Exact Simulation of the GHZ Distribution

Gilles Brassard *
Département d’informatique et de recherche opérationnelle
Université de Montréal

Luc Devroye †
School of Computer Science
McGill University

and

Claude Gravel ‡
Département d’informatique et de recherche opérationnelle
Université de Montréal

27 October 2013

Abstract

John Bell has shown that the correlations entailed by quantum mechanics cannot be reproduced by a classical process involving non-communicating parties. But can they be simulated with the help of bounded communication? This problem has been studied for more than twenty years and it is now well understood in the case of bipartite entanglement. However, the issue was still widely open for multipartite entanglement, even for the simulation of arbitrary von Neumann measurements on the tripartite GHZ state. Here, we give a complete solution for general n-partite GHZ states: von Neumann measurements can be simulated exactly with $O(n^2)$ bits of expected communication, and $O(n \log n)$ time is sufficient to carry out the protocol in parallel. Furthermore, only one party needs to be probabilistic and it needs to draw only $O(n)$ unbiased random bits. In the case of equatorial measurements, we improve earlier results with a protocol that needs only $O(n \log n)$ bits of communication and $O(n)$ parallel time. At the cost of a slight increase in the number of bits communicated, these tasks can be accomplished with a constant expected number of rounds.

Keywords and phrases. Entanglement simulation, Greenberger–Horne–Zeilinger (GHZ) state, Multiparty entanglement, von Neumann’s rejection algorithm, Universal method of inversion, Knuth-Yao’s sampling algorithm.

* brassard@iro.umontreal.ca
† lucdevroye@gmail.com
‡ claude.gravel@bell.net
1 Introduction

The issue of non-locality in quantum physics was raised in 1935 by Einstein, Podolsky and Rosen when they introduced the notion of entanglement [9]. Thirty years later, John Bell proved that the correlations entailed by entanglement cannot be reproduced by classical local hidden variable theories between noncommunicating (e.g. space-like separated) parties [2]. This momentous discovery led to the natural question of quantifying quantum non-locality.

A natural quantitative approach to the non-locality inherent in a given entangled quantum state is to study the amount of resources that would be required in a purely classical theory to reproduce exactly the probabilities corresponding to measuring this state. More formally, we consider the problem of sampling the joint discrete probability distribution of the outcomes obtained by people sharing this quantum state, on which each party applies locally some measurement on his share. Each party is given a description of his own measurement but not informed of the measurements assigned to the other parties. This task would be easy (for a theoretician!) if the parties were indeed given their share of the quantum state, but they are not. Instead, they must simulate the outcome of these measurements without any quantum resources, using as little classical communication as possible.

This conundrum was introduced by Tim Maudlin [16] in 1992 in the simplest case of linear polarization measurements at arbitrary angles on the two photons that form a Bell state such as $|\Phi^+\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$. Maudlin claimed that this required “the capacity to send messages of unbounded length”, but he showed nevertheless that the task could be achieved with a bounded amount of expected communication. Similar concepts were reinvented independently years later by other researchers [5, 19]. This led to a series of results, culminating with the protocol of Toner and Bacon to simulate arbitrary von Neumann measurements on a Bell state with a single bit of communication in the worst case [20], thus contradicting Maudlin’s claim. Later, Regev and Toner extended this result by giving a simulation of arbitrary binary von Neumann measurements (meaning that the outcome for each party can take only two values) on arbitrary bipartite states of any dimension using two bits of communication, also in the worst case [18]. Inspired by Steiner’s work [19], Cerf, Gisin and Massar showed that the effect of an arbitrary pair of positive-operator-valued measurements (POVMs) on a Bell state can also be simulated with a bounded amount of expected communication [7]. A more detailed early history of the simulation of quantum entanglement can be found in Section 6 of Ref. [4].

All this prior work is concerned strictly with the simulation of bipartite entanglement. Much less is known when it comes to simulating multipartite entanglement with classical communication, a topic that is still teeming with major open problems. Consider the simplest case, which is the simulation of independent arbitrary von Neumann measurements on the tripartite GHZ state, named after Greenberger, Horne and Zeilinger [13], which we shall denote $|\Psi_3\rangle = \frac{1}{\sqrt{2}}|000\rangle + \frac{1}{\sqrt{2}}|111\rangle$, or more generally on its $n$-partite generalization $|\Psi_n\rangle = \frac{1}{\sqrt{2}}|0^n\rangle + \frac{1}{\sqrt{2}}|1^n\rangle$.

The easiest situation arises in the case of equatorial measurements (defined below) on a GHZ state because all the marginal probability distributions obtained by tracing out one or more of the parties are uniform. Hence, it suffices in this case to simulate the $n$-partite correlation. Once this has been achieved, all the marginals can easily be made uniform [10]. Making the best of this observation, Bancal, Branciard and Gisin have given a protocol to simulate equatorial measurements on the tripartite and fourpartite GHZ states at an expected cost of 10 and 20 bits of communication, respectively [1]. Later on, Branciard and Gisin improved this in the tripartite case with a protocol using 3 bits of communication in the worst case [3]. The simulation of equatorial measurements on
\(|\Psi_n\rangle\) for \(n \geq 5\) was handled subsequently by Brassard and Kaplan, with an expected cost of \(O(n^2)\) bits of communication \([6]\).

Despite substantial effort, the case of arbitrary von Neumann measurements, even on the original tripartite GHZ state \(|\Psi_3\rangle\), was still wide open. Here, we solve this problem in the general case of the simulation of the \(n\)-partite GHZ state \(|\Psi_n\rangle\), for any \(n\), under a model in which a single party, whom we call the leader, has access to a source of unbiased independent random bits, all the other parties being purely deterministic. Furthermore, we have no needs for shared random variables between the parties. (Most of the prior art on the simulation of entanglement by classical communication required the parties to share continuous real random variables in an initialization phase \([16, 5, 19, 20, 18, 7]\), admittedly an unreasonable proposition, but there have been exceptions, such as Ref. \([15]\).) An expected number of \(6n + 14\) perfect random bits suffices to carry out our simulation—all of which can (but do not have to) be tossed by the leader. The expected communication cost is \(O(n^2)\) bits, but only \(O(n \log n)\) time if we count one step for sending bits in parallel provided no party sends or receives more than one bit in any single step. Furthermore, in the case of equatorial measurements, it is easy to modify our protocol and improve the earlier best result \([8]\) with an expected communication cost of only \(O(n \log n)\) bits and \(O(\log^2 n)\) parallel time. At the cost of a slight increase in the number of bits communicated and the number of required random bits, these tasks can be accomplished with a constant expected number of rounds.

