Two-Prover Bit-Commitments: Classical, Quantum and Non-Signaling

Fillinger, M.J.

Chapter 2

Preliminaries

2.1 Probabilities

2.1.1 Basic Notation

A (finite) *probability distribution* is a function $p : \mathcal{X} \to [0,1]$, $x \mapsto p(x)$, where $\mathcal{X}$ is a finite non-empty set such that $\sum_{x\in\mathcal{X}} p(x) = 1$. For $x_\circ \in \mathcal{X}$, we write $p(x = x_\circ)$ instead of $p(x_\circ)$. For any subset $\Lambda \subset \mathcal{X}$, called an *event*, the probability $p(\Lambda)$ (or $p(x \in \Lambda)$) is naturally defined as $p(\Lambda) = \sum_{x\in\Lambda} p(x)$, and it holds that

$$p(\Lambda) + p(\Gamma) = p(\Lambda \cup \Gamma) + p(\Lambda \cap \Gamma) \leq 1 + p(\Lambda \cap \Gamma) \tag{2.1}$$

for all $\Lambda, \Gamma \subset \mathcal{X}$, and, more generally, that

$$\sum_{i=1}^{k} p(\Lambda_i) \leq p(\Lambda_1 \cup \ldots \cup \Lambda_k) + \sum_{i<j} p(\Lambda_i \cap \Lambda_j) \leq 1 + \sum_{i<j} p(\Lambda_i \cap \Lambda_j) \tag{2.2}$$

for all $\Lambda_1, \ldots, \Lambda_k \subset \mathcal{X}$. For a distribution $p : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ on two variables and a relation R on $\mathcal{X} \times \mathcal{Y}$ (e.g., $x = y$, $x = f(y)$, $x \neq y$) the probability $p(R(x,y))$ is defined by

$$p(R(x,y)) = p\big(\{(x,y) \in \mathcal{X} \times \mathcal{Y} \,|\, R(x,y)\}\big) = \sum_{\substack{x\in\mathcal{X}, y\in\mathcal{Y} \\ \text{s.t. } R(x,y)}} p(x,y)\ .$$

The *marginals* $p(x)$ and $p(y)$ are given by $p(x) = \sum_y p(x,y)$ and $p(y) = \sum_x p(x,y)$, respectively. Vice versa, given two distributions $p(x)$ and $p(y)$, we say that a distribution $p(x,y)$ on two variables is a *consistent joint distribution* if the two marginals of $p(x,y)$ coincide with $p(x)$ and $p(y)$, respectively.

Definition 2.1. *Let $p(x)$ and $p(y)$ be two distributions over a common set $\mathcal{X}$. The* statistical distance *of the two distributions is defined as*

$$d(p(x), p(y)) = \frac{1}{2} \sum_{x_\circ \in \mathcal{X}} |p(x = x_\circ) - p(y = x_\circ)|$$

2.1.2 Some Basic Technical Results

We will make use of the following lemma regarding the existence of a consistent joint distribution that maximizes the probability that $x = y$.

Lemma 2.2. *Let $p(x)$ and $p(y)$ be two distributions on a common set $\mathcal{X}$. Then there exists a consistent joint distribution $p(x, y)$ such that $p(x = y = x_\circ) = \min\{p(x{=}x_\circ), p(y{=}x_\circ)\}$ for all choices of $x_\circ \in \mathcal{X}$. Additionally, $p(x, y)$ satisfies $p(x, y|x \neq y) = p(x|x \neq y) \cdot p(y|x \neq y)$.*

