On the Computational Complexity of Coin Flipping

Hemanta K. Maji Manoj Prabhakaran
Department of Computer Science
University of Illinois at Urbana-Champaign
{hmaji2,mmp}@cs.uiuc.edu

Amit Sahai
Department of Computer Science
University of California, Los Angeles
sahai@cs.ucla.edu

***Abstract*—Coin flipping is one of the most fundamental tasks in cryptographic protocol design. Informally, a coin flipping protocol should guarantee both (1) Completeness: an honest execution of the protocol by both parties results in a fair coin toss, and (2) Security: a cheating party cannot increase the probability of its desired outcome by any significant amount. Since its introduction by Blum [1], coin flipping has occupied a central place in the theory of cryptographic protocols. In this paper, we explore what are the implications of the existence of secure coin flipping protocols for complexity theory. As exposited recently by Impagliazzo [2], surprisingly little is known about this question.**

Previous work has shown that if we interpret the Security property of coin flipping protocols very strongly, namely that nothing beyond a negligible bias by cheating parties is allowed, then one-way functions must exist [3]. However, for even a slight weakening of this security property (for example that cheating parties cannot bias the outcome by any additive constant $\epsilon > 0$), the only complexity-theoretic implication that was known was that $\mathbf{PSPACE} \not\subseteq \mathbf{BPP}$.

We put forward a new attack to establish our main result, which shows that, informally speaking, the existence of any (weak) coin flipping protocol that prevents a cheating adversary from biasing the output by more than $\frac{1}{4} - \epsilon$ implies that $\mathbf{NP} \not\subseteq \mathbf{BPP}$. Furthermore, for constant-round protocols, we show that the existence of any (weak) coin flipping protocol that allows an honest party to maintain any noticeable chance of prevailing against a cheating party implies the existence of (infinitely often) one-way functions.

***Keywords*-weak coin-flipping; one-way functions; NP;**

DEDICATION

We dedicate this paper to the memory of Daniel Schreiber (1986 – 2010).

I. INTRODUCTION

A fundamental problem in cryptography is to design protocols that allow two mutually distrusting parties to agree on a random coin. The problem of coin flipping is certainly intrinsically fascinating, but moreover coin flipping protocols have proven to be extremely useful to the theory and design of secure protocols: For example, they are an essential ingredient in all known secure two-party and multi-party computation protocols (e.g. Goldreich, Micali, and Wigderson [4]). They have also proven to be influential more widely: For example, they provide a primary motivation for the utility of the Common Random String (CRS) model [5], one of the most popular models for cryptographic protocol design.

The problem of coin flipping was introduced in the seminal work of Blum [1], who described the task by means of the following scenario: Alice and Bob are divorcing, and have agreed to let the ownership of their favorite car be decided by a coin toss: Heads means that Alice gets the car, and Tails means that Bob gets it. Unfortunately, Alice and Bob are not willing to be in the same room, and need to implement this coin toss over the telephone. As such, Alice and Bob want a protocol such that (informally speaking):

1) The transcript of their conversation uniquely determines who gets the car.
2) If both Alice and Bob behave honestly, then Alice and Bob should both get the car with probability $\frac{1}{2}$.
3) If Alice behaves maliciously but Bob behaves honestly, then Alice cannot significantly increase the probability that she gets the car. Similarly, if Bob behaves maliciously but Alice behaves honestly, then Bob cannot significantly increase the probability that he gets the car.

Such a protocol incentivizes honest behavior by both parties, since they know that deviating from the protocol would not allow them to obtain any significant gain. Here, we are making the non-trivial assumption that both parties *want* to get the car – that is, we do not disallow a cheating Alice to increase the probability of Bob getting the car[1]. As such, this notion of coin flipping is often called "weak" coin flipping (explicitly in the quantum cryptography literature [6]), in contrast to "strong" coin flipping where neither party should

Research supported in part from NSF grants 0916574, 0830803, 0747027, 0716626, 0716389, and 0627781, an equipment grant from Intel, and an Okawa Foundation Research Grant.

[1]Note that by symmetry, our requirements imply that if a protocol fails to meet the requirements, it must be the case that either (1) a single party can bias the outcome significantly in both directions, or (2) both parties can bias the outcome significantly in the *same* direction. If neither of these attacks are possible, then there is always a renaming of Alice and Bob that implies that the protocol meets our requirements.

be able to bias the coin significantly in either direction.[2] Of course, a crucial parameter here is how much bias constitutes a "significant gain."

The Computational Complexity of Coin Flipping: The goal of this paper is to explore the implications of the existence of such (weak[3]) coin flipping protocols for complexity theory. Despite the centrality of randomness and coin flipping to complexity theory and cryptography, surprisingly little is known about this question, as was recently exposited by Impagliazzo[2].

We shall consider a continuum of security guarantees that could be provided by coin tossing protocols, ranging from *fully secure* protocols to *completely insecure* protocols. We say that a coin tossing protocol is $(1-2\epsilon)$-secure if neither party can bias the outcome of a fair coin towards their desired outcome by more that ϵ. So, a 1-secure protocol corresponds to the case where Alice and Bob can not alter the outcome of the coin; and, similarly, a 0-secure protocol is one where Alice or Bob can always force their desired outcome. This intuitive definition suffices to present previously known results and our results in a unified framework. The exact definition is deferred to Section II.

Previous results on the subject show, informally, that if one-way functions do not exist, then it must be possible to bias every D-round coin flipping protocol by an additive $\Theta(1/\sqrt{D})$ factor [3], [7], [8][4]. i.e. every D round protocol is at most $(1-\Theta(1/\sqrt{D}))$-secure. This does show that $(1-\text{negligible})$-secure weak coin flipping implies the existence of one-way functions. (This result is "tight" for this setting, since if one-way functions exist then weak coin flipping protocols do exist that rule out non-negligible additive bias [1], [9], [10], [11].)

However, what about other natural notions of significant gain? For example, what are the consequences of $(1-\epsilon)$-*secure weak coin flipping protocols* , for a constant ϵ.[5]

For both these questions, the only consequences known are of the flavor that $\mathbf{PSPACE} \nsubseteq \mathbf{BPP}$. Indeed, it is not difficult to see that if $\mathbf{PSPACE} \subseteq \mathbf{BPP}$, then for any coin flipping protocol, either a cheating Alice could force the output 1 with probability 1 or Bob could force the output 0 with probability 1. This attack would proceed by using the power of $\mathbf{PSPACE}$ to perform an iterated min-max (actually max-average) computation, with polynomial look-ahead depth for each round of the protocol. The question before us is whether a similar attack (but with relaxed success goals) could be carried out with much less computational power, for instance with only a constant level of look-ahead, or using a max (instead of max-average) computation – something that intuitively can be carried out with only the power of $\mathbf{NP}$ – even though the overall protocol can have polynomially many rounds?

Our Main Result and Intuition: In this work, we show (as our main result) that the existence of any $\left(\frac{1}{2}+\epsilon\right)$-secure weak coin flipping protocol implies that $\mathbf{NP} \nsubseteq \mathbf{BPP}$, where $\epsilon = 1/\text{poly}(k)$. This resolves an open question posed by Impagliazzo [2]: whether $(1-\epsilon)$-secure weak coin flipping protocols are possible if $\mathbf{P} = \mathbf{NP}$ (we show they are not). To prove this result, we introduce a new attack strategy that we call *Hedged Greedy* that is fundamentally different from previous attacks in this setting.