More formally, the quantum task that we want to simulate is as follows. Each party \(i\) holds one qubit of state \(|\Psi_n\rangle = \frac{1}{\sqrt{2}}|0^n\rangle + \frac{1}{\sqrt{2}}|1^n\rangle\) and is given the description of a von Neumann measurement \(M_i\). By local operations, they collectively perform \(\otimes_{i=1}^n M_i\) on \(|\Psi_n\rangle\), thus obtaining one outcome each, say \(x_i \in \{-1, +1\}\), which is their output. The joint probability distribution \(p(x)\) of the \(x_i\)'s is defined by the joint set of measurements. Our purpose is to sample exactly this joint probability distribution by a purely classical process that involves no prior shared random variables and as little communication as possible. As mentioned above, previous simulations \([1, 3, 6]\) were able to succeed only when each individual measurement was equatorial (defined in Section \(2\)). In order to overcome this limitation, our complete solution builds on four ingredients: (1) Gravel’s recently published decomposition of \(p(x)\) as a convex combination of two sub-distributions \([12]\); (2) Knuth and Yao’s algorithm \([14]\) to sample exactly probability distributions assuming only a source of unbiased identically independently distributed (i.i.d.) bits rather than a source of i.i.d. continuous uniform random variables on the interval \([0, 1)\); (3) the universal method of inversion (probability integral transform) \([8]\), for instance; and (4) our own distributed version of the classic von Neumann’s rejection algorithm \([17]\).

We define precisely our problem in Section \(2\) and we give our convex decomposition of the GHZ distribution, which is the key to its simulation. Then, we explain how to sample according to a Bernoulli distribution even when only approximations to the distribution’s parameter are available. We also explain how the classic von Neumann rejection algorithm can be used to sample in the sub-distributions defined by our convex decomposition. However, little attention is paid to the fact that the various parameters that define the joint distribution are not available in a single place. Section \(3\) is concerned with the communication complexity issues. It culminates with a complete protocol to solve our problem. Improvements are presented subsequently to save time by using parallelism between the parties and to reduce the expected number of rounds to a constant.
2 Sampling exactly the GHZ distribution in the random bit model

Any von Neumann measurement $M$ can be conveniently represented by a point on the Bloch sphere, which can be specified by two parameters $\theta \in [0, 2\pi)$ and $\varphi \in [-\pi/2, \pi/2]$, in which azimuthal angle $\theta$ represents the equatorial part of the measurement and elevation angle $\varphi$ represents its real part. Recall that this means that $M$ is the Hermitian idempotent operator defined by

$$M = x\sigma_1 + y\sigma_2 + z\sigma_3 = \begin{pmatrix} \sin\varphi & e^{-i\theta}\cos\varphi \\ e^{i\theta}\cos\varphi & -\sin\varphi \end{pmatrix},$$

where $x = \cos\theta\cos\varphi$, $y = \sin\theta\cos\varphi$, $z = \sin\varphi$, and $\sigma_1, \sigma_2$ and $\sigma_3$ are the Pauli operators. A von Neumann measurement is said to be equatorial when its elevation angle vanishes.

Consider a set of $n$ von Neumann single-qubit measurements $M_j$, represented by their parameters $(\theta_j, \varphi_j)$, $1 \leq j \leq n$. This set of operators defines a joint measurement $M = \otimes_{i=1}^n M_i$. In turn, this measurement defines a probability distribution $p$, which we shall call the GHZ distribution, on the set $\{-1, +1\}^n$ of its eigenvalues. This distribution corresponds to the probability of all possible outcomes when the $n$-partite GHZ state $|\Psi_n\rangle = \frac{1}{\sqrt{2}}|0^n\rangle + \frac{1}{\sqrt{2}}|1^n\rangle$ is measured according to $M$.

It is shown in [11, 12], albeit in the usual computer science language in which von Neumann measurements are presented as a unitary transformation followed by a measurement in the computational basis, that the probability of $x = (x_1, \ldots, x_n) \in \{-1, +1\}^n$ can be decomposed as

$$p(x) = \cos^2\left(\frac{\theta}{2}\right) p_1(x) + \sin^2\left(\frac{\theta}{2}\right) p_2(x),$$

where $\theta = \sum_{j=1}^n \theta_j$ and

$$p_1(x) = \frac{1}{2} \left( a_1(x) + a_2(x) \right)^2, \quad p_2(x) = \frac{1}{2} \left( a_1(x) - a_2(x) \right)^2,$$

$$a_1(x) = \prod_{j=1}^n \cos\left(\frac{1}{2}(\varphi_j - \frac{x}{2}x_j)\right), \quad a_2(x) = \prod_{j=1}^n -\sin\left(\frac{1}{2}(\varphi_j - \frac{x}{2}x_j)\right).$$

Hence, we see that distribution $p(x)$ is a convex combination of sub-distributions $p_1(x)$ and $p_2(x)$, in which the coefficients $\cos^2(\theta/2)$ and $\sin^2(\theta/2)$ depend only on the equatorial part of the measurements, whereas the sub-distributions depend only on their real part. Furthermore, the squares of $a_1$ and $a_2$ are themselves discrete probability distributions. A direct derivation of these formulas is sketched in Appendix A.

Sampling $p$ is therefore a matter of sampling a Bernoulli distribution with defining parameter $\cos^2(\theta/2)$ before sampling either $p_1$ or $p_2$, whichever is the case. Notice that sampling $p_2$ is like sampling $p_1$ if, say, we replace $\varphi_1$ by $\varphi_1 + 2\pi$. As we shall see, full knowledge of the parameters is not required to sample exactly. We shall see in subsection 2.1 how to sample a Bernoulli distribution with an arbitrary $p \in [0, 1]$ as parameter (not the same $p$ as our probability distribution for GHZ) using a sequence of approximants converging to $p$ and using an expected number of only two unbiased identically independently distributed (i.i.d.) random bits. Subsequently, we shall see in subsection 2.2 how to sample $p_1$ by modifying von Neumann’s rejection algorithm in a way that it uses sequences of approximants and unbiased i.i.d. random bits. For simulating exactly the GHZ distribution, an expected number of $6n + 14$ perfect random bits is sufficient. Note that this is close to what would be the optimal result of Knuth and Yao [14] even if all the parameters were in the same party’s hand, rather than having to deal with a distributed problem. Indeed, we remind the reader that their result states that, for any discrete probability distribution with entropy $H$, the optimal expected number of bits is between $H$ and $H + 2$. The entropy of the GHZ distribution is bounded by $n$ bits.
2.1 Sampling a Bernoulli distribution

Assume that only a random bit generator is available to sample a given probability distribution and that the parameters that specify this distribution are only accessible as follows: we can ask for any number of bits of each parameter, but will be charged one unit of cost per bit that is revealed. We shall also be charged for each random bit requested from the generator.

