Improving the security of device-independent weak coin flipping

Atul Singh Arora[1], Jamie Sikora[2], and Thomas Van Himbeeck[3,4]

[1] *California Institute of Technology, USA*
[2] *Virginia Polytechnic Institute and State University, USA*
[3] *University of Toronto, Canada*
[4] *Institute of Quantum Computing, University of Waterloo, Canada*

May 11, 2021

1 Introduction

Coin-flipping is the two-party cryptographic primitive where two parties, henceforth called Alice and Bob, wish to flip a coin, but, to make things interesting, they do not trust each other. This primitive was introduced by Blum [Blu83] who also introduced the first (classical) protocol. In this work, we concentrate on *weak* coin flipping (WCF) protocols where Alice and Bob desire opposite outcomes. Since then, a series of quantum protocols were introduced which kept improving the security. Mochon finally settled the question about the limits of the security in the quantum regime by proving the existence of quantum protocols with security approaching the ideal limit [Moc07]. Mochon's work was based on the notion of point games, a concept introduced by Kitaev. Since then, a sequence of works have continued the study of point games. In particular, the proof has been simplified [ACG+14] and made explicit [ARW, ARW19, ARV] **(Jamie:** missing the date on last ref**)** . Interestingly, Miller [Mil20] used Mochon's proof to show that protocols approaching the ideal limit must have an exponentially increasing number of messages. **(Atul:** Maybe add applications of coin flipping to bit commitment etc**)** We note that all of this work is in the *device-dependent* setting where *Alice and Bob trust their quantum devices*. In this work, we *revise* the security definitions such that when Alice or Bob cheat, they have control of each other's quantum devices, opening up a plethora of new cheating strategies that were not considered in the previously mentioned references. **(Atul:** mention noise here, perhaps?**)** **(Jamie:** I don't see why. That's seems like a completely different thing.**)**

The prefix *weak* in weak coin flipping refers to the situation where Alice and Bob desire opposite outcomes of the coin. (We have occasion to discuss *strong* coin flipping protocols, where Alice or Bob could try to bias the coin towards either outcome, but it is not the focus of this work.) When designing weak coin flipping protocols, the security goals are as follows.

Completeness for honest parties:	If Alice and Bob are honest, then they share the same outcome of a protocol $c \in \{0,1\}$, and c is generated uniformly at random by the protocol. **(Atul:** Umm, perhaps correctness is a better term?**)** **(Jamie:** I like completeness, since it goes with soundess better :) But, it's not a strong preference**)**
Soundness against cheating Bob:	If Alice is honest, then a dishonest (i.e., cheating) Bob cannot force the outcome $c = 1$.
Soundness against cheating Alice:	If Bob is honest, then a dishonest (i.e., cheating) Alice cannot force the outcome $c = 0$.

The commonly adopted goal of two-party protocol design is to assume perfect completeness and then minimize the effects of a cheating party, i.e., to make it as sound as possible. This way, if no parties cheats, then the protocol at least does what it is meant to still. With this in mind, we need a means to quantify the effects of a cheating party. It is often convenient to have a single measure to determine if one protocol is better than another. For this purpose, we use *cheating probabilities* (denoted p_B^* and p_A^*) and *bias* (denoted ϵ), defined as follows. **(Atul:** Pedantic: Is there a reason why p_B^* comes first?**)** **(Jamie:** I have always put Bob first. Not sure why. Feel free to change if you want!**)**

p_B^*: The maximum probability with which a dishonest Bob can force an honest Alice to accept the outcome $c = 1$.

p_A^*: The maximum probability with which a dishonest Alice can force an honest Bob to accept the outcome $c = 0$.

ϵ: The maximum amount with which a dishonest party can bias the probability of the outcome away from uniform. Explicitly, $\epsilon = \max\{p_B^*, p_A^*\} - 1/2$.

These definitions are not complete in the sense that we have not yet specified what a cheating Alice or a cheating Bob are allowed to do, or of their capabilities. In this work, we study *information theoretic security* meaning that Alice and Bob are only bounded by the laws of quantum mechanics. For example, they are not bounded by polynomial-time quantum computations. In addition to this, we study the security in the *device-independent* regime where we assume Alice and Bob have complete control over the quantum devices when they decide to "cheat".