At an intuitive level, previous attacks [3], [8] work by having the attacking party behave honestly until it notices that it has reached a node where its choice will have a significant effect on the expected outcome assuming honest behavior (conditioned on its choices so far) from that point onwards. The attack only deviates from honest behavior at this one point, and the non-triviality of this attack follows from an argument that there must be at least one round in the protocol where the attacker's choice influences the outcome by an additive factor of at least $\Theta(1/\sqrt{D})$ for an D-round protocol.

A conceptually even simpler attack strategy is a "greedy" strategy. To illustrate, let's consider a toy protocol, as described in the protocol tree drawn in Figure 1 below.

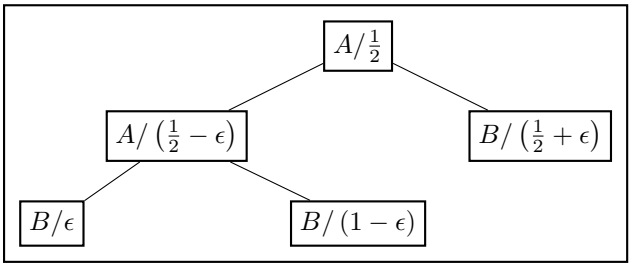

Figure 1. Motivating Hedged-Greedy

In this protocol tree, for instance the annotation "$A/\frac{1}{2}$" on the root node denotes two facts: (1) the *color* of this node is $\frac{1}{2}$, meaning that an honest execution of the protocol from this node would lead to a coin with expected value $\frac{1}{2}$ (i.e. a fair coin), and (2) the first message of the protocol is sent by Alice, and the bit that is sent determines which child of this node corresponds to the next step of the protocol. The honest Alice will place appropriate probabilities on its

[2]This is also closely related to the functionality of "coin flipping with abort," where, conceptually, a fair coin is first produced, but then a cheating party has the option of forcing an abort (possibly after observing the coin). Note that such a protocol immediately implies weak coin flipping, since we can define the output of the protocol to be Tails if Alice forces an abort, and similarly Heads if Bob forces an abort.

[3]Since strong coin flipping and "coin flipping with abort" both imply weak coin flipping, our results of course also apply to the existence of these other types of protocols.

[4]A bias of $\Theta(1/D)$ is proven in [3] (details in [7]), and a bias of $\Theta(1/\sqrt{D})$ can be obtained by combining a probabilistic lemma from [8] with the proof technique of [3].

[5]More precisely, for every constant $\epsilon > 0$, for large enough security parameters 1^k provided as common input to both Alice and Bob, the protocol should not allow additive bias of at least ϵ.

children so as to maintain the value of the coin – in this example at the root node, the honest Alice would proceed left with probability $\frac{1}{2}$, and proceed right otherwise. A leaf node marked "B/ϵ", for instance, just means that honest Bob declares the output to be 1 with probability ϵ, and declares the output to be 0 otherwise (but Bob has full control over the outcome of the protocol if this leaf node is reached).

We would define the "greedy" attack strategy for Alice (when she is attempting to bias the outcome towards 1) to be one where she always proceeds toward the child with higher color. As this example illustrates, however, this attack may obtain only a tiny additive bias: Here, greedy Alice would proceed to the right child at the root of the tree, obtaining an additive bias of only ϵ, which can be arbitrarily small. (In this example, however, greedy Bob would still be quite successful. In the full version of this paper, we show an extension of this example where greedy strategies for both Alice and Bob perform poorly.)

In this example, the optimal strategy for Alice is in fact to always proceed to the left child; however, in more complex versions of this example (where "dummy" rounds are added in between the actual rounds of the protocol), it may be very difficult for an attacking Alice to realize that she would have near complete control of the outcome on the left branch of the tree. Given only a bounded look-ahead capability to observe nodes in the protocol tree, for instance, Alice would not be able to ascertain whether or not she can control the output of the protocol on the left due to "dummy" rounds.

Instead, the basic intuitive idea behind our attack strategy is to implement a *hedged greedy* strategy: instead of always following the greedy strategy, our attack will "hedge its bet" by also proceeding to the other child with some probability. Intuitively, when the advantage of behaving greedily is clearer, the attack will potentially deviate more from honest behavior. In the example above, at the root node, since the values of the children are so close, the hedged greedy Alice strategy will place almost equal probabilities to both children (behaving very much like the honest Alice would). But at the level below, on the left, where the values of the two children are so different, the hedged greedy Alice strategy would proceed to the node with value $(1-\epsilon)$ with probability very close to 1 (thus deviating very strongly from how honest Alice would behave at this node).

In the example above, the hedged greedy Alice strategy would be able to bias the coin to nearly $\frac{3}{4}$. Through a careful choice of the exact hedging behavior of our strategy, we show that in fact we can guarantee similar performance for *any* protocol – in the sense that either hedged greedy Alice will be able to bias to at least roughly $\frac{3}{4}$ or hedged greedy Bob will be able to bias to at most roughly $\frac{1}{4}$. In fact we prove a more general tradeoff between the relative success of hedged greedy Alice and hedged greedy Bob: for example, if hedged greedy Bob *does not* significantly bias the outcome below $\frac{1}{2}$, then we show that hedged greedy Alice must be able to bias the outcome all the way to roughly 1. (In the example above, however, note that even greedy Bob would be able to guarantee the output 0. As such, the example above only illustrates the idea behind our attack, not the actual analysis of it, which is fairly delicate. We are not aware of any simpler attack and analysis that even guarantees a tiny constant additive bias.) We also show in the full version of the paper that our analysis of our attack is tight, by showing that in fact *any* attack that bases its decisions on only a bounded look-ahead view of the protocol tree (including the colors of the nodes, as illustrated above) cannot obtain better bias.

At a technical level, our proof proceeds in two stages: First, we use the power of $\mathbf{NP}$ to convert an arbitrary coin flipping protocol to a *stateless* coin flipping protocol with nearly identical security guarantees. In a stateless protocol, honest parties only need the transcript of the protocol so far (and fresh randomness) to determine their next move. We then show an unconditional polynomial-time "hedged greedy" attack on any stateless protocol. We believe this modular approach may be of independent interest.

A stronger result for constant-round protocols: For constant round protocols, it is not difficult to see that the "$\mathbf{PSPACE}$" attack described above can be implemented in $\mathbf{PH}$ and recall that if $\mathbf{NP} \subseteq \mathbf{BPP}$, then $\mathbf{PH} \subseteq \mathbf{BPP}$ [12]. A natural question is whether any potentially stronger consequence is true.

For this setting, informally speaking, we obtain essentially the best possible result[6]: even the existence of negligible-secure weak coin flipping protocols implies the existence of (infinitely often) one-way functions. The core difference between the setting of $\mathbf{NP} \subseteq \mathbf{BPP}$ and the non-existence of one-way functions is that in the latter case, one only obtains an inverse sampler and approximator that works with high probability over a fixed distribution of inputs. Our attack works by showing how to combine a constant number of inverters for a constant number of related functions to carry out the desired attack.

Conclusions and Future Directions.: This work revisits the fundamental question of the computational complexity implications of the existence of coin flipping protocols, where surprisingly little was known. We provide new results which show that in many natural settings of parameters, weak coin flipping protocols in fact imply $\mathbf{NP} \not\subseteq \mathbf{BPP}$, where previously only $\mathbf{PSPACE} \not\subseteq \mathbf{BPP}$ was known. We do this by introducing new techniques for this setting, including a *hedged greedy* attack strategy and a method for its analysis.