To warm up to this conundrum, consider the problem of generating a Bernoulli random variable with parameter \( p \in [0,1] \). If \( U = 0.U_1U_2\ldots \) is the binary expansion of a uniform \([0,1]\) random variable, i.e. \( U_1, U_2, \ldots \) is our source of unbiased independent random bits, and if \( p = 0.p_1p_2\ldots \) is the binary expansion of \( p \) (in case \( p = 1 \) we can proceed as if it were \( 0.p_1p_2\ldots \) with each \( p_i = 1 \)), we compare bits \( U_i \) and \( p_i \) for \( i = 1, 2, \ldots \) until for the first time \( U_i \neq p_i \). Then, if \( U_i = 0 < p_i = 1 \), we return \( X = 1 \), and if \( U_i = 1 > p_i = 0 \), we return \( X = 0 \). It is clear that \( X = 1 \) if and only if \( U < p \). Therefore, \( X \) is Bernoulli(\( p \)). The expected number of bits required from \( p \) is precisely 2. The expected number of bits needed from our random bit source is also 2.

Now, suppose that the parameter \( p \) defining our Bernoulli distribution is given by \( p = \cos^2(\theta/2) \), as in the case of our decomposition of the GHZ distribution. None of the parties can know \( \theta \) precisely since it is distributed as a sum of \( \theta_i \)'s, each of which is known only by one individual party. If we could obtain as many physical bits of \( p \) as needed (although the expected number of required bits is as little as 2), we could use the idea given above in order to sample according to this Bernoulli distribution. However, it is not possible in general to know even the first bit of \( p \), called \( p_1 \) above, given any fixed number of bits of the \( \theta_i \)'s. (For instance, if \( \theta \) is arbitrarily close to \( \pi/2 \), we need arbitrarily many bits of precision about it before we can tell if the first bit in the binary expansion of \( \cos^2(\theta/2) \) is 0 or 1). Nevertheless, we can use approximations of \( p \), rather than truncations, which in turn can come from approximations of \( \theta \). We postpone to Section 3.1 the detail of how these approximations can be obtained in a distributed setting. For the moment, assume that, for any \( k \), we can obtain \( p(k) \) so that \(|p(k) - p| \leq 1/2^k\). Then, setting \( U(k) = 0.U_1\ldots U_k \), we have that \( U \leq p \) if \( U(k) \leq p(k) - 2/2^k \) and that \( U \geq p \) if \( U(k) \geq p(k) + 1/2^k \). Considering furthermore that \( U = p \) can only happen with probability 0, the Algorithm 1 can be employed to check if \( U < p \) by generating only as many bits of \( U \) and increasingly good approximations of \( p \) as needed, and hence to generate a Bernoulli(\( p \)) random variable.

**Algorithm 1** Sampling a Bernoulli random variable with approximations of its defining parameter

1: Set \( k \leftarrow 1 \)
2: Set \( U(0) \leftarrow 0 \)
3: repeat forever
4: Generate an i.i.d. unbiased bit \( U_k \)
5: Compute \( U(k) \leftarrow U(k-1) + U_k/2^k \) \{hence \( U(k) = 0.U_1\ldots U_k \}\)
6: Obtain \( p(k) \) so that \(|p(k) - p| \leq 1/2^k\)
7: if \( U(k) \leq p(k) - 2/2^k \) then
8: \quad return \( Y = 1 \)
9: else if \( U(k) \geq p(k) + 1/2^k \) then
10: \quad return \( Y = 0 \)
11: else
12: \quad \( k \leftarrow k + 1 \)
13: end if
14: end repeat
The $Y$ generated by Algorithm 1 is Bernoulli($p$) because of the following lemma and the fact that $\mathbb{P}\{U < p\} = p$ if $U$ is a continuous uniform random variable on $(0, 1)$. In fact, the algorithm is directly inspired by this lemma.

**Lemma 1.** Let $(a_k)_{k \geq 0}$ and $(b_k)_{k \geq 0}$ be two sequences of real numbers converging to $a$ and $b$, respectively, with $a \neq b$. If, in addition, $(c_k)_{k \geq 0}$, $(c'_k)_{k \geq 0}$, $(d_k)_{k \geq 0}$, and $(d'_k)_{k \geq 0}$ are four decreasing sequences of positive real numbers converging to zero such that $-c_k < a_k - a < c'_k$ and $-d_k < b_k - b < d'_k$ for all $k$, then there exists a positive integer $K'$ such that for all $k > K'$ the following two statements hold:

\[
a_k - b_k \leq -(c_k + d'_k) \Rightarrow a < b, \quad a_k - b_k \geq c'_k + d_k \Rightarrow a > b.
\]

Choosing $a_k = U(k) \to U$, $b_k = p(k) \to p$ when $k \to \infty$, and $c_k = 2^{-k}$, $c'_k = 0$, and $d_k = d'_k = 2^{-k}$ in the previous lemma, we see that Algorithm 1 returns $Y = 1$ with probability $p$. Notice also that the algorithm never terminates if $a = U = p = b$, but this only happens with probability 0.

The number of iterations before returning a value, which is also the required number of independent unbiased random bits, is a random variable, say $K$. We have seen above that $\mathbb{E}\{K\}$, the expected value of $K$, would be exactly 2 if we could generate arbitrarily precise truncations of $p$. But since we can only obtain arbitrarily precise approximations instead, which is why we needed Algorithm 1 in the first place, we shall have to pay the price of a small increase in $\mathbb{E}\{K\}$.

\[
\mathbb{P}\{K > k\} = \mathbb{P}\{|U(k) - p(k)| \leq \frac{2}{2^k}\} \leq \mathbb{P}\{|U - p| \leq \frac{4}{2^k}\} \leq \frac{8}{2^k}.
\]

Therefore,

\[
\mathbb{E}\{K\} = \sum_{k=0}^{\infty} \mathbb{P}\{K > k\} \leq \sum_{k=0}^{\infty} \min\left(1, \frac{8}{2^k}\right) = 4.
\]

Note that the expected number of random bits used for sampling a Bernoulli by this process falls within the bounds of Knuth and Yao [14].