When studying device-independent (DI) protocols, one should first consider whether or not there are decent classical protocols (since these are not affected by the DI assumption). Indeed, Kitaev [Kit03] proved that any classical WCF protocol has bias $\epsilon = 1/2$, which is the worst possible value. Thus, it makes sense to study quantum WCF protocols in the DI setting, especially if one with bias $\epsilon < 1/2$ can be found. Indeed, Silman, Chailloux, Aharon, Kerenidis, Pironio, and Massar presented a protocol in [SCA+11] which has bias $\epsilon \approx 0.33664$.

In this work, we provide two techniques which can be applied to a wide range of protocols (including [SCA+11], mentioned above) which can improve the bias. To illustrate our ideas, we now present the protocol in [SCA+11].

Protocol 1 (DI-WCF protocol with $p_A^* = \cos^2 \pi/8$ and $p_B^* = 3/4$ [SCA+11]; see also Section **??**). *Alice has one box and Bob has two boxes. Each box takes one binary input and gives one binary output and are designed to play the optimal GHZ game strategy. (Who creates and distributes the boxes is not important in the DI setting.)*

1. *Alice chooses a uniformly random input to her box $x \in \{0, 1\}$ and obtains the outcome a. She chooses another uniformly random bit $r \in \{0, 1\}$ and computes $s = a \oplus (x \cdot r)$. She sends s to Bob.*

2. *Bob chooses a uniformly random bit $g \in \{0, 1\}$ and sends it to Alice. (We may think of g as Bob's "guess" for the value of x.)*

3. *Alice sends x and a to Bob. They both compute the output $c = x \oplus g$. This is the outcome of the protocol assuming neither Alice nor Bob abort.* ***(Jamie:*** *I guess only Bob may abort?)*

4. *Bob now tests to see if Alice was honest using the following two tests.*

   Test 1: *Bob sees if $s = a$ or $s = a \oplus x$. If this is not the case, he knows Alice cheated and aborts.*

   Test 2: *Bob chooses a uniformly random bit $y \in \{0, 1\}$ and computes $z = x \oplus y \oplus 1$. He inputs y and z into his two boxes and obtains respective outcomes b and c. He aborts if (a, b, c, x, y, z) does not satisfy the winning conditions of the GHZ game.*

5. *If Bob does not abort, they both accept the value of c as the outcome of the protocol.*

To obtain a bias $\epsilon \leq 0.33664$ using the protocol above, they compose the protocol many times.

Remark. Note that the above protocol is actually a strong coin-flipping protocol, but as such, we can always treat it as a WCF protocol.

In this work, we build on this protocol using novel pre- and post-processing steps, which we discuss next.

1.1 Our main result

We now state the main result of our work.

Theorem 2. *There exists device-independent weak coin flipping protocols with bias approaching $\epsilon \approx 0.3148$. Under a convergence assumption, the bias can be lowered to $\epsilon \approx 0.29104$.*

Author's note: *We believe the convergence assumption above to be true, we just did not have enough time to rigourously* ***(Jamie:*** *American or British English?)* *prove it before the QCRYPT submission deadline. Hence, we state it as an assumption. However, it does lead to a better bias, and thus we have decided to state it here.*

We now discuss the proof of our main theorem, above. The first step is that we turn the strong coin flipping protocol into a weak coin flipping protocol in a routine manner. Basically, since weak coin flipping has the notion of a "winner" (if $c = 0$ Alice wins and if $c = 1$ Bob wins) we have the person who does not win do the testing. We illustrate this new protocol, below.

Protocol 3 (Weak version of Protocol 1)**.** *Alice has one box and Bob has two boxes. Each box takes one binary input and gives one binary output and are designed to play the optimal GHZ game strategy. (Who creates and distributes the boxes is not important in the DI setting.)*

1. *Alice chooses a uniformly random input to her box $x \in \{0,1\}$ and obtains the outcome a. She chooses another uniformly random bit $r \in \{0,1\}$ and computes $s = a \oplus (x \cdot r)$. She sends s to Bob.*
2. *Bob chooses a uniformly random bit $g \in \{0,1\}$ and sends it to Alice. (We may think of g as Bob's "guess" for the value of x.)*
3. *Alice sends x and a to Bob. They both compute the output $c = x \oplus g$. This is the outcome of the protocol assuming neither Alice nor Bob abort.*
4. ***(Jamie:** Add Alice test.)*
5. ***(Jamie:** Add Bob test.)*
6. *If Alice and Bob do not abort, they both accept the value of c as the outcome of the protocol.*

We now append to this protocol with a pre-processing step which *self-tests* the boxes Alice and Bob share. This concept is not new, but it applies well to this setting.