A number of important natural questions remain open: For the parameters that we consider in our main result, can we conclude that one-way functions exist (and not just that

[6]The protocol in [1] along with the results of [9], [10], [11] provides a constant round $(1 - \text{negligible})$-secure coin tossing protocol, if one-way functions exist.

$\mathbf{NP} \not\subseteq \mathbf{BPP}$)? Is it possible that a constant-secure weak coin flipping protocol (with polynomially many rounds) can exist even if $\mathbf{NP} \subseteq \mathbf{BPP}$?

II. Preliminaries and Conventions

Consider any 2-party protocol π. We view the transcript generation procedure as traversal of a tree, called the *transcript tree* of π. Any transcript prefix v is a node in this tree and the two extensions of the transcript $v0$ and $v1$ are its two children in the tree. The leaves of the tree are labeled with an output 0 or 1. The depth of the tree D is the communication complexity (maximum number of bits exchanged) of the protocol. Each node is annotated as an Alice node or a Bob node, indicating which party must send the next message in the protocol. When a polynomial blow-up in the round complexity of the protocol is not important, we may consider the Alice and Bob nodes as alternating in any path in the tree. (In Section IV, where the number of rounds is important, we remove the restriction that the tree is binary, but will retain the convention that Alice and Bob nodes alternate.)

The protocol is specified by a randomized algorithm f_π which takes as input a transcript prefix v and a private "state" and outputs an updated state and a next bit (or, if v is a complete transcript, produces a deterministic binary output based only on v). A protocol is called *stateless* if the state variable is always empty.

Let χ_v be the probability of the output of the protocol being 1 conditioned on v being a prefix of the final transcript (when both parties honestly follow the protocol). We call this the *color* of the node v. We shall denote the subtree rooted at v by S_v. We will assume that the height D of the protocol π is the security parameter. When we mention that some event occurs with high probability (written as w.h.p.) it implies that the probability of that event is at least $1-\exp(-\Theta(D+1/\epsilon))$.

We define a μ-secure protocol for a χ^*-weak coin as follows:

Definition 1 (μ-secure implementation of χ^*-Weak coin)**.** *For $\chi^* \in [0,1]$ and $\mu \in [0,1]$, let $\chi^+ = 1 - \mu(D)(1-\chi^*)$ and $\chi^- = \mu(D)\chi^*$. A protocol π is said to be a μ-secure implementation of χ^* weak coin-flipping, if the outcome is a χ^*-coin if both parties follow the protocol honestly, and either*

1) (Secure when Alice wants 1 and Bob wants 0) *For any efficient (PPT) adversarial Alice strategy, the expected outcome of the protocol when playing against the honest Bob strategy is no higher than χ^+ and for any efficient Bob strategy, the expected outcome when playing against the honest Alice strategy is no lower than χ^-, or*
2) (Secure when Alice wants 0 and Bob wants 1) *For any efficient (PPT) adversarial Alice strategy, the expected outcome of the protocol when playing against the honest Bob strategy is no lower than χ^- and for any efficient Bob strategy, the expected outcome when playing against the honest Alice strategy is no higher than χ^+.*

With increasing μ, the protocol has a better security guarantee: if $\mu = 1$, then neither party can bias the coin away from χ^* towards their desired outcome. Against an adversary with access to a $\mathbf{PSPACE}$ oracle, it is easy to see that any efficient protocol is 0-secure. Our attack in Section III renders any protocol about $\frac{1}{2}$-secure; for $\chi^* = \frac{1}{2}$, this means that some party can bias the protocol to about $\frac{1}{4}$ (if its desired outcome is 0) or to about $\frac{3}{4}$ (if its desired outcome is 1). (Our attack in Section IV on the other hand renders any protocol μ secure with μ close to 0.)

In Section IV we show that if a constant round weak coin-flipping protocol must be secure, then a standard weaker variant of one-way functions, called infinitely-often one-way functions must exist. This variant appears in earlier work like [13], but seems to have been named so in [14]. A polynomial time computable function $f : \{0,1\}^* \to \{0,1\}*$ is called an infinitely-often one-way function if for any polynomial p and any PPT adversary A, if for infinitely many values n, $\Pr_{x \leftarrow \{0,1\}^n}[f(A(f(x)) = f(x)] < \frac{1}{p(n)}$ (where the probability is also over the coins of A). Thus, if f is not an infinitely-often one-way function, then there is a PPT adversary A which *for all but finitely many values of n* has a significant probability of inverting f on random inputs from $\{0,1\}^n$.

III. Complexity of Weak Coin-Tossing

In this section we show our main result, that if there is a polynomial time weak coin-tossing protocol, then $\mathbf{NP} \not\subseteq \mathbf{BPP}$. In fact, we show that any weak coin-tossing protocol can be attacked significantly (biasing the outcome by close to 0.75) by polynomial time adversaries with access to an $\mathbf{NP}$ oracle. We arrive at this result in a few steps:

- First, we observe that for any polynomial time protocol π, there exists a *state-less* protocol π' that runs in polynomial time with access to an $\mathbf{NP}$ oracle, such that π' is "as secure as" π when considering adversaries with access to $\mathbf{NP}$ oracles. (Lemma 4.)
- Next we show that, unconditionally, any state-less protocol for weak coin-flipping can be attacked efficiently, using just the protocol itself as a black-box.

Together, these give an attack on any polynomial time protocol π, wherein the attack will use an $\mathbf{NP}$ oracle (which will be used to implement the state-less protocol π' that will be accessed as a black-box by the attack).

The first of these follows rather easily from a result on uniform generation of $\mathbf{NP}$-witnesses given an $\mathbf{NP}$ oracle [15], [16]. (See Lemma 4.) We remark that while much weaker computational power (namely inverting a one-way function) is enough for carrying out such a reconstruction

in the normal course of the protocol, it is much harder to ensure that the reconstruction works with adequate accuracy even when the protocol is under attack and may result in a transcript distribution significantly different from that in the normal execution. However, by [15], [16], such reconstruction can be accurately carried out for any transcript history when an $\mathbf{NP}$-oracle is given.

Our main work then is in showing an attack on a stateless protocol. Surprisingly we can do this efficiently using the protocol itself as a black-box, and with no further computational complexity assumption. In Section III-A we provide an intuition for the way the attack works, assuming we have certain additional oracles related to the protcol. Then in Section III-B we present the actual attack, which involves additional checks to make the simpler attack robust, and then replaces the oracles it required by approximate implementations.

A. Simplified Sketch of the Attack

In this section we describe an attack on any weak coin flipping protocol π, given an oracle that, for any partial transcript v, can return χ_v, the color of v in the protocol π.

Given such an oracle for the colors, we define four attacks, two each for corrupt Alice and corrupt Bob — for each party, one to bias the outcome towards 0 and one to bias it towards 1. In Figure 2, we describe the attack for Alice to bias the outcome towards 1; the other attacks are symmetric.

Note that indeed $t_0 + t_1 = \frac{(p_0\chi_0+p_1\chi_1)-(p_0+p_1)\chi_0\chi_1}{\chi-\chi_0\chi_1} = 1$, since $p_0\chi_0 + p_1\chi_1 = \chi$ and $p_0 + p_1 = 1$.