Proof. We first extend the respective probability spaces given by the distributions $p(x)$ and $p(y)$ by introducing an event Δ and declaring that

$$p(x{=}x_\circ \wedge \Delta) = \min\{p(x = x_\circ), p(y = x_\circ)\} = p(y{=}x_\circ \wedge \Delta)$$

for every $x_\circ \in \mathcal{X}$. Note that $p(\Delta)$ is well defined (by summing over all $x_\circ$). As we will see below, Δ will become the event $x = y$. In order to find a consistent joint distribution $p(x, y)$, it suffices to find a consistent joint distribution $p(x, y|\Delta)$ for $p(x|\Delta)$ and $p(y|\Delta)$, and a consistent joint distribution $p(x, y|\neg\Delta)$ for $p(x|\neg\Delta)$ and $p(y|\neg\Delta)$. The former, we choose as

$$p(x = x_\circ \wedge y = x_\circ|\Delta) := \min\{p(x = x_\circ), p(y = x_\circ)\}/p(\Delta)$$

for all $x_\circ \in \mathcal{X}$, and $p(x = x_\circ \wedge y = y_\circ|\Delta) := 0$ for all $x_\circ \neq y_\circ \in \mathcal{X}$, and the latter we choose as

$$p(x = x_\circ \wedge y = y_\circ|\neg\Delta) := p(x = x_\circ|\neg\Delta) \cdot p(y = y_\circ|\neg\Delta)$$

for all $x_\circ, y_\circ \in \mathcal{X}$. It is straightforward to verify that these are indeed *consistent* joint distributions, as required, so that $p(x, y) = p(x, y|\Delta) \cdot p(\Delta) + p(x, y|\neg\Delta) \cdot p(\neg\Delta)$ is also consistent. Furthermore, note that $p(x{=}y|\Delta) = 1$ and $p(x{=}y|\neg\Delta) = 0$; the latter holds because we have $p(x{=}x_\circ \wedge \Delta) = p(x = x_\circ)$ or $p(y{=}x_\circ \wedge \Delta) = p(y = x_\circ)$ for each $x_\circ \in \mathcal{X}$, and thus $p(x{=}x_\circ \wedge \neg\Delta) = 0$ or $p(y = x_\circ \wedge \neg\Delta) = 0$. As such, Δ is the event $x = y$, and therefore $p(x = y = x_\circ) = p(x = x_\circ \wedge \Delta) = \min\{p(x = x_\circ), p(y = x_\circ)\}$ for every $x_\circ \in \mathcal{X}$ as required. Finally, the claim regarding $p(x, y|x \neq y)$ holds by construction. □

The following property of the statistical distance is well known (see e.g. [RK05]) and can easily be proved using Lemma 2.2.

Corollary 2.3. *Let $p(x)$ and $p(y)$ be two distributions over the same set $\mathcal{X}$ with statistical distance $d\big(p(x),p(y)\big)=\varepsilon$. Then, there exists a consistent joint distribution $p(x,y)$ over $\mathcal{X}\times\mathcal{X}$ such that $p(x\neq y)=\varepsilon$.*

Proof. We apply Lemma 2.2 to obtain a consistent joint distribution $p(x,y)$ such that for all $x_\circ\in\mathcal{X}$, $p(x=y=x_\circ)=\min\{p(x=x_\circ),p(y=x_\circ)\}$. We then have

$$\begin{aligned}
p(x\neq y) &= 1-\sum_{x_\circ\in\mathcal{X}} p(x=y=x_\circ)\\
&= 1-\sum_{x_\circ\in\mathcal{X}} \min\{p(x=x_\circ),p(y=x_\circ)\}\\
&= \frac{1}{2}\sum_{x_\circ\in\mathcal{X}} p(x=x_\circ)-\min\{p(x=x_\circ),p(y=x_\circ)\}\\
&\quad+\frac{1}{2}\sum_{x_\circ\in\mathcal{X}} p(y=x_\circ)-\min\{p(x=x_\circ),p(y=x_\circ)\}\\
&= \frac{1}{2}\sum_{x_\circ\in\mathcal{X}} |p(x=x_\circ)-p(y=x_\circ)|\\
&= \varepsilon
\end{aligned}$$

as claimed. □

The following is an immediate consequence.