### 2.2 Sampling $p_1$ (or $p_2$) in the random bit model

As mentioned already, it suffices to concentrate on $p_1$ since one can sample $p_2$ in exactly the same way provided one of the angles $\varphi_i$ is replaced by $\varphi_i + 2\pi$: this introduces the required minus sign in front of $a_2$ to transform $p_1$ into $p_2$. First of all, it is useful to rewrite $p_1(x)$. To this end, define

\[
\alpha_j = \cos\left(\frac{1}{2}(\varphi_j - \frac{\pi}{2})\right) = \sin\left(\frac{1}{2}(\varphi_j + \frac{\pi}{2})\right) \quad \text{and} \quad \beta_j = \cos\left(\frac{1}{2}(\varphi_j + \frac{\pi}{2})\right) = -\sin\left(\frac{1}{2}(\varphi_j - \frac{\pi}{2})\right),
\]

and recall that $\alpha_j^2 + \beta_j^2 = 1$. Consider an arbitrary $x = (x_1, \ldots, x_n) \in \{-1, +1\}^n$ and let

\[
A = \{j \mid x_j = -1\} \quad \text{and} \quad B = \{j \mid x_j = +1\}.
\]

Then $p_1(x)$ can conveniently be rewritten as follows:

\[
p_1(x) = \frac{1}{2} \left( \prod_{j \in A} \beta_j \prod_{j \in B} \alpha_j + \prod_{j \in A} (-\alpha_j) \prod_{j \in B} \beta_j \right)^2
\]

\[
= \frac{1}{2} \left( \prod_{j \in A} \beta_j^2 \prod_{j \in B} \alpha_j^2 + \prod_{j \in A} \alpha_j^2 \prod_{j \in B} \beta_j^2 + 2(-1)^{|A|} \prod_{j=1}^{n} \alpha_j \beta_j \right).
\]
This is a decomposition into three terms. Note that
\[ q_1(x) \overset{\text{def}}{=} \prod_{j \in A} \beta_j^2 \prod_{j \in B} \alpha_j^2 = a_1^2(x) \]
defines a probability distribution over \([-1, +1]^n\). Indeed, if we define \(X_j\) as a Rademacher random variable that takes value \(-1\) with probability \(\beta_j^2\) and \(+1\) with complementary probability \(\alpha_j^2\), then the random vector with independent components given by
\[(X_1, \ldots, X_n)\]
has distribution \(q_1\). Similarly, if we set
\[ q_2(x) \overset{\text{def}}{=} \prod_{j \in A} \alpha_j^2 \prod_{j \in B} \beta_j^2 = a_2^2(x), \]
then the random vector with independent components given by
\[(-X_1, \ldots, -X_n)\]
has distribution \(q_2\).

The key observation is that both \(q_1\) and \(q_2\) can be sampled without any needs for communication because each party \(j\) knows parameters \(\alpha_j^2\) and \(\beta_j^2\), which is sufficient to draw independently according to local Rademacher random variable \(X_j\) or \(-X_j\). Moreover, a single unbiased independent random bit \(s\) drawn by a designated party suffices to sample collectively from distribution \(q = \frac{1}{2}(q_1 + q_2)\), provided this bit is transmitted to all parties: everybody samples according to \(q_1\) if \(s = 0\) or to \(q_2\) if \(s = 1\). Now, It is easy to verify that \(p_1(x) + p_2(x) = a_1^2(x) + a_2^2(x) = q_1(x) + q_2(x)\) for all \(x \in \{-1, +1\}^n\), and therefore \(p_1(x) \leq q_1(x) + q_2(x) = 2q(x)\).

The relevance of all these observations is that we can apply von Neumann’s rejection algorithm \([17]\) to sample \(p_1(x)\) since it is bounded by a small constant (2) times an easy-to-draw probability distribution \(q\). For the moment, we assume once again the availability of a continuous uniform random generator, which we shall later replace by a source of unbiased independent random bits. We also assume for the moment that we can compute the \(\alpha_i\)'s, \(p_1(x)\), \(q_1(x)\) and \(q_2(x)\) exactly. This gives rise to Algorithm 2.

By the general principle of von Neumann’s rejection algorithm, because \(p_1(x) \leq 2q(x)\) for all \(x \in \{-1, +1\}^n\), the expected number of iterations round the loop before probability distribution \(p_1\)

**Algorithm 2** Sampling \(p_1\) using von Neumann rejection algorithm

1: repeat
2: Generate \(U\) uniformly on \([0, 1]\)
3: Generate independent Rademacher random variables \(X_1, \ldots, X_n\) with parameters \(\alpha_1^2, \ldots, \alpha_n^2\)
4: Generate an unbiased independent random bit \(S\)
5: if \(S = 1\) then
6: set \(X \leftarrow (X_1, \ldots, X_n)\)
7: else
8: set \(X \leftarrow (-X_1, \ldots, -X_n)\)
9: end if
10: until \((q_1(X) + q_2(X))U \leq p_1(X)\)
is successfully sampled is at most 2. Within one iteration, exactly 2 expected independent unbiased random bits suffice to generate each of the \( n \) Rademacher random variables by a process similar to what is explained in the second paragraph of Section 2.1. Hence an expected total of \( 2n + 1 \) random bits are needed each time round the loop for an expected grand total of no more than \( 4n + 2 \) bits. The entropy of \( p_1 \) being bounded by \( n \), we are not far from reaching Knuth and Yao’s bound \([13]\) on the number of expected random perfect bits that would be required to sample it exactly in the random bit model. But of course, this does not take account of the (apparent) need to generate continuous uniform \([0,1)\) random variable \( U \). It follows that the expected total amount of work required by Algorithm 2 is \( O(n) \), provided we count infinite real arithmetic at unit cost. Furthermore, the time taken by this algorithm, divided by \( n \), is stochastically smaller than a geometric random variable with constant mean, so its tail is exponentially decreasing.

Now, we modify and adapt this algorithm to eliminate the need for the continuous uniform \( U \) (and hence its generation), which is not allowed in the random bit model. Furthermore, we eliminate the need for infinite real arithmetic and for the exact values of \( q_1(X), q_2(X) \) and \( p_1(X) \), which would be impossible to obtain in our distributed setting since the parameters needed to compute these values are scattered among all participants, and replace them with approximations—we postpone to Section 3.2 the issue of how they can be computed. (On the other hand, arbitrarily precise values of the \( \alpha_i \)'s are available to generate independent Rademacher random variables with these parameter.)

