Unfair Coin Tossing

Grégory Demay Ueli Maurer
Department of Computer Science,
ETH Zürich, Switzerland
Email: {demayg,maurer}@inf.ethz.ch

***Abstract*—An ideal coin tossing resource for two parties outputs the same random bit to both parties. We introduce the notion of an unfair coin tossing resource by relaxing both the fairness and the non-influenceability guarantees that an ideal coin toss would provide. The presence of this non-ideal behavior is necessary in order to understand what coin tossing protocols really achieve in the setting of two distrustful parties, since it is known that such an ideal coin tossing resource cannot be constructed whenever a majority of players is dishonest.**

I. Introduction

A. Two Distrustful Parties and Coin Tossing

A two-party ideal coin tossing resource delivers the same uniform random bit to both parties. They are in particular guaranteed that the bit they received is *fair*, the first party receives a bit if and only if the second party receives the same bit, and *not influenced*, the bit a party receives is uniformly random no matter what the other party does.

Ideal resources (such as an ideal coin toss) are generally not available as a system in the physical world, but they must be constructed by cryptographic means using available resources (such as a communication channel) or assuming other resources (such as bit commitment). Constructive cryptography introduced in [1] (see also [2]), is a new approach to cryptography in which cryptographic protocols are seen as constructions of resources from other resources. The definition of the term *construction* depends on the setting, i.e., on who can potentially be dishonest and whether the security should be information-theoretic or computational. A composition theorem of constructive cryptography guarantees that constructive steps compose.

The goal of coin tossing protocols is of course to construct such an ideal coin tossing resource. However, since [3], it is known that such an attempt is doomed to fail in the setting of two distrustful parties (more generally any setting where a majority of players is dishonest). In this paper, we ask the following simple question. Since coin tossing protocols cannot construct an ideal coin tossing resource, what do they actually construct?

B. Related Work

Blum in [4] gave the first coin tossing protocol under the assumption that one-way functions exist. Later, Cleve showed in [3] that any coin tossing protocol with a majority of dishonest players would have a bias of $\Omega(\frac{1}{r})$, where r is the number of rounds of the protocol, because a malicious party could, by prematurely aborting the protocol, bias the output of the honest party. This issue has usually been tackled in two different manners. Either by restricting the security notion in allowing the honest parties not to output anything in case of a premature abortion of the protocol [5]. Or, if one still wants some security guarantees in case of abortion, to relax the metric, i.e., the distinguishing advantage, between the ideal coin tossing functionality and what the protocol achieves. The latter approach was initiated by [6] and lead to the notion of an optimally fair coin toss [7], [8].

C. Contributions and Outline

Unfortunately, security proofs in the second approach are not composable because of the huge relaxation of the metric ($1/p(r)$ for some fixed polynomial p instead of negligible) which might be critical for basic functionalities such as coin tossing. Instead, we propose a very natural approach which consists of making explicit the exact "ideal" resource constructed by the aforementioned coin tossing protocols. Of course, the constructed "ideal" resource will contain some form of non-ideal behavior (fairness and non-influenceability will have to be relaxed) and will be weaker than an ideal coin toss. The goal is not to introduce weaker resources per se, but simply to be able to state exactly what a protocol achieves.

Our results use the concept of secure construction defined in [1], [2], which will be briefly restated in Section II-B. For the sake of clarity, we introduce our unfair coin tossing resource in two steps. A first simplified version is presented in Section III and we show that Blum's protocol [4] can be seen as the construction of such a resource. Then, we introduce the complete unfair coin tossing resource in Section IV and show how having many of such resources can help in constructing a less unfair resource by using Cleve's majority protocol argument [3].

II. Preliminaries

A. Notation

We denote sets by calligraphic letters or capital greek letters (e.g., $\mathcal{X},\ \Sigma$). Throughout this paper, we consider only discrete random variables. A discrete random variable will be denoted by an upper-case letter X, its range by the corresponding calligraphic letter $\mathcal{X}$, and a realization of the random variable X will be denoted by the corresponding lower-case letter x. Unless stated otherwise, $X \in_R \mathcal{X}$ denotes a random variable X selected independently and uniformly at random in $\mathcal{X}$. A tuple of n random variables $(X_1, \ldots, X_n)$ will be denoted by X^n. For a binary vector $X^n \in \{0,1\}^n$, $w_H(X^n)$ will denote the