We also use the revised protocol in a new way of composing protocols which we subbed an *abort-phobic* composition.

1.2 First technique: Self-testing

(Tom: I changed the name of the section so that the structure of the paper can be read from the titles) **(Jamie:** Noice.)

In Protocols 1 and 3, a cheating party may control what measurement is performed in the boxes of other party and how the state of the boxes is correlated to its own quantum memory. This is more general then *device-dependent* protocols, where for instance, the measurements are known by the honest player. However, we employ the concept of self-testing to stop Bob *or Alice* from applying such a strategy. The case where Alice self-tests Bob is illustrated below.

Protocol 4 (Protocol with Alice self-testing)**.** *Alice starts with n boxes, indexed from 1_1 to 1_n. Bob starts with $2n$ boxes, the first half indexed by 2_1 to 2_n and the last half indexed by 3_1 to 3_n. The triple of boxes $(1_i, 2_i, 3_i)$ is meant to play the optimal GHZ game strategy.*

1. *Alice selects a uniformly random index $i \in \{1, \ldots, n\}$ and asks Bob to send her all the boxes* except *those indexed by 2_i and 3_i.*
2. *Alice plays $n-1$ GHZ games using the $n-1$ triples of boxes she has, making sure she has a space-like separation between the boxes. (She has long arms.)*
3. *Alice aborts if* any *of the GHZ games lose. Otherwise, she announces to Bob that they can use the remaining boxes for Protocol 3.* ***(Jamie:** I am referring to Protocol 3 now, as we don't use the SCF protocol (protocol 1).)*

The idea is that if n is chosen large enough, then this forces a dishonest Bob to not tamper with the boxes too much. Indeed, this step already allows us to reduce the cheating probabilities.

Lemma 5 (Informal. See Lemma **??** for a formal statement)**.** *When Alice self-tests Bob, the cheating probabilities of Protocol 4* ***(Jamie:** Protocol 3???)* *in the limit of large n are*

$$p_A^* = \cos^2(\pi/8) \approx 0.85355 \quad \text{and} \quad p_B^* \approx 0.6667. \tag{1}$$

Recall that for Protocol 1 [SCA+11], $p_A^* = \cos^2(\pi/8)$ and $p_B^* = 3/4$. To prove this lemma, we have to dive into two technical concepts, which we briefly discuss below. Note that for device-dependent protocols, where Alice and Bob trust their devices, cheating probabilities can be cast as semidefinite programs (see [Kit03] for details. In the DI regime, we cannot even bound the dimension of the state within the boxes, making it much harder to analyze and bound Alice and Bob's cheating probabilities.

(Tom: I think that the main idea that we use is the fact that for device-dependent protocols the maximum bias can be recast as an sdp. Me have not mentioned that up to now. Also it's not correct o say that we have prove rigidity**)**
(Jamie: Tom, why is it that we can't say we haven't proved rigidity? In any case, happy to have the continuity stuff reworded...**)**

Self-testing the GHZ game. We prove that Alice self-tests Bob and passes all $n-1$ plays of the GHZ game, then the remaining triple of boxes has to be approximately performing the optimal GHZ strategy. The differences between this approximation and the optimal strategy disappear in the limit of large n. See Section **??** for details.

Continuity of semidefinite programs. When Alice self-tests Bob, we are able to formulate Bob's cheating probability as a semidefinite program in the limit of large n. However, we cannot have a protocol with an infinite number of messages. Consequently, we study a family of protocols where Bob's cheating probabilities approach certain thresholds. Therefore, we need the semidefinite program values to capture the behaviour of the cheating probabilities as they approach the limit of large n. See Section **??** for details.

Remark. Both of these technical steps may find use in independent applications. In particular, the continuity of semidefinite programs section is written for general semidefinite programs for the most part.