In each iteration of Algorithm 2, we generated a pair \((U,X)\). However, we did not really need \( U \): we merely need to generate a Bernoulli random variable \( Y \) for which

\[
\Pr\{Y = 1\} = \Pr\{(q_1(X) + q_2(X))U \leq p_1(X)\}.
\]

For this, we adapt the method developed for Algorithm 1. Again, we denote by \( U(k) \) the \( k \)-th approximant of \( U \) obtained by truncating \( U \) to the first \( k \) bits in its binary expansion, so that \( U(k) < U < U(k) + 2^{-k} \), except with probability 0. Furthermore, we denote by \( L_k \) (\( L \) for left) and \( R_k \) (\( R \) for right) the \( k \)-th approximants of \( (q_1(X) + q_2(X))U \) and \( p_1(X) \), respectively, which means that \( |L_k - (q_1(X) + q_2(X))| \leq 2^{-k} \) and \( |R_k - p_1(X)| \leq 2^{-k} \). Then using \( \varepsilon_k \) to denote arbitrary real numbers so that \( |\varepsilon_k| \leq 1 \),

\[
|U(k)L_k - U(q_1(X) + q_2(X))| = \left| U(k)L_k - U\left(L_k + \frac{\varepsilon_k}{2^k}\right) \right|
\]

\[
= \left| (U(k) - U)L_k - \frac{U\varepsilon_k}{2^k} \right|
\]

\[
\leq \frac{L_k}{2^k} + \frac{1}{2^k}
\]

\[
\leq \frac{3}{2^k}.
\]

Similarly,

\[
|R_k - p_1(X)| \leq \frac{1}{2^k}.
\]

Thus, we know that \( Y = 1 \) if

\[
U(k)L_k + \frac{3}{2^k} < R_k - \frac{1}{2^k}.
\]

Also, \( Y = 0 \) if

\[
U(k)L_k - \frac{3}{2^k} > R_k + \frac{1}{2^k}.
\]
Algorithm 3 Generator for the stopping condition in Algorithm 2

1: Note: $X \in \{-1, +1\}^n$ is given to the algorithm, generated according to $\frac{q_1 + q_2}{2}$
2: Set $k \leftarrow 1$
3: Set $U(0) \leftarrow 0$
4: repeat forever
5: Generate an i.i.d. unbiased bit $U_k$
6: Compute $U(k) = U(k - 1) + U_k/2^k$ {hence $U(k) = 0.U_1 \ldots U_k$}
7: Compute $L_k$ and $R_k$ from $X$
8: if $U(k) L_k - R_k < -\frac{4}{2^k}$ then
9:    return $Y = 1$
10: else if $U(k) L_k - R_k > \frac{4}{2^k}$ then
11:    return $Y = 0$
12: else
13:    $k \leftarrow k + 1$
14: end if
15: end repeat

It follows that Algorithm 3 can be used to sample random variable $Y$, which is used as terminating condition in Algorithm 2 in order to eliminate its need for the generation of a continuous uniform random variable $U \in [0, 1)$ and for the precise values of $q_1(X), q_2(X)$ and $p_1(X)$.

Since $L_k \rightarrow q_1(X) + q_2(X)$ and $R_k \rightarrow p_1(X)$ as $k \rightarrow \infty$, Algorithm 3 halts with probability 1 by Lemma 1. If it has not halted after having processed $L_k$ and $R_k$, then we know that

$$|U(q_1(X) - q_2(X)) - p_1(X)| = \left| (U(q_1(X) - q_2(X)) - U(k)L_k) + (R_k - p_1(X)) + (-R_k + U(k)L_k) \right|$$

$$\leq |U(q_1(X) - q_2(X)) - U(k)L_k| + |R_k - p_1(X)| + |R_k - U(k)L_k|$$

$$\leq \frac{3}{2^k} + \frac{1}{2^k} + \frac{4}{2^k}$$

$$= \frac{8}{2^k}.$$  

But since $U$ is uniformly distributed on $[0, 1]$, the probability of this is no more than

$$\frac{8}{2^k (q_1(X) + q_2(X))}.$$  

Let $K$ be the value of $k$ upon exit from the loop in the algorithm, which is the number of bits needed from the binary representations of $q_1(X) + q_2(X)$ and $p_1(X)$ in order to reject or accept. We have

$$\mathbb{P}\{K > k|X\} \leq \frac{8}{2^k(q_1(X) + q_2(X))}.$$  

9
Thus by denoting $k_0 = 3 + \log_2 \left( \frac{1}{q_1(X) + q_2(X)} \right)$,

\[
E\{K|X\} = \sum_{k=0}^{\infty} P\{K > k | X\} \\
\leq \sum_{k=0}^{\infty} \min \left( 1, \frac{8}{2^k(q_1(X) + q_2(X))} \right) \\
\leq \sum_{k\leq k_0} 1 + \sum_{k>k_0} \frac{8}{2^k(q_1(X) + q_2(X))} \\
\leq (k_0 + 1) + 1 \\
= 5 + \log_2 \left( \frac{1}{q_1(X) + q_2(X)} \right).
\]

Now, we uncondition, and obtain

\[
E\{K\} \leq 5 + \sum_{x \in \{0,1\}^n} q_1(x) + q_2(x) \log_2 \left( \frac{1}{q_1(x) + q_2(x)} \right) \\
\leq 5 + \log_2 \left( \sum_{x \in \{0,1\}^n} \frac{q_1(x) + q_2(x)}{2} \times \frac{1}{q_1(x) + q_2(x)} \right) \\
\text{(by Jensen’s inequality)} \\
= 5 + (n - 1) \\
= n + 4.
\]

It is unimportant for what follows, but worth noting nevertheless, that the intermediate bound on $E\{K\}$ involves the binary entropy of the distribution $\frac{1}{2}(q_1 + q_2)$.

### 3 Communication complexity of sampling

In this section, we consider the case in which the sampler of the previous section no longer has full knowledge of the GHZ distribution to be simulated. The sampler, whom we call the leader in a distributed setting, has to communicate through classical channels in order to obtain partial knowledge of the parameters belonging to the other parties. Partial knowledge results in approximation of the distributions involved in sampling the GHZ distribution, but, as we saw in the previous section, we know how to sample exactly in the random bit model using approximations. We consider two models of communication: in the sequential model, the leader has a direct channel with everyone else and all the communication takes place sequentially because the leader cannot listen to everyone at the same time; in the parallel model, parties communicate with one another in a tree-structured way, with the leader at the root, which allows us to save on communication time, at the expense of a small increase in the total number of bits that need to be communicated. Although all parties toss random coins in what follows, it is easy to modify the protocol in a way that only the leader needs the availability of a source of unbiased identically independently distributed random bits, all the other parties being purely deterministic, at a negligible cost in terms of communication complexity.