978-1-4799-0446-4/13/$31.00 ©2013 IEEE

Hamming weight of X^n, i.e., the number of ones in X^n. The probability distribution of a random variable X will be denoted as P_X. For two probability distributions P_X and Q_X, their *statistical distance* is denoted by $\mathsf{d}\left(\mathsf{P}_X, \mathsf{Q}_X\right)$ and is defined as follows: $\mathsf{d}\left(\mathsf{P}_X, \mathsf{Q}_X\right) := \frac{1}{2}\sum_{x\in\mathcal{X}}\left|\mathsf{P}_X\left(x\right) - \mathsf{Q}_X\left(x\right)\right|$. For a binary random variable X, let $\mathsf{b}\left(X\right) \in [-1, 1]$ denote the *bias* of the bit X, where

$$\mathsf{b}\left(X\right) := 1 - 2 \cdot \mathsf{P}_X\left(1\right).$$

Note that a bias of -1 (respectively, $+1$) corresponds to the constant bit 1 (respectively, 0).

B. Constructive Cryptography

We use the concept of abstract systems [1], [2] to formulate our results, and partly follow the concise exposition of [9]. At the highest level of abstraction, a system is an object with interfaces via which it interacts with other systems. Every two systems can be composed by connecting one interface of each system, and the resulting object is again a system. Also, we assume that every two different systems are mutually independent.

We consider three distinct types of systems: *resources*, *converters*, and *distinguishers*. Resources are denoted by upper-case boldface letters such as $\mathbf{R}$ and $\mathbf{S}$. In this paper, we always consider resources with two interfaces, the left interface will be referred to as Alice's, while the right interface will be referred to as Bob's. In our scenario, either Alice or Bob could be dishonest, and the case where both are dishonest does not need to be considered. Resources $\mathbf{R}$ and $\mathbf{S}$ could also be used in parallel, and the resulting resource, denoted $\mathbf{R}||\mathbf{S}$, is again a 2-interface resource, where each of the interfaces of $\mathbf{R}||\mathbf{S}$ allows access to the corresponding interface of both subsystems $\mathbf{R}$ and $\mathbf{S}$.

Converters are systems with one *inside* and one *outside* interface, and are denoted by lower-case Greek letters, such as α, β, σ. The set of all converters will be denoted by Σ. A converter α can be attached to a resource $\mathbf{R}$ by connecting the inside interface of α to one of the two interfaces of $\mathbf{R}$. For example, if α is attached to the left interface of $\mathbf{R}$, then the resulting resource is denoted by $\alpha\mathbf{R}$, and is again a 2-interface system whose left interface is now the outside interface of α. Similarly, attaching the converter β to the right interface of $\mathbf{R}$ is denoted by $\mathbf{R}\beta$.

A distinguisher $\mathbf{D}$ is a system that connects to all interfaces of a resource $\mathbf{R}$ and outputs at a separate interface a single bit denoted B. The complete interaction between $\mathbf{D}$ and $\mathbf{R}$ defines a random experiment, and the probability that $\mathbf{D}$ outputs 1 in this random experiment is denoted by $\mathsf{P}^{\mathbf{DR}}(B = 1)$. The *distinguishing advantage* of $\mathbf{D}$ in distinguishing the system $\mathbf{R}$ from $\mathbf{S}$ is defined as $\Delta^{\mathbf{D}}(\mathbf{R}, \mathbf{S}) := \left|\mathsf{P}^{\mathbf{DR}}(B = 1) - \mathsf{P}^{\mathbf{DS}}(B = 1)\right|$. The set of all distinguishers is denoted by $\mathcal{D}$, and we define $\Delta^{\mathcal{D}}(\mathbf{R}, \mathbf{S}) := \sup_{\mathbf{D}\in\mathcal{D}} \Delta^{\mathbf{D}}(\mathbf{R}, \mathbf{S})$. Note that $\Delta^{\mathcal{D}}$ defines a pseudo-metric, and for convenience we will use the following notation, $\mathbf{R} \approx_\varepsilon \mathbf{S} :\Leftrightarrow \Delta^{\mathcal{D}}(\mathbf{R}, \mathbf{S}) \leq \varepsilon$, where $\varepsilon \in [0, 1]$. The next simple lemma shows that deterministic distinguishers are optimal.