We shall show that our choice of the probabilities t_0 and t_1 are such that no matter what the protocol is, these attacks break the security of the protocol. More precisely, if we denote the four attacks by $\mathsf{Adv}_A^{(0)}$, $\mathsf{Adv}_A^{(1)}$, $\mathsf{Adv}_B^{(0)}$ and $\mathsf{Adv}_B^{(1)}$ (with $\mathsf{Adv}_A^{(0)}$ corresponding to Alice trying to bias towards 0 and so on), we show that in any such protocol either $\mathsf{Adv}_A^{(1)}$ biases the outcome to 1 with probability at least 0.75, or $\mathsf{Adv}_B^{(0)}$ biases the outcome to 0 with probability at least 0.75. Also, either $\mathsf{Adv}_A^{(0)}$ biases the outcome to 0 or $\mathsf{Adv}_B^{(1)}$ biases the outcome to 1 with probability at least 0.75. Then, the protocol π is not a secure weak coin-flipping protocol.

In order to analyze these attacks, we will define the following score function. Consider an attack whose goal is to bias towards a bit b. If the original color at a node is χ, but the attack changes it to x, then we let the score for (the failure of) the attack at that node be $s_b(x,\chi) := \frac{|b-x|}{|b-\chi|}$ (for $\chi \neq b$). That is,

$$s_1(x,\chi) = \frac{1-x}{(1-\chi)} \qquad s_0(x,\chi) = \frac{x}{\chi}$$

In addition, we define $s_0(0,0) := 0$ and $s_1(1,1) := 0$. Note that the lower the score, the better the performance in biasing towards b.

For any node v in the transcript tree, let $A^{(0)}(v), A^{(1)}(v), B^{(0)}(v), B^{(1)}(v)$ denote the colors induced at the node v by our four attacks. Then, we will show that:

$$\begin{aligned} &A^{(1)}(v), B^{(1)}(v) \in [\chi_v, 1], \quad \text{and} \\ &A^{(0)}(v), B^{(0)}(v) \in [0, \chi_v] \end{aligned} \tag{1}$$

$$s_1(A^{(1)}(v), \chi_v) + s_0(B^{(0)}(v), \chi_v) \leq 1 \tag{2}$$

$$s_0(A^{(0)}(v), \chi_v) + s_1(B^{(1)}(v), \chi_v) \leq 1 \tag{3}$$

Intuitively, equation (1) says that our attacks are at least as good as an honest execution of the protocol. And the inequalities in equation (2) and equation (3) state that if Alice fails to bias the output to b by a significant amount then Bob can bias the output to $(1-b)$ by a significant amount. More precisely, when v is the root of a protocol that yields a fair coin under honest execution ($\chi_v = \frac{1}{2}$), the first equation above shows that either $s_1(A^{(1)}(v), \chi_v) \geq \frac{1}{2}$ (which implies that $A^{(1)}(v) \geq \frac{3}{4}$) or $s_0(B^{(0)}(v), \chi_v) \geq \frac{1}{2}$ (which implies that $B^{(0)}(v) \leq \frac{1}{4}$). That is, the protocol is not secure against an Alice who prefers 1 and a Bob who prefers 0. Similarly, the second equation shows that protocol is not secure when Alice and Bob prefer 0 and 1 respectively either.

We will prove the result by induction on the height h of S_v the subtree rooted at v. If $h = 1$, it is trivial to see that both the conditions are satisfied. Let the four tuple associated with the performance of our attack on the S_{vb} be $(A_b^{(0)}, A_b^{(1)}, B_b^{(0)}, B_b^{(1)}) = (A^{(0)}(vb), A^{(1)}(vb), B^{(0)}(vb), B^{(1)}(vb))$. We will only show how the induction works for the first case, i.e. $s_1(A^{(1)}(v), \chi_v) + s_0(B^{(0)}(v), \chi_v) \leq 1$. By induction hypothesis, we know that: $B_b^{(0)} \leq \chi_b$ and

$$\frac{1-A_b^{(1)}}{(1-\chi_b)} + \frac{B_b^{(0)}}{\chi_b} \leq 1 \implies (1-A_b^{(1)}) \leq \left(1 - \frac{B_b^{(0)}}{\chi_b}\right)(1-\chi_b).$$

(This inequality in fact holds for the extreme cases of $\chi_0 = 0$ and $\chi_0 = 1$ as well: when $\chi_0 = 1$, we have $A_1^{(0)} \in [\chi_0, 1] \implies A_1^{(0)} = 1$; when $\chi_0 = 0$, then $B_0^{(0)} = 0$ and our convention for the score will interpret $\frac{B_0^{(0)}}{\chi_0}$ as 0.)

Suppose v is an Alice node and she outputs b as the next message with probability t_b, where $b \in \{0,1\}$. Then $A^{(1)}(v) = t_0 A_0^{(1)} + t_1 A_1^{(1)}$ and $B^{(0)}(v) = p_0 B_0^{(0)} + p_1 B_1^{(0)}$.

$$\begin{aligned} &s_1(A^{(1)}(v),\chi) + s_0(B^{(0)}(v),\chi) \\ &\quad = \frac{1-A^{(1)}(v)}{(1-\chi)} + \frac{B^{(0)}(v)}{\chi} \\ &\quad = \frac{t_0(1-A_0^{(1)}) + t_1(1-A_1^{(1)})}{(1-\chi)} + \frac{p_0 B_0^{(0)} + p_1 B_1^{(0)}}{\chi} \\ &\quad \leq B_0^{(0)} T_0 + B_1^{(0)} T_1 + \frac{t_0(1-\chi_0) + t_1(1-\chi_1)}{(1-\chi)} \end{aligned}$$

Intuition of Attack $\mathsf{Adv}_A^{(1)}$

A D round protocol π with for a χ^*-coin is given. We have access to an oracle which provides the exact color χ_v of any node v.

Suppose the protocol is currently at an Alice-node v (i.e., the next message is sent by Alice). Let $v0$ and $v1$ be its two children. For convenience we write χ, χ_0 and χ_1 respectively for χ_v, χ_{v0}, and χ_{v1}. Let $p_0 = \Pr_\pi[v0|v]$ and $p_1 = \Pr_\pi[v1|v]$ so that $p_0 + p_1 = 1$ and $\chi = p_0\chi_0 + p_1\chi_1$. Given χ, χ_0 and χ_1 we can calculate p_0 and p_1.

- Let $t_b = \frac{p_b\chi_b\left(1-\chi_{(1-b)}\right)}{(\chi-\chi_0\chi_1)}$, for $b \in \{0,1\}$. Send 0 as the next message with probability t_0 and 1 as the next message with probability t_1.

Figure 2. *Intuition of Attack $\mathsf{Adv}_A^{(1)}$ for Alice to bias towards outcome 1.*

where $T_0 = \left[\frac{p_0}{\chi} - \frac{t_0(1-\chi_0)}{(1-\chi)\chi_0}\right]$ and $T_1 = \left[\frac{p_1}{\chi} - \frac{t_1(1-\chi_1)}{(1-\chi)\chi_1}\right]$. If we show that $T_0 \geq 0$ and $T_1 \geq 0$, then indeed

$$\begin{aligned} & s_1(A^{(1)}(v),\chi) + s_0(B^{(0)}(v),\chi) \\ \leq & \chi_0 T_0 + \chi_1 T_1 + \frac{t_0(1-\chi_0) + t_1(1-\chi_1)}{(1-\chi)} \\ = & \frac{p_0\chi_0 + p_1\chi_1}{\chi} = 1 \end{aligned}$$

(using the fact that $B_b^{(0)} \leq \chi_b$). Now, substituting $t_0 = \frac{p_0\chi_0(1-\chi_1)}{(\chi-\chi_0\chi_1)}$ and $t_1 = \frac{p_1\chi_1(1-\chi_0)}{(\chi-\chi_0\chi_1)}$, we observe that

$$\begin{aligned} T_0 &= \frac{p_0\left[(1-\chi)(\chi-\chi_0\chi_1) - \chi(1-\chi_0)(1-\chi_1)\right]}{\chi(1-\chi)(\chi-\chi_0\chi_1)} \\ &= \frac{p_0(\chi_1-\chi)(\chi-\chi_0)}{\chi(1-\chi)(\chi-\chi_0\chi_1)} \geq 0 \end{aligned}$$

(using the fact that $\min\{\chi_0,\chi_1\} \leq \chi \leq \max\{\chi_0,\chi_1\}$), and similarly $T_1 \geq 0$.