**Definition:** A $k$-bit approximation of a quantity $v$ is any $\hat{v}$ such that $|v - \hat{v}| \leq 2^{-k}$. For instance, $v$ can be a sum of numbers, a product of numbers or a real-valued function of one or many variables.
A special case of $k$-bit approximation is the $k$-bit truncation $\hat{v} = \lfloor v 2^k \rfloor / 2^k$. For convenience, we sometimes use the shortcuts $k$-approximation and $k$-truncation.

### 3.1 Sampling a Bernoulli distribution whose parameter is distributed

In order to sample the GHZ distribution, we know from Section 2 that we must first sample the Bernoulli distribution with parameter $\cos^2(\theta/2)$, where $\theta = \sum_{j=1}^{n} \theta_j$. Let us say that the leader is party number 1. Since he knows only $\theta_1$, he must communicate with the other parties to obtain partial knowledge about $\theta_i$ for $i \geq 2$. The problem of sampling a Bernoulli distribution with probability $\cos^2(\theta/2)$ reduces to learning the sum $\theta$ with sufficient precision in order to use Algorithm 1.

The problem of computing a $k$-bit approximation of $\cos^2(\theta/2) = \cos^2(\sum_{i=1}^{n} \theta_i/2)$ is relatively easy. Define $\theta_j = \theta_j/2$ and $\theta_i = \theta_i/2$ for each $i$. If the leader obtains an $\ell$-bit approximation $\hat{\theta}_i$ of each $\theta_i$, $i \geq 2$, and if we define $\hat{\theta} = \sum_{i=1}^{n} \hat{\theta}_i$, we need to find the value of $\ell$ for which $\cos^2(\hat{\theta})$ is a $k$-bit approximation of $\cos^2(\theta) = \cos^2(\theta/2)$. By virtue of results on Taylor series expansion, we have

$$|\cos^2(\hat{\theta}) - \cos^2(\hat{\theta})| \leq \left( \sup_{(\theta_1, \ldots, \theta_n)} \| \nabla(\cos^2(\hat{\theta})) \| \right) \| \hat{\theta} - \hat{\theta} \| \leq \sqrt{n} \frac{\sqrt{n}}{2^\ell} = \frac{n}{2^\ell},$$

where $\| \cdot \|$ denotes the Euclidean norm of a vector. It suffices to choose $\ell = k + \lceil \log_2 n \rceil$ in order to conclude as required that $|\cos^2(\hat{\theta}) - \cos^2(\hat{\theta})| \leq \frac{1}{2^\ell}$. Taking into account the integer part of each $\theta_i$, which must also be communicated, and remembering that $0 \leq \theta_i \leq 2\pi$ since it is an angle,

the required number of communicated bits in the sequential model is therefore $(n - 1)(\ell + 3) = (n - 1)(3 + k + \lceil \log_2 n \rceil)$, which is $O(kn + n \log n)$. In this case, the expected value of $k$ is bounded by 4 (see the analysis of the Bernoulli sampling Section 2.1), so that this operation requires an expected communication of $O(n \log n)$ bits in the sequential model.

### 3.2 Approximating a product of bounded numbers

Once the leader has produced a bit $B$ with probability $\cos^2(\theta/2)$, he samples either $p_1$ or $p_2$, depending on whether he got $B = 0$ or $B = 1$. The problem of sampling $p_2$ reduces to sampling $p_1$ if the leader replaces his own $\varphi_1$ with $\varphi_1 + 2\pi$; thus we concentrate on sampling $p_1$. Of course, the leader does not know $\varphi_i$ for $i \geq 2$. This problem reduces to learning with sufficient precision the products $a_1(X) = \prod_{j=1}^{n} \cos \left( \frac{1}{2} (\varphi_j - \frac{\pi}{2} X_j) \right)$ and $a_2(X) = \prod_{j=1}^{n} \sin \left( \frac{1}{2} (\varphi_j - \frac{\pi}{2} X_j) \right)$, given that the $X_j$’s are independent Rademacher distributions with parameters $\alpha_j^2$, $1 \leq i \leq n$. Once these products are known with, say, $k + 2$ bits of precision, the left and right $k$-bit approximations $L_k$ and $R_k$ are easily computed, which allows us to run the modified von Neumann’s rejection algorithm from the previous section.

In this section, we explain how to compute a $k$-bit approximation to $a_1(X)$ and $a_2(X)$ at an expected communication cost of $O(kn + n \log n)$ bits in the sequential model, which translates to an expected $O(k \log n + \log^2 n)$ time in the parallel model. For our specific application of simulating the GHZ distribution, we proved at the end of Section 2.2 that the expected value of $k$ is bounded by $n + 4$. It follows that an expected cost of $O(n^2)$ bits suffices in the sequential model and $O(n \log n)$ time in the parallel model.

---

1 Actually, $0 \leq \theta_i \leq \pi$ since $\theta_i$ is a half angle and one fewer bit is needed to communicate its integer part, but we prefer to consider here the more general case of approximating the cosine square of a sum of arbitrary angles.
Given $X = (X_1, \ldots, X_n)$ with the $X_i$’s distributed according to non-identical independent Rademachers with parameter $\cos^2\left(\frac{1}{2}(\varphi_i - \frac{\pi}{2})\right)$ or $\cos^2\left(\frac{1}{2}(\varphi_i + \frac{\pi}{2})\right)$, we need to compute $k$-bit approximations of $a_1(X)$ and $a_2(X)$. We use $c_j$ and $s_j$ to denote $\cos\left(\frac{1}{2}(\varphi_j - \frac{\pi}{2}X_j)\right)$ and $-\sin\left(\frac{1}{2}(\varphi_j - \frac{\pi}{2}X_j)\right)$, respectively, as well as $\hat{c}_j$ and $\hat{s}_j$ to denote their respective $\ell$-approximations obtained by truncating $c_j$ and $s_j$ to $\ell$ bits. We need to determine $\ell$ such that the products $\prod_{j=1}^n \hat{c}_j$ and $\prod_{j=1}^n \hat{s}_j$ are $k$-approximations of $a_1(X)$ and $a_2(X)$, respectively. Notice that each party knows exactly his own $c_j$ and $s_j$, and hence $\hat{c}_j$ and $\hat{s}_j$ can be transmitted directly to the leader, rather than approximations of the $\varphi_i$’s. For each $c_j$, there exists $\varepsilon_j \in [-1, 1]$ such that $c_j = \hat{c}_j + \frac{\varepsilon_j}{2^\ell}$; thus, using $I$ to denote $\{1, 2, \ldots, n\}$, we have