Lemma 1. *Consider two arbitrary resources* $\mathbf{R}$ *and* $\mathbf{S}$. *Let* $\mathbf{D}_1, \ldots, \mathbf{D}_n$ *be* n *distinguishers, and define the distinguisher* $\mathbf{D}$ *to be the distinguisher* $\mathbf{D}_i$ *with probability* $\mathsf{P}_I(i)$, *where* I *is an independent random variable over* $\{1, \ldots, n\}$. *Then,*

$$\Delta^{\mathbf{D}}\left(\mathbf{R}, \mathbf{S}\right) \leq \max_{i\in\{1,\ldots,n\}} \Delta^{\mathbf{D}_i}\left(\mathbf{R}, \mathbf{S}\right).$$

A protocol, in our case a pair of converters, is used to construct a specific ideal resource from available real resources, where the meaning of "construct" is now made precise.

Definition 1. A two-party protocol $\pi = (\alpha, \beta) \in \Sigma^2$, where only one party could be dishonest, securely constructs a resource $\mathbf{S}$ from a resource $\mathbf{R}$ within ε, denoted $\mathbf{R} \xrightarrow{(\pi,\varepsilon)} \mathbf{S}$, if and only if

$$\begin{aligned} \alpha\mathbf{R}\beta &\approx_\varepsilon \mathbf{S}, \\ \exists\sigma \in \Sigma : \alpha\mathbf{R} &\approx_\varepsilon \mathbf{S}\sigma, \\ \exists\tau \in \Sigma : \mathbf{R}\beta &\approx_\varepsilon \tau\mathbf{S}. \end{aligned}$$

The converter σ in Definition 1 acts as a *simulator*, i.e., it guarantees that what a malicious Bob could do in the real world (when connected to $\alpha\mathbf{R}$), he could also do it in the ideal world (when connected to $\mathbf{S}\sigma$). The role of the converter τ is symmetric.

Note that as a specific instantiation of abstract cryptography, [1, Th. 2] ensures us that our security definition is *generally composable*. That is, a protocol which is secure according to Definition 1 will remain secure under arbitrary sequential or parallel composition.

C. Auxiliary Resources

Throughout this paper, we will use only two auxiliary resources. The first one is a perfect communication channel from Bob to Alice, denoted $\leftarrow$. The second one is an ideal bit commitment functionality denoted by $\mathbf{COM}$. The resource for bit commitment is represented in Figure 1 and can be informally described as follows:

1) on input $x \in \{0, 1\}$ at the left interface, output "committed" at the right interface;
2) on input "open" at the left interface, output x at the right interface.

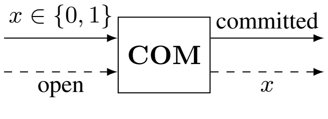

Figure 1. Ideal Bit Commitment Resource.

III. Single Unfair Coin Tossing Resource

We are now ready to describe our (simplified) unfair coin tossing resource. To do so, let $a \in [0, 1]$ be a non-negative bias. The resource for unfair coin tossing, denoted by $\mathbf{UCT}^a$, is represented in Figure 2, and can be described as follows,

1) output a uniform bit $C \in_R \{0, 1\}$ at the left interface,

2) on input a bit $b \in \{0,1\}$ at the left interface, output the bit C' at the right interface, where C' can be viewed as a random function (of C and b) and is defined in (1).

$$C' := \begin{cases} C & \text{if } b = 0, \\ C \oplus N & \text{otherwise,} \end{cases} \tag{1}$$

where N is an independent binary random variable such that $\mathsf{P}_N(1) = a$.

Note that the resource $\mathbf{UCT}^a$ is doubly unfair in the sense that not only Alice sees what the value of the coin toss could be before Bob, but depending on this value she could also try to flip the bit output to him. Based on the definition of C' in (1), it is readily verified that for a uniform bit C and an arbitrary strategy to choose the flipping bit b, we have $|\mathsf{b}(C')| \leq a$. Thus, a is a measure of how biased Bob's output can be, and Bob is guaranteed that the bit he received from $\mathbf{UCT}^a$ has a bias in the interval $[-a, a]$.

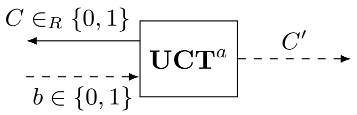

Figure 2. (Simplified) Unfair Coin Tossing Resource.