Lemma 2.4. *Let $p(x_0,y_0)$ and $p(x_1,y_1)$ be distributions with $d\big(p(x_0),p(x_1)\big)=\varepsilon$. Then, there exists a consistent joint distribution $p(x_0,x_1,y_0,y_1)$ such that $p'(x_0\neq x_1)=\varepsilon$ and, as a consequence, $d\big(p(x_0,y_1),p(x_1,y_1)\big)\leq\varepsilon$.*

Proof. We first apply Corollary 2.3 to $p(x_0)$ and $p(x_1)$ to obtain a consistent joint distribution $p(x_0,x_1)$, and then we set

$$p(x_0,x_1,y_0,y_1)=p(x_0,x_1)\cdot p(y_0|x_0)\cdot p(y_1|x_1)\,.$$

It is easy to see that this distribution is consistent with $p(x_0,y_0)$ and $p(x_1,y_1)$. For the last claim, we note that

$$\begin{aligned}
p(x_0,y_1) &= p(x_0=x_1)\cdot p(x_0,y_1|x_0=x_1)+p(x_0\neq x_1)\cdot p(x_0,y_1|x_0\neq x_1)\\
&= p(x_0=x_1)\cdot p(x_1,y_1|x_0=x_1)+p(x_0\neq x_1)\cdot p(x_0,y_1|x_0\neq x_1)
\end{aligned}$$

and

$$p(x_1,y_1)=p(x_0=x_1)\cdot p(x_1,y_1|x_0=x_1)+p(x_0\neq x_1)\cdot p'(x_1,y_1|x_0\neq x_1)$$

and the claim follows because $p(x_0\neq x_1)=\varepsilon$. □

When applying the above lemma, we say that we "glue together" the distributions $p(x_0, y_0)$ and $p(x_1, y_1)$ along x_0 and x_1.

Remark 2.5. *In the special case where $p(x_0)$ and $p(x_1)$ are distributed identically, we obviously have $p(x_0, y_1) = p(x_1, y_1)$.*

Remark 2.6. *It is easy to see from the proof of Lemma 2.4 that the following natural property holds. If $p(x_0, x_1, y_0, y_1, y_0', y_1')$ is obtained by gluing together $p(x_0, y_0, y_0')$ and $p(x_1, y_1, y_1')$ along x_0 and x_1, then the marginal $p(x_0, x_1, y_0, y_1)$ coincides with the distribution obtained by gluing together the marginals $p(x_0, y_0)$ and $p(x_1, y_1)$ along x_0 and x_1.*

Let $p(x, y, z)$ be a distribution over $\mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$ and let $\Lambda \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$. We then write $x \to y \to z$ to express that $p(x)$ and $p(z)$ are independent when conditioned on y, i.e., $p(x, z|y) = p(x|y)p(z|y)$. Similarly, we write $x \to \Lambda \to y$ to express that $p(x, y|\Lambda) = p(x|\Lambda)p(y|\Lambda)$, etc. We show the following property for conditionally independent variables.

Lemma 2.7. *If $x \to y \to z$ and $x \to x \neq y \to y$, then $x \to x \neq y \to z$.*

Proof. We assume that $x \to y \to z$ and $x \to x \neq y \to y$. We first observe that

$$\begin{aligned} p(x, x \neq y, z|y) &= p(x, x \neq y|y)\, p(z|x, y, x \neq y) \\ &= p(x, x \neq y|y)\, p(z|x, y) \\ &= p(x, x \neq y|y)\, p(z|y)\,, \end{aligned}$$

which means that $(x, x \neq y) \to y \to z$, and, by summing over x, implies $x \neq y \to y \to z$. It follows that

$$p(z|x, y, x \neq y) = p(z|y) = p(z|y, x \neq y)\,,$$

which actually means that $x \to (y, x \neq y) \to z$. Therefore,

$$\begin{aligned} p(x, z|x \neq y) = \sum_y p(x, y, z|x \neq y) &= \sum_y p(x, y|x \neq y)\, p(z|x, y, x \neq y) \\ &= p(x|x \neq y) \sum_y p(y|x \neq y)\, p(z|y, x \neq y) \\ &= p(x|x \neq y) \sum_y p(y, z|x \neq y) \\ &= p(x|x \neq y)\, p(z|x \neq y)\,, \end{aligned}$$