Now we need to show that $A^{(1)}(v) \in [\chi, 1]$ and $B^{(0)}(v) \in [0,\chi]$. Note that if $\chi_0 \geq \chi_1$ then $t_0 \geq p_0$ and if $\chi_1 \geq \chi_0$, then $t_1 \geq p_1$. Hence we have $A^{(1)}(v) = t_0\chi_0 + t_1\chi_1 \geq p_0\chi_0 + p_1\chi_1 = \chi$. Also, since $B_0^{(0)} \leq \chi_0$ and $B_1^{(0)} \leq \chi_1$, we have $B^{(0)}(v) = p_0B_0^{(0)} + p_1B_1^{(0)} \leq p_0\chi_0 + p_1\chi_1 = \chi$. This completes the analysis of this simplified attack, which assumes t_0 and t_1 can be computed correctly.

Our actual attack is significantly complicated than the one explained in this section, by the fact that we do not have oracles to find χ_v exactly (even given an **NP** oracle). In fact, we can only estimate χ_v with a small additive error term. The effect of this error on our attack can be severe when χ is very close to 1 or $(\chi - \chi_0\chi_1)$ is very small. The actual attack takes care of these special cases separately.

B. The Actual Attack

In this section, we first describe our actual attack against a stateless protocol and assuming existence of three oracles Π, Π_H and Π_T. The oracle Π is the next message oracle for the stateless protocol π; and the oracle Π_H (resp. Π_T) extends a partial transcript by one bit conditioned on the event that the output of the transcript is heads (resp. tails). The attack closely follows the intuition above, but significantly differs in the details and the analysis. The differences arise from the fact that the above attack depended on accurately knowing certain ratios, which simply cannot be estimated sufficiently accurately.

As described in Figure 3, the attack has two additional checks before carrying out an approximate version of the above attack. Firstly, if the color of the current node is very close to 0 or 1, the attack continues by simply following the protocol honestly (even if it later encounters nodes with different colors). Note that this is done even on reaching a node with the color opposite to what the attack desires. The second check is more subtle, and is designed to handle the technical difficulty in accurately estimating t_0 and t_1 when the denominator is close to 0. In the case of $\mathsf{Adv}_A^{(1)}$ (Alice biasing towards 1), this denominator is $\chi - \chi_0\chi_1$; if we see that χ is close to $\min\{\chi_0,\chi_1\}$, then the current step of the attack is changed to weigh the two children using the contribution (in the honest execution) that they make to the color of the current node, i.e., using probabilities $h_b = \frac{p_b\chi_b}{\chi}$ instead of t_b. Note that if we call $\Pi_H(v)$ then it outputs bit b with probability h_b. So, we use the Π_H oracle to send the next message in this case[7]. If these checks pass, then the original attack (but with ratios calculated according to the estimated values) is carried out.

The various quantities which we used in our attack can be efficiently computed with significant accuracy. The following result summarizes our estimation algorithm:

Lemma 1 (Estimation)**.** *In the oracle world we can efficiently find $\tilde{\chi}$, $\tilde{\chi}_0$, $\tilde{\chi}_1$, $\tilde{p}_b$, $\tilde{h}_b$ and $\tilde{t}_b$ such that, w.h.p.:*

1) $\left|\tilde{\chi}-\chi\right|, \left|\tilde{\chi}_0-\chi_0\right|, \left|\tilde{\chi}_1-\chi_1\right|, \left|\tilde{p}_c-p_c\right| \leq \lambda$, *and*
2) $\left|\tilde{h}_b - h_b\right| \leq 3\lambda^{1/3}$, *if $\chi \geq \delta \geq \lambda^{1/3}$ and $\lambda \leq 1/3$.*
3) $\left|\tilde{t}_b - t_b\right| \leq 9\lambda^{1/3}$, *if $\chi - \chi_0\chi_1 \geq \lambda^{1/3}$ and $\lambda \leq \frac{1}{2^9}$.*

Proof Sketch: χ can be approximated by generating $N = (D+1/\epsilon)/\lambda^2$ transcripts (using the next message oracle Π) and taking $\tilde{\chi}$ to be the fraction of transcripts with

[7]When Bob wants to bias the outcome towards 0, i.e. $\mathsf{Adv}_B^{(0)}$, he will use the oracle Π_T in this step.

Attack $\mathsf{Adv}_A^{(1)}$

A D round protocol π with bias χ^* (along with corresponding oracles Π and Π_H) is given. The attack is parametrized by a function of the security parameter $0 < \epsilon < 1$. Let $\delta = \min\left\{\frac{(1-\chi^*)\epsilon}{4}, \frac{\chi^*\epsilon}{4}\right\}$ and $\lambda = \min\left\{\frac{\delta^3}{3^6 D}, \frac{\epsilon^3\delta^3}{(72)^3 D^3}, \frac{1}{2^9}\right\} = \frac{\epsilon^3\delta^3}{(72)^3 D^3}$.

Alice performs the following attack at all nodes v where she is supposed to send the next bit. First, compute the following estimates, as described in Lemma 1. Let $\tilde{\chi}$ be an estimate of $\chi := \chi_v$, so that $|\tilde{\chi} - \chi| \leq \lambda$ w.h.p.. Similarly, let $\tilde{\chi}_0$ and $\tilde{\chi}_1$ be estimates of $\chi_0 := \chi_{v0}$ and $\chi_1 := \chi_{v1}$ respectively. Then proceed as follows:

1) If $\tilde{\chi} \geq 1 - (\delta + \lambda)$ or $\tilde{\chi} \leq (\delta + \lambda)$: Henceforth, follow the protocol honestly by making calls to Π. This case takes care of nodes in the transcript tree such that χ is too close to 0 or 1.
2) Else, if $\tilde{\chi} - \min\{\tilde{\chi}_0\, \tilde{\chi}_1\} < \lambda^{1/3} + 2\lambda$, then output $d = \Pi_H(v)$ as the next message. This case takes care of nodes in the transcript tree such that $\chi - \chi_0\chi_1$ is too small.
3) Else (here, $\chi \in [\delta, 1-\delta]$ and $\chi - \min\{\chi_0, \chi_1\} \geq \lambda^{1/3}$), we perform a variant of our original attack. Let $c' \in \{0,1\}$ be such that $\min\{\tilde{\chi}_0, \tilde{\chi}_1\} = \tilde{\chi}_{c'}$ (i.e., the child with lower estimated probability of heads). Let p_0 and p_1 be the probabilities assigned by π to the two possible next messages at v, so that $p_0 + p_1 = 1$ and $\chi = p_0\chi_0 + p_1\chi_1$. Let $\tilde{h}_{c'}$ be an estimation of $h_{c'} = \frac{p_{c'}\chi_{c'}}{\chi}$ such that $\left|\tilde{h}_{c'} - h_{c'}\right| \leq 3\lambda^{1/3}$ (see Lemma 1). Evaluate $\tilde{t}_{c'}$ which is an approximation of
$$t_{c'} = \frac{p_{c'}\chi_{c'}\left(1 - \chi_{(1-c')}\right)}{(\chi - \chi_0\chi_1)},$$
such that $\left|\tilde{t}_{c'} - t_c\right| \leq 9\lambda^{1/3}$ (Lemma 1). Set $\tilde{r}_{c'} = \min\{\tilde{t}_{c'}, \max\{0, \tilde{h}_{c'} - 3\lambda^{1/3}\}\}$.
Send the bit c' with probability $\tilde{r}_{c'}$ and send $1 - c'$ with probability $1 - \tilde{r}_{c'}$.