A. Filtered Resources

Consider the simple converter Ψ showed in Figure 3 and which can be described as follows. On input $c \in \{0,1\}$ at the inside interface, Ψ outputs 0 at the inside interface and outputs c at the outside interface. Note that the system $\Psi\mathbf{UCT}^a$ corresponds exactly to an ideal coin tossing resource, where the same uniform random bit is output to both interfaces. The converter Ψ can be seen as a *filter* restricting Alice's access to the resource $\mathbf{UCT}^a$. The corresponding filtered resource, denoted $\mathbf{UCT}^a_\Psi$, models the fact that Alice is guaranteed to have access to $\Psi\mathbf{UCT}^a$, but by behaving maliciously she could potentially have a direct access to $\mathbf{UCT}^a$ and then be able to bias Bob's output for example.

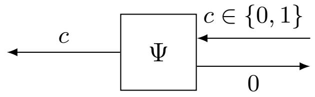

Figure 3. Filter for the Unfair Coin Tossing Resource.

More generally, we could consider a filter for each interface of a resource $\mathbf{R}$, leading to the filtered resource $\mathbf{R}_\phi$, where $\phi = (\phi_l, \phi_r) \in \Sigma^2$ and similarly for a filtered resource $\mathbf{S}_\psi$, $\psi = (\psi_l, \psi_r) \in \Sigma^2$. The construction notion stated in Definition 1 generalizes to filtered resources as follows.

Definition 2. A two-party protocol $\pi = (\alpha, \beta) \in \Sigma^2$, where only one party could be dishonest, securely constructs a filtered resource $\mathbf{S}_\psi$ from a filtered resource $\mathbf{R}_\phi$ within ε, denoted $\mathbf{R}_\phi \xrightarrow{(\pi,\varepsilon)} \mathbf{S}_\psi$, if and only if

$$\begin{aligned} & \alpha\phi_l\mathbf{R}\phi_r\beta \approx_\varepsilon \psi_l\mathbf{S}\psi_r, \\ \exists\sigma \in \Sigma : \ & \alpha\phi_l\mathbf{R} \quad\ \ \approx_\varepsilon \psi_l\mathbf{S}\sigma, \\ \exists\tau \in \Sigma : \ & \quad\ \mathbf{R}\phi_r\beta \approx_\varepsilon \tau\mathbf{S}\psi_r. \end{aligned}$$

B. Blum's protocol

To justify the definition of our unfair coin tossing resource, we show that Blum's coin tossing protocol [4] constructs such a resource. In the following, let $\pi_{\mathcal{B}} = (\alpha_{\mathcal{B}}, \beta_{\mathcal{B}}) \in \Sigma^2$ be a pair of converters corresponding to Blum's coin tossing protocol [4]. That is, assuming that we have at our disposal a commitment resource $\mathbf{COM}$ (described in Figure 1) and a perfect communication channel $\leftarrow$ from Bob to Alice, the protocol $(\alpha_{\mathcal{B}}, \beta_{\mathcal{B}})$ can be informally described as follows. The converter $\beta_{\mathcal{B}}$ waits for $\alpha_{\mathcal{B}}$ to commit to a random bit X before sending its random bit Y. Once $\alpha_{\mathcal{B}}$ received the bit Y from the communication channel, it opens its commitment revealing X to $\beta_{\mathcal{B}}$. Then, both converters $\alpha_{\mathcal{B}}$ and $\beta_{\mathcal{B}}$ output $X \oplus Y$. In case of a party aborting the protocol prematurely, the other party outputs a uniform random bit. An honest execution of Blum's protocol is shown in Figure 4.

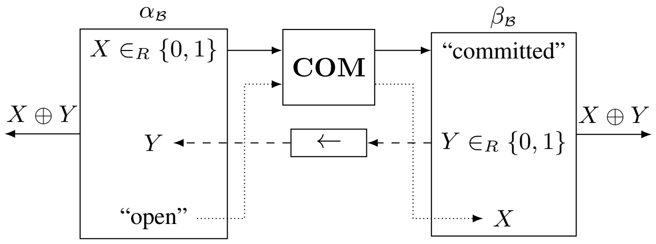

Figure 4. Blum's Coin Tossing Protocol.