$$\prod_{j=1}^n c_j = \sum_{A \in P(I)} \prod_{j \in A} \hat{c}_j \prod_{j \not\in A} \frac{\varepsilon_j}{2^\ell} = \prod_{j=1}^n \hat{c}_j + \sum_{A \in P(I) \setminus I} \prod_{j \in A} \hat{c}_j \prod_{j \not\in A} \frac{\varepsilon_j}{2^\ell}$$

and hence we can bound the error as follows:

$$\left|\prod_{j=1}^n c_j - \prod_{j=1}^n \hat{c}_j\right| \leq \sum_{j=1}^n \left(\binom{n}{j} \frac{1}{2^{\ell}}\right) - 1 = \left(1 + \frac{1}{2^\ell}\right)^n - 1.$$ 

Setting $\ell = \left\lceil -\log_2 \left( (1 + 2^{-k})^{1/n} - 1 \right) \right\rceil$ then

$$\left|\prod_{j=1}^n c_j - \prod_{j=1}^n \hat{c}_j\right| \leq \frac{1}{2^k}.$$ 

Notice that if $k \geq 1$ and $n \geq 1$, then

$$-\log_2 \left( (1 + 2^{-k})^{1/n} - 1 \right) \leq k + \log_2 n + \frac{1}{2 \log 2}.$$ 

Taking account of the need to transmit the $\ell$-approximations to both $c_j$ and $s_j$, which consists of the sign of these numbers in addition to the first $\ell$ bits of their binary expansion, the expected communication cost in the sequential model is $2(n - 1)(\ell + 1)$ bits, which once again is $O(kn + n \log n)$.

To bring this down to $O(k \log n + \log^2 n)$ time, we introduce the parallel model, in which parties communicate with one another according to a binomial tree structure. For simplicity, we may assume that $n$ is a power of 2. To understand the algorithm, think of a tree with nodes containing $s_j$ and $c_j$, for $1 \leq j \leq n$. Node number $j$ can be thought of as belonging to party number $j$, who knows $s_j$ and $c_j$ exactly. We pair parties by groups of 2. For a given pair, say $(j, j + 1)$, with $j$ odd, party $j + 1$ sends $\hat{s}_{j+1}$ and $\hat{c}_{j+1}$ to party $j$, who computes $\hat{s}_j \hat{s}_{j+1}$ and $\hat{c}_j \hat{c}_{j+1}$. Then, party $j$ is matched with party $j + 2$, where $j - 1$ is divisible by 4. This process gives rise to the new pair $(j, j + 2)$, from which emerges the products $\hat{s}_j \hat{s}_{j+1} \hat{s}_{j+2} \hat{s}_{j+3}$ and $\hat{c}_j \hat{c}_{j+1} \hat{c}_{j+2} \hat{c}_{j+3}$, and so on up to the leader, who is party 1 at the root of the tree. This process is illustrated in Figure 1 in which numbers in the nodes correspond to parties and numbers next to the arrows correspond to the order in which data is transmitted.
Figure 1: Binomial tree structure

Algorithm 4 Computing products in parallel, communicating along a binomial tree configuration

1: If \( n \) is not a power of 2, add virtual parties with \( c_j = s_j = 1 \) for \( n < j \leq 2^\lceil \log_2 n \rceil \)  
   (These parties, being dummy, have no effect on the overall communication complexity)
2: \( \ell \leftarrow k + \lceil \log_2 n \rceil + 1 \)
3: for \( j \leftarrow 1 \) to \( n \) in parallel do
4:   Party \( j \) does \( \tilde{c}_j \leftarrow \hat{c}_j \) and \( \tilde{s}_j \leftarrow \hat{s}_j \), which are \( \ell \)-approximations of \( c_j \) and \( s_j \), respectively
5: end for
6: \( m \leftarrow 1 \)
7: repeat
8:   for \( j \leftarrow 1 \) to \( n \) by step of \( 2m \) in parallel do
9:     Party \( j + m \) sends \( \tilde{c}_{j+m} \) and \( \tilde{s}_{j+m} \) to party \( j \)
10:    Party \( j \) computes \( \tilde{c}_j \leftarrow \hat{c}_j \hat{c}_{j+m} \) and \( \tilde{s}_j \leftarrow \hat{s}_j \hat{s}_{j+m} \), both truncated to \( \ell \) bits
11: end for
12: \( m \leftarrow 2m \)
13: until \( m \geq n \)
14: Party 1 (the leader) outputs \( \tilde{c}_1 \) and \( \tilde{s}_1 \)

This approach is formalized in Algorithm 4 in which new variables \( \tilde{c}_j \) and \( \tilde{s}_j \) are introduced to hold approximations of products of increasingly many \( c \)'s and \( s \)'s as the process unfolds. The issue of the required precision at each level of the process has to be reconsidered because the leader will no longer receive the entire list of \( \hat{c}_j \)'s and \( \hat{s}_j \)'s since subproducts are calculated en route by intermediate parties. It is shown in Appendix B that if \( \hat{x} \) and \( \hat{y} \) are \( \ell \)-truncations of \( x \) and \( y \), respectively, for arbitrary real numbers \( x \) and \( y \) in \( [-1,1] \) and integer \( \ell \), then \( \hat{x}\hat{y} \) is an \( (\ell - 1) \)-approximation of \( xy \). However, \( \hat{x}\hat{y} \) could have \( 2\ell \) bits of precision and we do not want to transmit so many bits up the binomial tree. There is an apparent problem if we transmit the \( \ell \)-truncation of \( \hat{x}\hat{y} \) instead, as we do indeed in Algorithm 4, because it is an \( (\ell - 2) \)-approximation of \( xy \) but not necessarily an \( (\ell - 1) \)-approximation. Nevertheless, it is shown in Appendix B that the recursive application of pairwise multiplications followed by \( \ell \)-truncations results in the loss of only one bit of precision per
cussed in Section 3.1 to reduce the number of rounds and to adapt it to make it work in the parallel model, as well as to make it go much faster in the case of equatorial measurements. It follows immediately that the parallel complexity of Algorithm 4 is $O(k \log n + \log^2 n)$.