1.3 Second technique: abort-phobic composition

It can happen, that for a given WCF protocol, $p_B^* \neq p_A^*$, in which case we say the protocol is polarised **(Jamie:** American or British English?**)** . It is known (e.g. [SCA+11]) that composing a polarized protocol with itself (or other protocols) can effectively reduce the bias. Our second improvement is a modified way of composing protocols, when there is a positive probability that the honest player catches the cheating player. Let us start by recalling the standard way of composing protocols.

Standard composition. For a protocol with cheating probabilities p_B^* and p_A^*, we say that it has polarity towards Alice when it satisfies $p_A^* > p_B^*$. Similarly, we say that it has polarity towards Bob when $p_B^* > p_A^*$. Given a polarized protocol $\mathcal{P}$, we may switch the roles of Alice and Bob since the definition of coin-flipping is symmetric. To make the polarity explicit, we define $\mathcal{P}_A$ to be the version of the protocol with $p_A^* > p_B^*$ and $\mathcal{P}_B$ to be the version with $p_B^* > p_A^*$. With this in mind, we can now define a simple composition.

Protocol 6 (Winner-gets-polarity composition)**.** *Alice and Bob agree on a protocol $\mathcal{P}$.*

1. *Alice and Bob perform protocol $\mathcal{P}$.*
2. *If Alice wins, she polarizes the second protocol towards herself, i.e., they now use the protocol $\mathcal{P}_A$ to determine the outcome of the (entire) protocol.*
3. *If Bob wins, he polarizes the second protocol towards himself, i.e., they now use the protocol $\mathcal{P}_B$ to determine the outcome of the (entire) protocol.*

The standard composition above is a decent way to balance the cheating probabilities of a protocol. For instance, if $\mathcal{P}$ has cheating probabilities p_A^* and p_B^* with $p_A^* > p_B^*$, then the composition gets to decide "who gets to be Alice" in the second run. We can easily compute Alice's cheating probability in the composition as

$$(p_A^*)^2 + (1-p_A^*)p_B^* < p_A^* \tag{2}$$

and Bob's as

$$p_B^* p_A^* + (1-p_B^*)p_B^* < p_A^*. \tag{3}$$

This does indeed reduce the bias since the maximum cheating probability is now smaller.

Abort-phobic composition. The "traditional" way of considering WCF protocols is to view them as only having two outcomes "Alice wins" (when $c = 0$) or "Bob wins" ($c = 1$). This is because Alice can declare herself the winner if she catches Bob cheating. Similarly, Bob can declare himself the winner if he catches Alice cheating. This is completely fine when we consider "one-shot" versions of these protocols, but we lose something when we compose them. For instance, in the simple composition used in Protocol 6, Bob should not really accept to continue onto the second protocol if he catches Alice cheating in the first. That is, if he knows Alice cheated, he can declare himself the winner of the entire protocol! In other words, the cheating probabilities (2) and (3) may get reduced even further. For purposes of this discussion, suppose Bob adopts a cheating strategy which has a probability v_B of him winning ($c = 1$), a probability v_A of him losing ($c = 0$), and a probability $v_\perp$ of Alice catching him cheating. Then his cheating probability in the (abort-phobic) version of the simple composition is now

$$v_B \cdot p_A^* + v_A \cdot p_B^* + \gamma \cdot 0. \tag{4}$$

This quantity may be a strict improvement if $\gamma > 0$ when $v_B = p_B^*$.

The concept of abort-phobic composition is simple. Alice and Bob keep using WCF protocols and the winner (at that round) gets to choose the polarity of the subsequent protocol. However, if either party *ever aborts*, then it is game over and the cheating player loses *the entire composition protocol*.

One may think it is tricky to analyze abort-phobic compositions, but we may do this one step at time. To this end, we introduce the concept of *cheat vectors*.

Definition 7 (Alice and Bob's cheat vectors)**.** Given a protocol $\mathcal{I}$, we say that $(v_A, v_B, v_\perp)$ is a cheat vector for (dishonest) Bob if there exists a cheating strategy where:

- v_B is the probability with which Alice accepts the outcome $c = 1$,
- v_A is the probability with which Alice accepts the outcome $c = 0$,
- $v_\perp$ is the probability with which Alice aborts.