Theorem 1. *Blum's protocol perfectly constructs a filtered $\frac{1}{2}$-biased unfair coin tossing resource, i.e.,*

$$(\mathbf{COM} \parallel \leftarrow) \xrightarrow{(\pi_{\mathcal{B}},0)} \mathbf{UCT}^{\frac{1}{2}}_\Psi.$$

Proof: Recalling Definition 2, we have to show

$$\alpha_{\mathcal{B}}(\mathbf{COM} \parallel \leftarrow)\beta_{\mathcal{B}} \approx_0 \Psi\mathbf{UCT}^{\frac{1}{2}}, \tag{2}$$

$$\exists\sigma_{\mathcal{B}} \in \Sigma : \alpha_{\mathcal{B}}(\mathbf{COM} \parallel \leftarrow) \quad \approx_0 \Psi\mathbf{UCT}^{\frac{1}{2}}\sigma_{\mathcal{B}}, \tag{3}$$

$$\exists\tau_{\mathcal{B}} \in \Sigma : \quad (\mathbf{COM} \parallel \leftarrow)\beta_{\mathcal{B}} \approx_0 \tau_{\mathcal{B}}\mathbf{UCT}^{\frac{1}{2}}. \tag{4}$$

Note that (2) trivially holds, and (3) is readily verified by considering the following simulator $\sigma_{\mathcal{B}}$:

1) on input $C \in \{0,1\}$ at the inside interface, output "committed" at the outside interface,
2) on input $y \in \{0,1\}$ at the outside interface, output $C \oplus y$ at the outside interface.

In order to prove (4), consider the simulator $\tau_{\mathcal{B}}$ described in System 1. To simplify the notation, we shall denote by $\mathbf{T}$ the system $(\mathbf{COM} \parallel \leftarrow)\beta_{\mathcal{B}}$ and by $\mathbf{U}$ the system $\tau_{\mathcal{B}}\mathbf{UCT}^{\frac{1}{2}}$ in the remaining of the proof. Consider a distinguisher $\mathbf{D}$ trying to distinguish $\mathbf{T}$ from $\mathbf{U}$. Note that by Lemma 1, and given the limited interaction that $\mathbf{D}$ can have when connected

System 1 Simulator $\tau_{\mathcal{B}}$ for Blum's protocol

```
**Input:** $C \in \{0,1\}$ at the inside interface
**Input:** $X \in \{0,1\}$ at the outside interface
    output $\hat{Y} := X \oplus C$ at the outside interface
    **on input**: $t$ at the outside interface
        $b \leftarrow 0$
        **if** $t \neq$ "open" **then**                    // Try to flip $C$
            $b \leftarrow 1$
        output $b$ at the inside interface.
    **end on input**
```

to the left interface of either $\mathbf{T}$ or $\mathbf{U}$, it is sufficient to consider a distinguisher which sends a random value to the commitment resource and then does *not* open its commitment. When connected to $\mathbf{T}$, the distinguisher $\mathbf{D}$ sees the random variables X, Y, Z, where Z denotes the random bit output by $\beta_{\mathcal{B}}$ at the outside interface. By the specification of $\beta_{\mathcal{B}}$, those three random variables X, Y, Z are independent and uniformly distributed. When connected to $\mathbf{U}$, the distinguisher $\mathbf{D}$ sees the random variables $X, \hat{Y}, C'$. Firstly, note that $\hat{Y} = X \oplus C$ is an independent uniform bit when conditioned on X. Secondly, by definition of $\tau_{\mathcal{B}}$ and (1), we have in this random experiment $C' = C \oplus N$, where N is an independent uniform random variable. Thus, the statistical distance between (X, Y, Z) and $(X, \hat{Y}, C')$ is 0 and so is the distinguishing advantage of the distinguisher $\mathbf{D}$. ■

Note that Blum's coin tossing protocol cannot construct a less biased coin tossing resource $\mathbf{UCT}^{a}$, where $a < \frac{1}{2}$. To see this consider (4) and the system $(\mathbf{COM} \| \leftarrow) \beta_{\mathcal{B}}$. A malicious Alice could choose to never open her commitment when the output of the protocol would be 0, which would result in the probability of Bob outputting 1 to be $\frac{3}{4}$, corresponding to a bias of $-\frac{1}{2}$.