which was to be proven. □

2.2 Two-Prover Commitment Schemes

2.2.1 Protocols

In this work, we will consider 3-party interactive *protocols*, where the parties are named P, Q and V (the two "provers" and the "verifier"). Such a protocol prot_{PQV} consists of a triple $(\mathsf{prot}_P, \mathsf{prot}_Q, \mathsf{prot}_V)$ of L-round *interactive algorithms* for some $L \in \mathbb{N}$. Each interactive algorithm takes an input, and for every round $\ell \leq L$ computes the messages to be sent to the other algorithms/parties in that round as deterministic functions of its input, the messages received in the previous rounds, and the local randomness. In the same way, the algorithms produce their respective outputs after the last round. We write

$$(out_P \| out_Q \| out_V) \leftarrow \big(\mathsf{prot}_P(in_P) \| \mathsf{prot}_Q(in_Q) \| \mathsf{prot}_V(in_V)\big)$$

to denote the execution of the protocol prot_{PQV} on the respective inputs in_P, in_Q and in_V, and that the respective outputs out_P, out_Q and out_V are produced. Clearly, for any protocol prot_{PQV} and any input in_P, in_Q, in_V, the probability distribution $p(out_P, out_Q, out_V)$ of the output is naturally well defined.

If we want to make the local randomness explicit, we write $\mathsf{prot}_P[\xi_P](in_P)$ etc., and understand that ξ_P is correctly sampled. Furthermore, we write $\mathsf{prot}_P[\xi_{PQ}](in_P)$ and $\mathsf{prot}_Q[\xi_{PQ}](in_Q)$ to express that prot_P and prot_Q use *the same* randomness, in which case we speak of *joint randomness*.

We can *compose* two interactive algorithms prot_P and prot'_P in the obvious way, by applying prot'_P to the output of prot_P. The resulting interactive algorithm is denoted as $\mathsf{prot}'_P \circ \mathsf{prot}_P$. Composing the respective algorithms of two protocols $\mathsf{prot}_{PQV} = (\mathsf{prot}_P, \mathsf{prot}_Q, \mathsf{prot}_V)$ and $\mathsf{prot}'_{PQV} = (\mathsf{prot}'_P, \mathsf{prot}'_Q, \mathsf{prot}'_V)$ results in the composed protocol $\mathsf{prot}'_{PQV} \circ \mathsf{prot}_{PQV}$. If prot_P is a *non-interactive* algorithm, then $\mathsf{prot}'_{PQV} \circ \mathsf{prot}_P$ is naturally understood as the protocol $\mathsf{prot}'_{PQV} \circ \mathsf{prot}_P = (\mathsf{prot}'_P \circ \mathsf{prot}_P, \mathsf{prot}'_Q, \mathsf{prot}'_V)$, and similarly $\mathsf{prot}'_{PQV} \circ \mathsf{prot}_{QV}$ in case prot_{QV} is a protocol among Q and V only.

2.2.2 Defining Commitment Schemes

We model a two-prover commitment scheme as two protocols, one for the commit phase, one for the opening phase. In the commit phase, the input for the provers is an element s of the scheme's domain D which is the value they want to commit to. The verifier has no input. All three participants may output some state information. The protocol for the opening phase then takes this state information as input, and the verifier outputs an element of $D \cup \{\perp\}$. An output of $s \in D$ indicates that the provers successfully opened their commitment to s while an output of $\perp$ indicates that they failed to open their commitment.

Dishonest parties may execute arbitrary protocols. Dishonest provers do not have an input in the commit phase: intuitively speaking, they do not know which value they eventually want to open to at this time.

Definition 2.8. *A* 2-prover (string) commitment scheme $\mathcal{S}$ *with domain* D *consists of a pair of interactive protocols* $\mathsf{com}_{PQV} = (\mathsf{com}_P, \mathsf{com}_Q, \mathsf{com}_V)$ *and* $\mathsf{open}_{PQV} = (\mathsf{open}_P, \mathsf{open}_Q, \mathsf{open}_V)$ *between the provers* P *and* Q *and the verifier* V*, with the following syntactics. The* commit protocol com_{PQV} *uses joint randomness* ξ_{PQ} *for* P *and* Q *and takes an element* $s \in D$ *as input for* P *and* Q *(and independent randomness and no input for* V*), and it outputs a* commitment *com to* V *and some state information to* P *and* Q*:*

$$(state_P \| state_Q \| c) \leftarrow \left(\mathsf{com}_P[\xi_{PQ}](s) \| \mathsf{com}_Q[\xi_{PQ}](s) \| \mathsf{com}_V(\emptyset)\right).$$

