. Consider the following partition. $\mathcal{S}_1$ contains $\lfloor \frac{1}{2} n_1 \rfloor$ number of 1-supporters, $\mathcal{S}_3$ contains $\lceil \frac{1}{2} n_1 \rceil$ number of 1-supporters and $\mathcal{S}_2$ contains the single 0-supporter. Then Π can be viewed as a three-party coin toss protocol between $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$.

One can easily verify that, due to the maximin fairness, Π should satisfy the lone-wolf condition (LBC1) and the wolf-minion condition (LBC2)

- For the lone-wolf condition (LBC1): A single corrupted $\mathcal{S}_1$ (or $\mathcal{S}_3$) with at most $\lceil \frac{1}{2} n_1 \rceil$ players cannot bias the output towards 1 by a non-negligible amount, otherwise it harms the benefit of $\mathcal{S}_2$. Also, it cannot bias towards 0 by a non-negligible amount, otherwise it harms the benefit of $\mathcal{S}_3$ (or $\mathcal{S}_1$).

- For the wolf-minion condition (LBC2): A coalition of $\mathcal{S}_1$ and $\mathcal{S}_2$ (or $\mathcal{S}_3$ and $\mathcal{S}_2$) with at most $\lceil \frac{1}{2} n_1 \rceil + 1$ players cannot bias the output towards 0 by a non-negligible amount, since otherwise this will harm the remaining honest 1-supporters.

If we can further show that Π should also satisfy the T_2-equity condition (LBC3) where T_2 is the single 0-supporter's randomness, then by Theorem D.4 we have a contradiction and thus there is no protocol that achieves maximin fairness among one 0-supporter and n_1 number of 1-supporters against $\lceil \frac{1}{2} n_1 \rceil$ semi-malicious adversaries.

Now we show that indeed, Π should satisfy the T_2-equity condition. Let $f(T_2)$ denote the expected output of an honest execution of Π conditioned on $\mathcal{S}_2$'s randomness T_2. Recall that λ denotes the security parameter. We have the following — we stress that the proof of following Lemma E.2 needs to make use of a *semi-malicious* attack. This is the only place where semi-malicious corruption is needed in the proof of Theorem 6.1 for the case $n_0 = 1$ (*c.f.* Chung et al. [CGL+18] showed that it is possible to tolerate all but one fail-stop corruptions for the case of maximin fairness and $n_0 = 1$).

Lemma E.2. *For any T_2, it must be that $f(T_2) \geq \frac{1}{2} - \frac{1}{p(\lambda)}$ for any polynomial function $p(\cdot)$.*

Proof. The proof was given in Chung et al. [CGL+18]. For completeness, we describe their proof below, and observing that the attack here only needs to corrupt $\mathcal{S}_2$, and the corrupted $\mathcal{S}_2$ must be allowed to choose its coin T_2 to its advantage (note that this is a semi-malicious attack).

For the sake of contradiction, suppose that there exists a T_2^* and a polynomial $q(\cdot)$ such that $f(T_2^*) < \frac{1}{2} - 1/q(\lambda)$, then a semi-malicious adversary corrupting only $\mathcal{S}_2$ can always choose T_2^* as its randomness and can bias the output of Π towards 0 by a non-negligible amount. This breaks the maximin fairness of Π. $\square$

Note that by the correctness of Π, in an all-honest execution, we have that $\mathbb{E}_{T_2}[f(T_2)] = \frac{1}{2}$. Suppose for the sake of contradiction that Π does not satisfy T_2-equity. That is, there exist polynomials $p(\cdot)$ and $q(\cdot)$ such that for $\frac{1}{p(\lambda)}$ fraction of T_2, $|f(T_2) - \frac{1}{2}| > \frac{1}{q(\lambda)}$. By Lemma E.2, there must exists a polynomial $p'(\cdot)$ such that for $\frac{1}{p'(\lambda)}$ fraction of T_2, $f(T_2) > \frac{1}{2} + \frac{1}{q(\lambda)}$. Otherwise $|f(T_2) - \frac{1}{2}| \leq \frac{1}{q(\lambda)}$ for almost all T_2. Again by Lemma E.2, we have that, for any polynomial $q'(\cdot)$,

$$\begin{aligned}
\mathbb{E}_{T_2}[f(T_2)] &\geq \frac{1}{p'(\lambda)} \left(\frac{1}{2} + \frac{1}{q(\lambda)} \right) + \left(1 - \frac{1}{p'(\lambda)} \right) \left(\frac{1}{2} - \frac{1}{q'(\lambda)} \right) \\
&= \frac{1}{2} + \frac{1}{p'(\lambda) q(\lambda)} - \frac{1}{q'(\lambda)} \left(1 - \frac{1}{p'(\lambda)} \right),
\end{aligned}$$

which is greater than $\frac{1}{2}$ for sufficiently large polynomial $q'(\cdot)$. This contradicts the fact that $\mathbb{E}_{T_2}[f(T_2)] = \frac{1}{2}$. Therefore, Π must satisfy T_2-equity. $\square$

E.2 Formal Proof of Upper Bound

Theorem E.3 (Theorem 6.3, restated: Upper bound for maximin fairness)**.** *Without loss of generality, assume that $n_1 \geq n_0 \geq 1$ and $n_0 + n_1 > 2$. There exists a maximin-fair n-party coin toss protocol among n_0 players who prefer 0 and n_1 players who prefer 1, which tolerates up to t malicious adversaries where*

$$t := \begin{cases} \lceil \frac{1}{2}(n_0 + n_1) \rceil - 1, & \text{if } n_0 \geq 2, \\ \lceil \frac{1}{2} n_1 \rceil, & \text{if } n_0 = 1. \end{cases} \tag{3}$$

Proof. Note that except for the special case $n_0 = 1$ and $n_1 = odd$, we can simply run honest-majority MPC with guaranteed output delivery [GMW87, RB89]. For the special case where $n_0 = 1$ and $n_1 = odd$, we have the following result: □

Lemma E.4. *Assume that $n_0 = 1$ and n_1 is odd. Protocol 6.2 satisfies maximin fairness against any non-uniform p.p.t. coalition of size up to $\lceil \frac{1}{2} n_1 \rceil$.*

Proof. According to the protocol, if the single 0-supporter is corrupted and fails to open s_0 correctly, then the protocol outputs 1, which will not harm any honest player. Hence, we only consider the case in which s_0 is successfully opened. We use t_0 and t_1 to denote the number of corrupted 0-supporters and 1-supporters respectively.

Case 1: $t_1 = \lceil \frac{1}{2} n_1 \rceil$**.** Then the single 0-supporter is honest. Due to the hiding property of the commitment scheme, the corrupted coalition's view is computationally independent from s_0 before the 0-supporter opens s_0. Therefore, the final output $s_0 \oplus s_1$ is computationally indistinguishable from a uniform coin.

Case 2: $t_1 < \lceil \frac{1}{2} n_1 \rceil$**.** Then we have honest majority among the 1-supporters. The output of the honest majority MPC will be a uniformly random coin and the honest 1-supporters will win the majority vote. Thus s_1 is a uniformly random coin. Moreover, note that the single 0-supporter commit to s_0 before the honest-majority MPC, and it has to open the coin s_0 correctly, s_0 and s_1 are statistically independent. Therefore, the final output $s_0 \oplus s_1$ is a uniform coin.

Combining the above cases, Protocol 6.2 satisfies maximin fairness against any non-uniform p.p.t. coalition of size up to $\lceil \frac{1}{2} n_1 \rceil$. □