IV. Multiple Unfair Coin Tossing Resources

Assume that we have at our disposal n unfair coin tossing resources. Can we construct a less unfair coin tossing resource, where the bias a malicious Alice could inflict to Bob's output would be smaller? In order to answer positively to this question, we need to handle two subtleties in the definition of our unfair coin tossing resource which we voluntarily left out so far for the sake of simplicity. Namely, Blum's coin tossing protocol actually constructs a stronger resource (for Bob) not in terms of bias, but in the following sense:

1) by choosing when to send his random bit Y to Alice (via the communication channel $\leftarrow$), Bob decides when Alice knows the outcome of the protocol, i.e., the coin toss;
2) Bob knows when Alice aborted the protocol (assuming that the systems considered are synchronous), hence knows when Alice tried to "flip" his output.

Consequently, we consider from now onwards the following (complete) unfair coin tossing resource $\mathbf{CT}^{a}$, represented in Figure 5, and which can be described as follows,

1) on input "toss" at the right interface, output a uniform random bit $C \in_R \{0,1\}$ at the left interface,
2) on input a bias $b \in [-a, a]$ at the left interface, output $(b', C') \in \{0,1\}^2$ at the right interface, where the bit b' is 1 if and only if $b \neq 0$, and C' is defined in (5).

$$C' := \begin{cases} C & \text{if } (C = 0 \wedge b \geq 0) \vee (C = 1 \wedge b \leq 0), \\ C \oplus N & \text{otherwise,} \end{cases} \tag{5}$$

where N is an independent binary random variable such that $\mathsf{P}_N(1) = |b|$. The reason why we define C' differently than in (1) is to be able to inflict a smaller bias than the maximum to Bob's output, while ensuring that b' will be 1. Looking ahead, this will be needed for the simulator τ_m in the proof of Theorem 3.

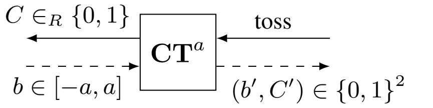

Figure 5. (Complete) Unfair Coin Tossing Resource.

The following theorem is a trivial generalization of Theorem 1, where the protocol considered is a slight modification of Blum's protocol in order to take into account the new input/output behavior of $\mathbf{CT}^{a}$. We omit the proof.

Theorem 2. *A modification of Blum's protocol perfectly constructs the $\frac{1}{2}$-biased unfair coin tossing resource depicted in Figure 5, i.e.,*

$$\exists \pi \in \Sigma^2 : \; (\mathbf{COM} \; \| \; \leftarrow) \xrightarrow{(\pi,0)} \mathbf{CT}_{\Psi}^{\frac{1}{2}}.$$

The next theorem shows that having n unfair coin tossing resources is useful to construct a less unfair resource in the sense that the bias a malicious Alice could inflict to Bob's output can be reduced by a factor which is equivalent to $\frac{2}{\sqrt{2\pi n}}$.

Theorem 3. *There exists a two party protocol $\pi \in \Sigma^2$ such that for an odd number n of unfair coin tossing resources,*

$$\mathbf{CT}_{\Psi}^{a_1} \| \cdots \| \mathbf{CT}_{\Psi}^{a_n} \xrightarrow{(\pi,0)} \mathbf{CT}_{\Psi}^{a},$$

where $a := 2^{-(n-1)} \binom{n-1}{\frac{n-1}{2}} \max_{j \in \{1,\dots,n\}} a_j.$

Proof (sketch): The general idea of the protocol is to ensure that Alice can flip at most one bit out of the n bits that Bob would normally receive. To do so, consider the protocol $\pi_m = (\alpha_m, \beta_m) \in \Sigma^2$ together with the system $\alpha_m (\Psi\mathbf{CT}^{a_1} \| \cdots \| \Psi\mathbf{CT}^{a_n}) \beta_m$, and let maj denote the boolean majority function[1]. After having first received the message "toss" at its outside interface, the converter β_m acts as follows. During round i, $i \in \{1, \dots, n\}$, β_m outputs the message "toss" at its inside interface to $\mathbf{CT}^{a_i}$ in order to release the coin toss C_i to Alice. Upon receiving the pair

[1] $\mathrm{maj}(X^n) = 1 \Leftrightarrow \sum_{i=1}^{n} X_i \geq \lceil \frac{n}{2} \rceil, \; X^n \in \{0,1\}^n$.