Figure 3. *Attack $\mathsf{Adv}_A^{(1)}$ for Alice to bias towards outcome 1.*

outcome 1. By a Chernoff bound, we can conclude that $\tilde{\chi}$ is close to χ w.h.p. Similarly, we can estimate p_b.

The quantities $h_b = \frac{p_b\chi_b}{\chi}$ and $t_b = \frac{p_b\chi_b(1-\chi_{(1-b)})}{\chi-\chi_0\chi_1}$ are estimated by substituting the approximate values of p_b, χ, χ_0 and χ_1 instead of their actual values. Bounds on the denominators ensure that the resulting error is bounded. ■

The oracles Π_H and Π_T can be substituted by statistically close approximations using only the next message oracle Π, when the color of the queried node is not too close to 0 or 1.

Lemma 2. *Given black-box access to the next message function of a stateless protocol π (Π), we can provide statistically close approximations of Π_H and Π_T on queries v such that $\chi_v \in [\delta, 1-\delta]$.*

Proof Sketch: To implement Π_H, we generate $(D + 1/\epsilon)/\delta$ transcripts with prefix v using the oracle Π. If there are no transcripts with outcome 1, then we output the next bit as 0. Otherwise, we pick the first transcript with outcome 1 and output the bit following v in that transcript. It is easy to see that this is a statistically close approximation of Π_H if $\chi_v \geq \delta$. Similarly, we can also implement Π_T. ■

At the heart of the analysis is an analogue of the inductively maintained inequalities equation (1)-equation (3). These inequalities had depended on the fact the we could arrange the quantities T_0 and T_1 to be positive. Unfortunately, this is no more the case in the analysis of the actual attack. But by carefully choosing our parameters, we can carry out a case analysis and prove:

Lemma 3. *For a stateless weak coin-flipping protocol π, if v is a node in the protocol tree at height h, we have $A^{(0)}(v), B^{(0)}(v) \in [0, \chi_v]$ and $A^{(1)}(v), B^{(1)}(v) \in [\chi_v, 1]$; and*
$$s_1(A^{(1)}(v), \chi_v) + s_0(B^{(0)}(v), \chi_v) \leq 1 + \frac{\delta}{\chi_v(1-\chi_v)} + \nu_h$$
$$s_0(A^{(0)}(v), \chi_v) + s_1(B^{(1)}(v), \chi_v) \leq 1 + \frac{\delta}{\chi_v(1-\chi_v)} + \nu_h$$
where $A^{(b)}(v)$ (resp. $B^{(b)}$), for $b \in \{0,1\}$, is the expectation of the outcome when Alice runs the attack $\mathsf{Adv}_A^{(b)}$ against honest Bob (resp. Bob runs $\mathsf{Adv}_B^{(b)}$ against honest Alice), and $\nu_0 = 0$ and $\nu_{h+1} = \frac{9\lambda^{1/3}}{\delta} + \nu_h\left(1 + \frac{9\lambda^{1/3}}{\delta}\right)$.

Finally, we can use Lemma 3 to prove the following result:

Theorem 1. *Let π be a D round stateless coin-flipping protocol with expected outcome (under honest execution) $\chi \in (0,1)$. For any function (of the security parameter) $0 < \epsilon < 1$ define $\chi^- = \chi - \frac{\chi}{2}(1-\epsilon)$, and $\chi^+ = \chi + \frac{(1-\chi)}{2}(1-\epsilon)$. Then there exist attacks $\mathsf{Adv}_A^{(0)}$, $\mathsf{Adv}_A^{(1)}$, $\mathsf{Adv}_B^{(0)}$ and $\mathsf{Adv}_B^{(1)}$ which use black-box access to π and run in poly$(\frac{1}{\epsilon} + D)$ time, such that*

1) *$A^{(1)} \geq \chi^+$ or $B^{(0)} \leq \chi^-$, (i.e., not secure if Alice wants 1 and Bob wants 0)*
2) *and, $B^{(1)} \geq \chi^+$ or $A^{(0)} \leq \chi^-$ (i.e., not secure if Bob wants 1 and Alice wants 0).*

where $A^{(b)}$ (resp. $B^{(b)}$), for $b \in \{0,1\}$, is the expectation of the outcome when Alice runs the attack $\mathsf{Adv}_A^{(b)}$ against honest Bob (resp. Bob runs $\mathsf{Adv}_B^{(b)}$ against honest Alice).

As a corollary of Theorem 1, we can conclude that:

Corollary 2. *If a stateless protocol π is a μ-secure polynomial time implementation of χ^*-weak coin for any constant $0 < \chi^* < 1$, then $\mu \leq \frac{1}{2} + negl(D)$, where D is the number of rounds in π and negl is a negligible function.*

Note that an ideal secure protocol for weak coin-flipping would be a 1-secure protocol. Relative to adversaries with access to a $\mathbf{PSPACE}$ oracle, all protocols are 0-secure protocols. Our attack is not as effective as an attack with a $\mathbf{PSPACE}$ oracle, but instead renders any (stateless) protocol at most $\frac{1}{2} + \text{negl}(D)$-secure.

General protocols: Access to an $\mathbf{NP}$-oracle can be used to reconstruct a correctly distributed random tape for a party in a protocol, using just the public history of the protocol. We remark that while much weaker computational power (namely inverting a one-way function) is enough for carrying out such a reconstruction in the normal course of the protocol, it is much harder to ensure that the reconstruction works with adequate accuracy even when the protocol is under attack and may result in a transcript distribution significantly different from that in the normal execution. However, by a result on uniform generation of $\mathbf{NP}$-witnesses given an $\mathbf{NP}$ oracle [15], [16], such reconstruction can be accurately carried out for any transcript history when an $\mathbf{NP}$-oracle is given. More formally, we need the following result.

Lemma 4. *For any polynomial time protocol π for χ^* weak coin-flipping that is μ-secure against polynomial time adversaries with access to $\mathbf{NP}$ oracles, there is a* state-less *protocol π' that runs in (expected) polynomial time with access to an $\mathbf{NP}$ oracle, and is also a χ^* weak coin-flipping that is μ-secure against polynomial time adversaries with access to $\mathbf{NP}$ oracles.*

Finally, from Theorem 1 and Lemma 4 we obtain our main result.