The opening protocol open_{PQV} *uses joint randomness* η_{PQ} *and outputs a string or a rejection symbol to* V*, and nothing to* P *and* Q*:*

$$(\emptyset \| \emptyset \| s) \leftarrow \left(\mathsf{open}_P[\eta_{PQ}](state_P) \| \mathsf{open}_Q[\eta_{PQ}](state_Q) \| \mathsf{open}_V(c)\right)$$

with $s \in \{0,1\}^n \cup \{\perp\}$*. If* $D = \{0,1\}$*, we refer to* $\mathcal{S}$ *as a* bit*-commitment scheme instead, and we tend to use* b *rather than* s *to denote the committed bit.*

Remark 2.9. *Note that we still speak of string-commitment schemes even if the domain* D *does not consist of bit-strings.*

Remark 2.10. *By convention, we assume throughout the paper that the commitment* c *output by* V *equals the* communication *that takes place between* V *and the provers during the commit phase. This is without loss of generality since, in general,* c *is computed as a (possibly randomized) function of the communication, which* V *just as well can apply in the opening phase.*

Remark 2.11. *Note that we specify that* P *and* Q *use* fresh *joint randomness* η_{PQ} *in the opening phase, and, if necessary, the randomness* ξ_{PQ} *from the commit phase can be "handed over" to the opening phase via* $state_P$ *and* $state_Q$*; this will be convenient later on.*

Whenever we refer to such a 2-prover commitment scheme, we take it as understood that the scheme is complete and hiding, as defined below, for "small" values of γ and δ. Since our focus will be on the binding property, we typically do not make the parameters γ and δ explicit.

Definition 2.12. *A 2-prover commitment scheme is* γ-complete *if in an honest execution* V*'s output* s *of* open_{PQV} *equals* P *and* Q*'s input* s *to* com_{PQV} *except with probability* η*, for any choice of* P *and* Q*'s input* $s \in D$*.*

The standard definition for the hiding property is as follows:

Definition 2.13. *A 2-prover commitment scheme is* δ-hiding *if for any commit strategy* $\overline{\mathsf{com}}_V$ *and any two strings* s_0 *and* s_1*, the distribution of the commitments* c_0*,* c_1*, produced as*

$$(state_P \| state_Q \| c_b) \leftarrow (\mathsf{com}_P[\xi_{PQ}](s_b) \| \mathsf{com}_Q[\xi_{PQ}](s_b) \| \overline{\mathsf{com}}_V(\emptyset)), b = 0, 1$$

have statistical distance at most δ*. A* 0*-hiding scheme is also called* perfectly hiding.

Defining the binding property is more subtle. First, note that an attack against the binding property consists of a possible commit strategy $\overline{\mathsf{com}}_{PQ} = (\overline{\mathsf{com}}_P, \overline{\mathsf{com}}_Q)$ and a possible opening strategy $\overline{\mathsf{open}}_{PQ} = (\overline{\mathsf{open}}_P, \overline{\mathsf{open}}_Q)$ for P and Q. Any such attack fixes $p(s)$, the distribution of $s \in \{0,1\}^n \cup \{\perp\}$ that is output by V after the opening phase, in the obvious way.

A somewhat accepted definition for the binding property of a 2-prover *bit* commitment scheme, as it is for instance used in [CSST11, LKB$^+$15, FF15] (up to the factor 2 in the error parameter), is the *probability sum-binding* property defined below. Here, we assume it has been specified which attacks are *possible*, e.g., those where P and Q do not communicate during the course of the scheme.

Definition 2.14. *A 2-prover bit-commitment scheme is* ε-sum-binding *if for every possible commit strategy* $\overline{\mathsf{com}}_{PQ}$*, and for every pair of possible opening strategies* $\overline{\mathsf{open}}^0_{PQ}$ *and* $\overline{\mathsf{open}}^1_{PQ}$*, which fix distributions* $p(b_0)$ *and* $p(b_1)$ *for* V*'s respective outputs, it holds that*

$$p(b_0 = 0) + p(b_1 = 1) \leq 1 + 2\varepsilon\,.$$

In the literature (see e.g. [CSST11] or [LKB$^+$15]), the two probabilities $p(b_0 = 0)$ and $p(b_1 = 1)$ above are usually referred to as p_0 and p_1, respectively.