System 2 Simulator τ_m for the majority protocol π_m

```
Input: $C \in \{0,1\}$ at the inside interface
  select $\hat{C}^n \in_R \{0,1\}^n$ until $\mathrm{maj}(\hat{C}^n) = C$
  for $i = 1$ to $n$ do
      output $\hat{C}_i$ at the outside interface
      on input: $b_i \in [-a_i, a_i]$ at the outside interface
          if $b_i \neq 0$ then
              if $(\hat{C}_i = 0 \wedge b \geq 0) \vee (\hat{C}_i = 1 \wedge b \leq 0)$ then
                  $b \leftarrow -a$ if $C = 1$; $b \leftarrow a$ if $C = 0$
              else
                  $b \leftarrow 2^{-(n-1)}\binom{n-1}{(n-1)/2} b_i$
              output $b$ at the inside interface
              halt
      end on input
  output $0$ at the inside interface
```

$(b'_i, C'_i) \in \{0,1\}^2$ from $\mathbf{CT}^{a_i}$, β_m proceeds to the next round if and only if $b'_i = 0$. In case $b'_i = 1$, indicating that a malicious Alice did try to flip the i^{th} coin toss, β_m does *not* reveal the remaining coin tosses from $\mathbf{CT}^{a_{i+1}}, \dots, \mathbf{CT}^{a_n}$ to Alice and instead selects the remaining bits $C'_{i+1}, \dots, C'_n$ independently and uniformly at random. In both cases, β_m finally outputs the pair (b', V) at the outside interface, where b' indicates whether Alice cheated (formally it is the logical OR of the b'_i's that $\beta_m$received) and V corresponds to the majority of $C'_1, \dots, C'_n$. Similarly, α_m outputs the majority of C^n and in case a malicious Bob would not have revealed the i^{th} coin toss (assuming the systems considered are synchronous), selects the remaining bits $C_i, \dots, C_n$ independently and uniformly at random. Recalling Definition 2, we have to show the existence of two simulators $(\sigma_m, \tau_m) \in \Sigma^2$ such that [2]

$$\alpha_m(\Psi\mathbf{CT}^{a_1}\| \cdots \|\Psi\mathbf{CT}^{a_n})\,\beta_m \approx_0 \Psi\mathbf{CT}^{a}, \tag{6}$$

$$\alpha_m(\Psi\mathbf{CT}^{a_1}\| \cdots \|\Psi\mathbf{CT}^{a_n}) \approx_0 \Psi\mathbf{CT}^{a}\sigma_m, \tag{7}$$

$$(\mathbf{CT}^{a_1}\| \cdots \|\mathbf{CT}^{a_n})\,\beta_m \approx_0 \tau_m\mathbf{CT}^{a}. \tag{8}$$

Note that (6) trivially holds because both converters α_m and β_m apply the same deterministic function, the majority rule, and the latter preserves the uniform distribution.

Equation (7) is verified by considering the following simulator σ_m. The simulator σ_m has to emulate n outside sub-interfaces corresponding each to the right interface of $\Psi\mathbf{CT}^{a_1}, \dots, \Psi\mathbf{CT}^{a_n}$, respectively. It transmits the first "toss" message received at its outside interface to $\mathbf{CT}^a$ and receives $(0, C)$ at its inside interface. It then keeps selecting n independent uniform bits $\hat{C}^n \in_R \{0,1\}^n$ until their majority corresponds to C. Finally, it outputs $(0, \hat{C}_i)$ to the outside i^{th} sub-interface where the "toss" message was input.