We denote the set of cheat vectors for (dishonest) Bob by $\mathbb{C}_B(\mathcal{I})$. Cheat vectors for (dishonest) Alice and $\mathbb{C}_A(\mathcal{I})$ are analogously defined keeping the notation v_A for her winning, v_B for her losing, and $v_\perp$ for Bob aborting.

(Jamie: rewrote what is below to make T-note happier.**)**

In this work, we show how to capture cheat vectors as the feasible region of a semidefinite program, from which we can optimize

$$v_B \cdot p_A^* + v_A \cdot p_B^* + v_\perp \cdot 0. \tag{5}$$

For this to work, we assume we have p_A^* and p_B^* for the protocol that comes in the second round. The neat thing is that once we solve for the optimal cheating probabilities in the abort-phobic composition in this way, we can then fix those probabilities and compose again! In other words, we are recursively composing the abort-phobic composition. Therefore, we calculate the cheating probabilities from the *bottom-up*.

By using protocols where Alice self-tests and abort-phobic compositions, we are able to find protocols which converge onto a bias of $\epsilon \approx 0.3148$ proving the main result of this work. We give more details below.

1.4 Putting all the pieces together to improve the bias of weak coin flipping

Let $\mathcal{I}$ denote Protocol 1 **(Jamie:** we now also have protocol 3. Which one are you using?**)** (the DI protocol introduced in [SCA+11]). We use $p_A^*(\mathcal{I})$ and $p_B^*(\mathcal{I})$, to denote the cheating probabilities of Alice and Bob when they execute protocol $\mathcal{I}$. Similarly, we use $\mathbb{C}_A(\mathcal{I})$ and $\mathbb{C}_B(\mathcal{I})$ to denote the cheat vectors for Alice and Bob, respectively, when they execute $\mathcal{I}$.

Protocols

(Tom: WHY LL notation ?**)** We introduce two variants of protocol $\mathcal{I}$, which we call $\mathcal{P}$ and $\mathcal{Q}$.

- $\mathcal{P}$ is essentially the same as $\mathcal{I}$ except that Alice self-tests her boxes before starting the protocol and performs an additional test to ensure Bob does not cheat **(Jamie:** refer to protocol 3 now**)** . We show that $p_A^*(\mathcal{P}) \approx 0.853\ldots$ and $p_B^*(\mathcal{P}) \approx 0.667\ldots$ **(Jamie:** Is this not a previous lemma in this intro?**)** . We also show that the set of cheat vectors $\mathbb{C}_B(\mathcal{P})$ can be cast as an SDP.

- $\mathcal{Q}$ is also essentially the same as $\mathcal{I}$ except that Bob self-tests his boxes before starting the protocol. In this case, $p_A^*(\mathcal{Q}) = p_A^*(\mathcal{I})$ and $p_B^*(\mathcal{Q}) = p_B^*(\mathcal{I})$ so the advantage is not immediate. However, now $\mathbb{C}_A(\mathcal{Q})$ can be cast as a semidefinite program which, as we shall see, yields an advantage when $\mathcal{Q}$ is composed. *In other words, Alice's cheat vectors make it so that the protocols compose nicely.* **(Jamie:** Atul, what do you think? Is this ok?**)**

Compositions

Anticipating the notation used later, we denote by $C^{LL}(\mathcal{X})$, the repeated "standard composition" introduced above, of some protocol $\mathcal{X}$. Further, Let $\epsilon(\mathcal{X})$ denote the bias of $\mathcal{X}$.

- The bias of protocol $\mathcal{I}$ under the standard composition is given by

  $$\epsilon(C^{LL}(\mathcal{I})) \approx 0.33664$$

  while for our improved protocol $\mathcal{P}$, it is given by

  $$\epsilon(C^{LL}(\mathcal{P})) \approx 0.3199. \tag{6}$$

The standard composition does not yield any improvement for $\mathcal{Q}$ because the cheating probabilities are identical to those of $\mathcal{I}$. We can extract an advantage by using a composition technique that uses the cheat vectors, i.e., the abort-phobic composition. We denote the abort phobic composition (of protocol $\mathcal{X}$) by either $C^{\perp L}(\mathcal{X})$ or $C^{L\perp}(\mathcal{X})$. The notation is explained in later sections when the choice of superscripts is made clear.