In the information theoretic setting, a commitment scheme can not be both hiding and binding with good (i.e., low) parameters. Thus, we have to assume some restriction on the provers, e.g., that they are unable to communicate during the execution of the scheme. However, we might also be more liberal and allow some limited communication during the protocol, as in the Lunghi *et al.* multi-round scheme.

If we rely on relativity to enforce the communication restrictions, we need to make sure that the provers are at the appropriate distance from each other. We do not address this problem in this thesis and assume that the provers are at fixed known positions. However, depending on the commitment scheme, it can be possible to also split the verifier into two separate agents that each only need to communicate with one prover. These are then placed next to the provers they communicate with, and brought together at the end of the opening phase to compute the result. For the Lunghi *et al.* scheme, this is possible – in fact, it is how the scheme is presented in [LKB$^+$15]. For simplicity, we describe protocols with only one verifier.

Our proof of the commitment scheme relies on different, but stronger or equivalent, notions of a binding commitment. We explain these, and the relations among them in Chapter 3.

2.2.3 The $\mathcal{CHSH}^q$ Scheme

Our main example is a generalization of the bit-commitment scheme by Crépeau *et al.* [CSST11]. Let q be a prime power and let $\mathbb{F}_q$ be the finite field with q elements. The bit-commitment scheme $\mathcal{CHSH}^q$ works as follows: The commit phase com_{PQV} instructs V to sample and send to P a uniformly random $a \in \mathbb{F}_q$, and it instructs P to return $x := r + a \cdot b$ to V, where $r \in \mathbb{F}_q$ is the provers' joint randomness and b is the bit to commit to. The opening phase open_{PQV} instructs Q to send $y := -r$ to V, and V outputs the (smaller) bit b that satisfies $x + y = a \cdot b$, and $b := \perp$ in case no such bit exists. Note that the provers in this scheme use the same randomness in the commit and opening phase; thus, formally, Q needs to output the shared randomness as $state_Q$. The opening phase uses no fresh randomness.

It is easy to see that this scheme is q^{-1}-complete and perfectly hiding (completeness fails in case $a = 0$). For *classical* provers that do not communicate at all, the scheme is $\left(q^{-1}/2\right)$-sum-binding. As for *quantum* provers, Crépeau *et al.* showed that the scheme $\mathcal{CHSH}^{2^n}$ is $2^{-n/2}$-binding; this was recently minorly improved to $2^{-(n+1)/2}$ by Sikora, Chailloux and Kerenidis [SCK14].

We also want to consider an extended version of the scheme where, instead of a bit, the provers commit to an arbitrary field element $s \in \mathbb{F}_q$, thus making $\mathbb{F}_q$ the domain of the scheme. In the opening phase, the verifier's output is picked as above, except that it is selected from the set $\mathbb{F}_q$ instead of $\{0, 1\}$. If $a \neq 0$, the output is $s = a^{-1}(x + y)$. In general, we will thus view $\mathcal{CHSH}^q$ as a string-commitment scheme, and explicitly mention when we restrict its domain to $\{0, 1\}$.

However, it is a priori not clear what a suitable definition for the binding property is, especially because for this particular scheme, the dishonest provers can always honestly commit to a string s, and can then decide to correctly open the commitment to s by announcing $y := r$, or open to a *random* string by announcing a randomly chosen y — any y satisfies $x + y = a \cdot s$ for *some* s (unless $a = 0$, which almost never happens).[1]

Due to its close relation to the CHSH game [CHSH69], in particular to the arbitrary-finite-field version considered in [BS15], we will refer to this *string* commitment scheme as $\mathcal{CHSH}^q$.

[1] This could easily be prevented by requiring Q to announce s (rather than letting V compute it), but we want the information announced during the opening phase to fit into the domain of the commitment scheme.