In order to prove (8), consider the simulator τ_m described in System 2. Similarly to σ_m, the converter τ_m first finds uniformly at random n bits $\hat{C}_1, \dots, \hat{C}_n$ such that their majority corresponds to the bit C given by the ideal resource $\mathbf{CT}^a$ when the message "toss" was input. Then, τ_m sequentially releases each $\hat{C}_i$ after having previously received a 0-bias input by Alice. In case the bias input was not 0, the simulator τ_m will bias Bob's output accordingly. In order to keep shorter notations, we will denote the system $(\mathbf{CT}^{a_1}\| \cdots \|\mathbf{CT}^{a_n})$ by $\mathbf{V}$ during the remaining of the proof. By definition of β_m, a malicious Alice could bias at most 1 bit of $\mathbf{V}$. Hence, by Lemma 1 it is sufficient to consider the following distinguisher. Let $j \in \{1, \dots, n\}$ and consider the distinguisher $\mathbf{D}_j$ which at the left interface of $\mathbf{V}\beta_m$ acts honestly up to the $j-1$ step (always input a 0 bias), and then tries to influence $\mathbf{CT}^{a_j}$ towards 1 by inputing $b_j = -a_j$ to $\mathbf{CT}^{a_j}$ if C_j was 0. Recall that by definition of $\mathbf{CT}^{a_i}$, all the bits $C'_1, \dots, C'_n$ output to β_m are independently and uniformly distributed, except naturally for the random bit C'_j which takes on the value 1 with probability $\frac{1+a_j}{2}$. Then, the probability that β_m outputs 1 in this random experiment, denoted $\mathsf{P}^{\mathbf{D}_j\mathbf{V}\beta_m}(V = 1)$, is

$$\begin{aligned}
&\mathsf{P}^{\mathbf{D}_j\mathbf{V}\beta_m}(V = 1) \\
&= \sum_{\substack{c^n \in \{0,1\}^n, c_j = 0, \\ w_H(c^n) \geq \lceil \frac{n}{2} \rceil}} \mathsf{P}^{\mathbf{D}_j\mathbf{V}}_{C'^n}(c^n) + \sum_{\substack{c^n \in \{0,1\}^n, c_j = 1, \\ w_H(c^n) \geq \lceil \frac{n}{2} \rceil}} \mathsf{P}^{\mathbf{D}_j\mathbf{V}}_{C'^n}(c^n) \\
&= 2^{-(n-1)} \left(\sum_{k=\lceil \frac{n}{2} \rceil}^{n} \binom{n-1}{k} \frac{1-a_j}{2} + \binom{n-1}{k-1} \frac{1+a_j}{2} \right) \\
&= \frac{1}{2} + 2^{-n} \binom{n-1}{\frac{n-1}{2}} a_j.
\end{aligned}$$

Thus, the value of the bias output by τ_m guarantees that the statistical distance between the random variables involved in both random experiments, $\mathbf{D}_j\mathbf{V}\beta_m$ and $\mathbf{D}_j\tau_m\mathbf{CT}^a$, is 0. ■

Acknowledgments

Our research was supported by the Zurich Information Security and Privacy Center (ZISC) and the Swiss National Science Foundation (SNF), project no. 200020-132794.

References

[1] U. Maurer and R. Renner, "Abstract cryptography," in *ICS 2011*. Tsinghua University Press, Jan. 2011, pp. 1–21.

[2] U. Maurer, "Constructive cryptography – a new paradigm for security definitions and proofs," in *TOSCA 2011*, vol. 6993. Springer-Verlag, Apr. 2011, pp. 33–56.

[3] R. Cleve, "Limits on the security of coin flips when half the processors are faulty," in *STOC 1986*. ACM, 1986, pp. 364–369.

[4] M. Blum, "Coin flipping by telephone a protocol for solving impossible problems," *SIGACT News*, vol. 15, pp. 23–27, Jan. 1983.

[5] O. Goldreich, *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, 2004.

[6] J. Katz, "On achieving the "best of both worlds" in secure multiparty computation," in *STOC 2007*. ACM, 2007, pp. 11–20.

[7] T. Moran, M. Naor, and G. Segev, "An optimally fair coin toss," in *TCC 2009*. Springer Berlin / Heidelberg, 2009, vol. 5444, pp. 1–18.

[8] A. Beimel, E. Omri, and I. Orlov, "Protocols for multiparty coin toss with dishonest majority," in *CRYPTO 2010*. Springer Berlin / Heidelberg, 2010, vol. 6223, pp. 538–557.

[9] U. Maurer, A. Rüedlinger, and B. Tackmann, "Confidentiality and integrity: A constructive perspective," in *TCC 2012*, vol. 7194. Springer, 2012, pp. 209–229.

[2] For synchrony reasons we would need to modify the filter Ψ to output the bit C at its outside interface only at a specified time and to add a filter at Bob's interface whose sole purpose would also be to delay the output of $\mathbf{CT}^a$ at the right interface. We omit such technicalities here.