Theorem 3. *Let π be a polynomial time (possibly stateful) coin-flipping protocol with expected outcome (under honest execution) $\chi \in (0,1)$. For any function (of the security parameter) $0 < \epsilon < 1$ define $\chi^- = \chi - \frac{\chi}{2}(1-\epsilon)$, and $\chi^+ = \chi + \frac{(1-\chi)}{2}(1-\epsilon)$. Then there exist attacks $\mathsf{Adv}_A^{(0)}$, $\mathsf{Adv}_A^{(1)}$, $\mathsf{Adv}_B^{(0)}$ and $\mathsf{Adv}_B^{(1)}$ which use an $\mathbf{NP}$ oracle, but otherwise run in poly$(\frac{1}{\epsilon} + D)$ time, such that at*

1) $A^{(1)} \geq \chi^+$ or $B^{(0)} \leq \chi^-$,
2) and, $B^{(1)} \geq \chi^+$ or $A^{(0)} \leq \chi^-$,

where $A^{(b)}$ (resp. $B^{(b)}$), for $b \in \{0,1\}$, is the expectation of the outcome when Alice runs the attack $\mathsf{Adv}_A^{(b)}$ against honest Bob (resp. Bob runs $\mathsf{Adv}_B^{(b)}$ against honest Alice).

Proof Sketch: Consider the stateless protocol π' guaranteed by Lemma 4. (This protocol is polynomial time given an $\mathbf{NP}$ oracle.) By Theorem 1, there are adversaries $\mathsf{Adv}_A^{(0)}$, $\mathsf{Adv}_B^{(0)}$, $\mathsf{Adv}_A^{(1)}$, $\mathsf{Adv}_B^{(1)}$, which attack π', and access π' as a black-box. Thus these adversaries can be implemented in polynomial time, given an $\mathbf{NP}$ oracle. By Lemma 4 (or rather, its proof) these adversaries have the same advantage with π as with π', and hence one of the four conditions stated in Theorem 1 hold with respect to π and these adversaries. Note that these are the same conditions described above, arranged differently. ■

Finally, note that if $\mathbf{NP} \subseteq \mathbf{BPP}$, then a highly accurate $\mathbf{NP}$ oracle can be implemented in probabilistic polynomial time, and hence the adversaries in the above theorem can be converted to PPT adversaries.

Corollary 4. *If π is a μ-secure polynomial time implementation of χ^*-weak coin for any constant $0 < \chi^* < 1$, then unless $\mathbf{NP} \not\subseteq \mathbf{BPP}$, $\mu \leq \frac{1}{2} + negl(D)$, where D is the number of rounds in π and negl is a negligible function.*

In particular, if there is a weak coin-flip protocol for a coin of bias $\frac{1}{2}$ which is μ-secure with $\mu > \frac{1}{2} + \alpha$ for some non-negligible function α (corresponding to limiting the bias approximately to the range $[\frac{1}{4}, \frac{3}{4}]$), then $\mathbf{NP} \not\subseteq \mathbf{BPP}$.

IV. Constant Round Weak Coin-Flipping

In this section we show a much stronger intractability implication of a weak coin-flipping protocol with a very weak unbiasability guarantee, if the protocol has only constantly many rounds. Note that we do allow the communication complexity of the protocol to be polynomial. We show the following result:

Theorem 5. *If infinitely-often one-way functions do not exist then for any constant round coin-tossing protocol π, there exist attacks $\mathsf{Adv}_A^{(0)}$, $\mathsf{Adv}_A^{(1)}$, $\mathsf{Adv}_B^{(0)}$ and $\mathsf{Adv}_B^{(1)}$, such that for any $\epsilon = 1/poly(k)$ ($0 < \epsilon < 1$), the attacks run in polynomial time in k, and for sufficiently large k:*

1) $\min\left\{1 - A^{(1)}, B^{(0)}\right\} \leq \epsilon$, and
2) $\min\left\{A^{(0)}, 1 - B^{(1)}\right\} \leq \epsilon$,

where $A^{(b)}$ (resp. $B^{(b)}$) for $b \in \{0,1\}$, is the expectation of the outcome when Alice runs the attack $\mathsf{Adv}_A^{(b)}$ against honest Bob (resp. Bob runs $\mathsf{Adv}_B^{(b)}$ against honest Alice).

In other words, if infinitely-often one-way functions do not exist then with probability close to 1 either Alice can bias the outcome to b or Bob can bias it to $(1-b)$ starting from any transcript prefix v.

The attack has the following intuitive form: use the fact that any polynomial time function can be inverted to implement next message function oracle for the protocol. At any point in the protocol, use this to sample a polynomial-sized sub-tree of the protocol (with the density of children

sampled at each node increasing with depth[8]), and run the **PSPACE** attack on this sampled tree to decide on the next move in the attack. While conceptually simple, this idea runs into two complications.

- At each round, the response from the honest party may not fall within the sub-tree that was sampled for the attack at that round; and as such the original attack computed may have no further relevance, and no use in deciding the response in subsequent rounds. Further, the **PSPACE** attack involves evaluating a max-average tree, and by sampling it is quite possible to miss the maximum.
- During the attack the distributions on the nodes at each level of the tree can deviate significantly from that under the honest execution, and the next message function oracles need to work well on these distributions. These distributions depend on the behavior of the attack in the previous rounds, which however carries out recursive look-aheads (in implementing the **PSPACE** attack), and these look-aheads in turn involve accessing the next message function oracles. A simple attempt at implementing the next message function oracles can lead to circularity.

Our attack is based on realizing an efficient algorithm I, called *inverter*, which can efficiently perform the following task: Given a partial transcript v, it outputs a k-bit message m such that $\Pr(m|v)$ is identical to the probability of π generating a transcript vm conditioned on the fact that v is generated as a partial transcript. Alternately, if we are able to sample uniformly at random from the set of randomness R_v of pairs (r_A, r_B) such that Alice and Bob with local randomness r_A and r_B generate the partial transcript v when running the protocol π, then we can implement I. We shall reverse sample from the set R_v and run the protocol for one more round and obtain a transcript prefix vm.

To address the first issue, we work in a hybrid world, where we assume that we are provided access to the inverter $I(\cdot)$. Next, we provide efficient algorithms to create an approximation of $I(\cdot)$.

Evaluating Performance of Attacks: The first issue mentioned above alludes to the problem of correctly evaluating the performance of Alice and Bob's attack on a subtree of the transcript tree. Instead of exactly evaluating Alice and Bob's performance, $A^{(b)}(v)$ and $B^{(1-b)}(v)$, we compute approximations to these quantities which are correct with high probability. Formally, for the case when Alice wants to bias the output to 1 and Bob wants to bias the output to 0, we define functions $\tilde{A}^{(1)}(v), \tilde{B}^{(0)}(v)$ such that, with probability $(1-\epsilon_h)$, the following conditions are satisfied:

1) $\left|A^{(1)}(v) - \tilde{A}^{(1)}(v)\right| \leq \epsilon_h$,
2) $\left|B^{(0)}(v) - \tilde{B}^{(0)}(v)\right| \leq \epsilon_h$, and
3) $\min\left\{1 - \tilde{A}^{(1)}(v), \tilde{B}^{(0)}(v)\right\} \leq \epsilon_h$,

where h is the height of v in the transcript tree, $\epsilon_{h+1} = k^{1/2}\epsilon_h^{1/6}$ and $\epsilon_1 = k^{3/5}\epsilon^{6^D}$. Figure 4 explicitly defines the algorithm to compute $\tilde{A}^{(0)}(v)$ for any node v in the transcript tree.