- Composing the protocol $\mathcal{P}$ with itself many times, using the abort-phobic compositions gives a bias

  $$\epsilon(C^{\perp L}(\mathcal{P})) \approx 0.3148$$

  which is a further improvement.

- Composing the protocol $\mathcal{Q}$ with itself many times, using the abort-phobic composition gives a bias

  $$\epsilon(C^{L\perp}(\mathcal{Q})) \approx 0.3226$$

  which is worse than Equation (6) **(Jamie:** This seems like it shouldn't be true...**)** .

- However, when we compose the protocol Q **(Jamie:** Replace n with "many"?**)** n times with itself, followed by protocol P, we find

  $$\epsilon(C^{L\perp}(\mathcal{Q}, \mathcal{Q}, \ldots, \mathcal{Q}, \mathcal{P})) \approx 0.29104\ldots$$

where we use the same composition technique except that at the last "level" we use[1] $\mathcal{P}$ instead of $\mathcal{Q}$.

(Jamie: this notation needs to be explained since it's not clear what is above and what is in the footnote.**)** **(Jamie:** Also, I am not sure why we mention the rest explicitly. Is it for build-up or knowledge? Is there a reason we just don't say the protocol composition that works best?**)**

1.5 Applications

The concept of polarity extends well beyond finding WCF protocols and, as such, the "winner-gets-polarity" concept allows for WCF to be used in many other compositions. Indeed, we can use it to balance the cheating probabilities in *any* polarized protocol for any symmetric two-party cryptographic task for which such notions can be properly defined.

For instance, many *strong* coin-flipping protocols can be thought of as polarized. For an example, the protocol **??** is indeed a strong coin-flipping protocol. Thus, by balancing the cheating probabilities of that protocol using our DI WCF protocol and a winner-gets-polarity composition (not even an abort-phobic one!), we get the following theorem.

Theorem 8. *There exists DI strong coin-flipping protocols where no party can cheat with probability greater than* ?????.

(Atul: Using winner-gets-polarity, $\epsilon \leq 0.334904$, combining abort phobic, $\epsilon \leq 0.33439$, and combining Alice and Bob self testing (under a convergence assumption), $\epsilon \leq 0.33192$. To contrast, for [SCA+11], $\epsilon \leq 0.336637$.**)** There are likely more examples of protocols which can be balanced in a DI way using this idea.

[1] $C^{\perp L}(\mathcal{P}, \mathcal{P}, \ldots, \mathcal{P}, \mathcal{Q})$ is strictly worse than considering $C^{\perp L}(\mathcal{P}, \mathcal{P}, \ldots, \mathcal{P}, \mathcal{P})$; this should become evident later.

References

[ACG+14] Dorit Aharonov, André Chailloux, Maor Ganz, Iordanis Kerenidis, and Loïck Magnin, *A simpler proof of existence of quantum weak coin flipping with arbitrarily small bias*, SIAM Journal on Computing **45** (2014), no. 3, 633–679.

[ARV] Atul Singh Arora, Jérémie Roland, and Chrysoula Vlachou, *Analytic quantum weak coin flipping protocols with arbitrarily small bias*, pp. 919–938.

[ARW] Atul Singh Arora, Jérémie Roland, and Stephan Weis, *Quantum weak coin flipping*.

[ARW19] Atul Singh Arora, Jérémie Roland, and Stephan Weis, *Quantum weak coin flipping*, Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing - STOC 2019, ACM Press, 2019.

[Blu83] Manuel Blum, *Coin flipping by telephone a protocol for solving impossible problems*, SIGACT News **15** (1983), no. 1, 23–27.

[Kit03] Alexei Kitaev, *Quantum coin flipping*, Talk at the 6th workshop on Quantum Information Processing, 2003.

[Mil20] Carl A. Miller, *The impossibility of efficient quantum weak coin flipping*, Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (New York, NY, USA), STOC 2020, Association for Computing Machinery, 2020, pp. 916–929.

[Moc07] Carlos Mochon, *Quantum weak coin flipping with arbitrarily small bias*, arXiv:0711.4114 (2007).

[SCA+11] J. Silman, A. Chailloux, N. Aharon, I. Kerenidis, S. Pironio, and S. Massar, *Fully distrustful quantum bit commitment and coin flipping*, Physical Review Letters **106** (2011), no. 22.