Implementing Inverters: The second issue can be taken care of by carefully defining a family of next message function oracles, which not only depend on what depth in the protocol it is sampling a message for, but also on which iteration in the **PSPACE** attack it appears in. Consider the attack performed by Alice. When she tries to attack a node v with height $i \in [D]$, she queries the inverter at nodes which are at height $i, i+1, \ldots, D$. Let $Q_{i,j}$, for $i \in [D]$ and $i \leq j \leq D$, be the set of nodes of height j which are queried to attack a node which has height i. We will construct an efficient inverter $\tilde{I}_{i,j}$ which works for Alice when she wants to query the nodes in $Q_{i,j}$.

Lemma 5. *If infinitely-often one-way functions do not exist, then, for sufficiently large k, there exists a class $\mathcal{I} = \{\tilde{I}_{i,j} | i \in [D] \text{ and } i \leq j \leq D\}$ of efficient inverters, such that if Alice uses $\tilde{I}_{i,j}$ to invert nodes at height j when she is attacking a node with height i then the behavior of $\mathsf{Adv}_A^{(1)}$ is at most $1/poly(k)$ different from the case when she uses the actual inverter I.*

Let $f(x) = y$ be a polynomial time function and $\mathcal{D}$ be the distribution of $f(x)$ when x is uniformly sampled. If one-way functions do not exist, then there exists an efficient algorithm A such that $f(A(y)) = y$ and $A(y)$ is $1/k^c$ close to the uniform distribution when y is sampled according to the distribution $\mathcal{D}$. Note that the guarantee is only for the distribution $\mathcal{D}$ and not for any arbitrary distribution.

So, we can not use this result to directly create an inverter. We need to additionally show that the distribution of queried nodes is drawn from a distribution which can be generated in polynomial time. The main observation required is that all queries in $Q_{i,j}$ could be performed simultaneously. In other words, they only depend on the nodes inverted while attacking previous rounds or higher nodes in the current round. More formally, define

$$\mathcal{Q}_{i,j}^{\text{pre}} = \bigcup_{\substack{i'>i, \text{ or} \\ i'=i \text{ and } j'>j}} Q_{i',j'}$$

So, we define the the execution of $\mathsf{Adv}_A^{(1)}$ just before it inverts $Q_{i,j}$ as the function f. Now, the inverter $\tilde{I}_{i,j} = A$ could be used to invert all partial transcripts in $Q_{i,j}$.

It is worth mentioning that the time complexity of $I_{i,j}$ is only guaranteed to be polynomial in the time complexity of all the inverters $\{I_{i',j'} | i' > i \text{ or } (i' = i \text{ and } j' > j)\}$. So, the time complexity of the inverter $I_{D,D}$ turns out to be $k^{\Theta(1)^D}$, which is polynomial if and only if D is constant. Therefore, this approach works only when D is a constant.

[8]The exact description of the tree is provided in Figure 4.

Subroutine to compute $\tilde{A}^{(1)}(v)$

Define $N_j = \epsilon_{j-1}^{-1/2}$ and $M_j = k^{1/3}\epsilon_{j-1}^{-1/3}$.

1) If v is a leaf, return $\tilde{A}^{(1)}(v) = \chi_v$.
2) If v is an Alice node at height j: Sample $\{u_1, \ldots, u_{N_j M_j}\}$ honest extensions of v by calling $I(v)$. Return

$$\tilde{A}^{(1)}(v) = \frac{\sum_{k=1}^{M_j}\left[\max_{k'=1}^{N_j}\tilde{A}^{(1)}\left(u_{(k-1)N_j+k'}\right)\right]}{M_j}$$

3) If v is a Bob node at height j: Sample $\{u_1, \ldots, u_{M_j}\}$ honest extensions of v by calling $I(v)$. Return

$$\tilde{A}^{(1)}(v) = \frac{\sum_{k=1}^{M_j}\tilde{A}^{(1)}(v)}{M_j}$$

Algorithm $\mathsf{Adv}_A^{(1)}$

Let v be an Alice node at height j.

1) Sample $\{u_1, \ldots, v_{N_j}\}$ honest extensions of v using $I(\cdot)$.
2) Output the k-bit message m such that $vm = \text{argmax}_{k\in[N_j]}\tilde{A}^{(1)}(u_k)$.

Figure 4. *Computation of $\tilde{A}^{(1)}(v)$ to help Alice bias towards outcome 1, assuming access to the inverter oracle $I(\cdot)$.*

ACKNOWLEDGEMENTS

We are grateful to Russell Impagliazzo for proposing this intriguing area of investigation, and a number of useful conversations. We also thank Boaz Barak and Yael Kalai for collaboration at an early stage of this research.

REFERENCES

[1] M. Blum, "Coin flipping by phone," in *Proc. 24th IEEE Computer Conference (CompCon)*, 1982, pp. 133–137, see also *SIGACT News*, Vol. 15, No. 1, 1983.

[2] R. Impagliazzo, "5 worlds of problems," Talk at the workshop Complexity and Cryptography: Status of Impagliazzo's Worlds, Princeton, NJ., 2009.

[3] R. Impagliazzo and M. Luby, "One-way functions are essential for complexity based cryptography (extended abstract)," in *Proc. 30th FOCS*. IEEE, 1989, pp. 230–235.

[4] O. Goldreich, S. Micali, and A. Wigderson, "How to play ANY mental game," in *Proc. 19th STOC*, ACM, Ed. ACM, 1987, pp. 218–229, see [17, Chap. 7] for more details.

[5] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge and its applications (extended abstract)," in *STOC*, 1988, pp. 103–112.

[6] I. Kerenidis and A. Nayak, "Weak coin flipping with small bias," *Inf. Process. Lett.*, vol. 89, no. 3, pp. 131–135, 2004.

[7] R. Impagliazzo, "Pseudo-random generators for cryptography and for randomized algorithms." Ph.D. dissertation, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 1992.

[8] R. Cleve and R. Impagliazzo, "Martingales, collective coin flipping and discrete control processes," 1993.

[9] O. Goldreich and L. A. Levin, "A hard-core predicate for all one-way functions," in *Proc. 21st STOC*. ACM, 1989, pp. 25–32.

[10] M. Naor, "Bit commitment using pseudo-randomness (extended abstract)," in *CRYPTO*, ser. Lecture Notes in Computer Science, G. Brassard, Ed., vol. 435. Springer, 1989, pp. 128–136.

[11] J. Hastad, R. Impagliazzo, L. A. Levin, and M. Luby, "A pseudorandom generator from any one-way function," vol. 28, no. 4, pp. 1364–1396, 1999, preliminary versions appeared in STOC' 89 and STOC' 90.

[12] S. Zachos, "Probabilistic quantifiers and games," *J. Comput. Syst. Sci.*, pp. 433–451, 1988.

[13] R. Ostrovsky and A. Wigderson, "One-way functions are essential for non-trivial zero-knowledge," International Computer Science Institute, Berkeley, CA, Tech. Rep. TR-93-073, Nov. 1993, preliminary version in Proc. 2nd Israeli Symp. on Theory of Computing and Systems, 1993, pp. 3–17.

[14] E. A. Hirsch and D. Itsykson, "An infinitely-often one-way function based on an average-case assumption," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 14, no. 117, 2007.

[15] M. Jerrum, L. G. Valiant, and V. V. Vazirani, "Random generation of combinatorial structures from a uniform distribution," *Theor. Comput. Sci.*, vol. 43, pp. 169–188, 1986.

[16] M. Bellare, O. Goldreich, and E. Petrank, "Uniform generation of NP-witnesses using an NP-oracle," *Information and Computation*, vol. 163, 2000.

[17] O. Goldreich, *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.