Therefore, this takes $O(k \log n)$ time provided $k > \log_2 n$.

3.3 Protocol for sampling the GHZ distribution

We are finally ready to glue all the pieces together into Algorithm 5 (on next page), which has an expected complexity of $O(n^2)$ bits of communication in the sequential model. Variations are discussed in Section 3.4 to reduce the number of rounds and to adapt it to work in $O(n \log n)$ time in the parallel model, as well as to make it go much faster in the case of equatorial measurements.

**Correctness of the protocol:** The part occurring before the first “repeat” (line 5) samples a Bernoulli with parameter $\cos^2(\sum_{i=1}^{n} \theta_i/2)$, which allows the leader to decide whether to sample $X$ according to $p_1$ (by leaving his $\varphi_1$ unchanged) or according to $p_2$ (by adding $2\pi$ to his $\varphi_1$). Notice that the leader does not have to inform the other parties of this decision since they do not need to know if the sampling will be done according to $p_1$ or $p_2$. In Section 3.1, we showed how to sample exactly a Bernoulli with parameter $\cos^2(\sum_{i=1}^{n} \theta_i/2)$ when the $\theta_i$’s are not known to the leader for $i \geq 2$.

The part within the outer “repeat” loop (lines 5 to 27) is essentially von Neumann’s rejection algorithm, which has been adapted and modified to work in a distributed scenario. The leader must first know which of $q_1$ or $q_2$ to sample. For this purpose, he generates an unbiased random bit $S$ and broadcasts it to the other parties. Sampling either $q_1$ or $q_2$ can now be done locally and independently by each party $j$, yielding a tentative $X_j \in \{-1, +1\}$. The parties will output these $X_j$’s only at the end, provided this round is not rejected. Now, each party uses his $X_j$ to compute locally $c_j = \cos(\frac{1}{2}(\varphi_j - \frac{\pi}{2} X_j))$ and $s_j = -\sin(\frac{1}{2}(\varphi_j - \frac{\pi}{2} X_j))$, which will be sent bit by bit to the leader upon request, thus allowing him to compute increasingly precise approximations $L_k$ and $R_k$ of $q_1(X) + q_2(X)$ and $p_1(X)$, respectively. These values are used to decide whether a decision can be made to accept or reject this particular $X$, or whether more information is needed from the other parties to make this decision. As shown in the Section 2.2, the expected number of bits needed in $L_k$ and $R_k$ before we can break out of the “repeat forever” loop is $k \leq n + 4$. At that point, flag $Y$ tells the leader whether or not this was a successful run of von Neumann’s rejection algorithm. If $Y = 0$, the entire process has to be restarted from scratch, except for the initial Bernoulli sampling, at line 6. On the other hand, once the leader gets $Y = 1$, he can finally tell the other parties that they can output their $X_j$’s because, according to von Neumann’s rejection principle, this signals that the vector $(X_1, \ldots, X_n)$ is distributed according to $p_1$ (or $p_2$, depending on the initial Bernoulli). No more than $C = 2$ rounds of the outer “repeat” loop are expected to be required before we can thus conclude successfully.
Algorithm 5 Complete protocol for sampling the GHZ distribution in the sequential model

1: The leader, who is party number 1, generates a random bit $B$ according to a Bernoulli random distribution with parameter $\cos^2(\theta/2)$ using algorithm 1 modified according to Section 3.1 to take account of the fact that the value of $\theta = \sum_{i=1}^n \theta_i$ is distributed among the parties
2: if $B = 1$ then
3: The leader adds $2\pi$ to his own $\varphi$-parameter i.e. $\varphi_1 \leftarrow \varphi_1 + 2\pi$ {to sample $p_2$ rather than $p_1$}
4: end if
5: {Now entering the modified von Neumann’s “distributed” sampler for $p_1$}
6: repeat forever
7: The leader generates an i.i.d. unbiased bit $U_k$
8: end if
9: The leader starts talking with the other parties to decide whether or not to accept $X$
10: end if
11: The leader sets $k \leftarrow 1$
12: The leader sets $U(0) \leftarrow 0$
13: repeat forever
14: The leader generates a fair random bit $S$ and broadcasts it to the other parties
15: {The bit $S$ determines whether to sample $q_1$ or $q_2$}
16: The leader sets $k \leftarrow k + 1$
17: The leader requests $(k + 3 + \lceil \log_2 n \rceil)$-approximations of $c_j$ and $s_j$ from each party $j \geq 2$
18: The leader uses this information to compute $(k + 2)$-bit approximations of $a_1(X)$ and $a_2(X)$, as explained in Section 3.2 These approximations are used in turn to compute $k$-bit approximations $L_k$ of $q_1(X)$ and $q_2(X)$ and $R_k$ of $p_1(X)$
19: if $U(k)L_k - R_k < -\frac{4}{2\pi}$ then
20: Set $Y \leftarrow 1$ and break from the repeat forever loop. {Vector $X$ is accepted}
21: end if
22: Set $Y \leftarrow 0$ and break from the repeat forever loop. {Vector $X$ is rejected}
23: else
24: Set $k \leftarrow k + 1$ and continue the repeat forever loop
25: end if
26: until $Y = 1$ {accepting}
27: The leader informs all the other parties that the simulation is complete and, therefore, that the time has come for each party $j$ (including the leader himself) to output his current value of $X_j$
Expected communication cost and number of random coin: The expected amount of randomness used in this process is upper-bounded by $6n + 14$ bits. This is calculated as follows: the expected number of bits for sampling Bernoulli $B$ is bounded by 4. This is followed by an expectation of no more than $C = 2$ rounds of von Neumann’s rejection algorithm (the outer “repeat” loop). In each of these rounds, we need 1 bit for $S$ and expect 2 bits for each of the $X_j$’s (hence $2n$ in total), before entering the “repeat forever” loop. We expect to go round this loop $n + 4$ times, needing one more random bit $U_k$ each time. Putting it all together, the expected number of random bits is no more than $4 + 2(1 + 2n + (n + 4)) = 6n + 14$.

The expected amount of communication is dominated by the leader’s need to obtain increasingly accurate approximations of $c_j$ and $s_j$ from all other parties at line 17 in order to compute increasingly accurate approximations of $L_k$ and $R_k$, which he needs in order to decide whether or not to break from the “repeat forever” loop and, in such case, whether or not to accept $X$ as final output. On the $k$-th time round the “repeat forever” loop, the leader needs $k + 3 + \lceil \log_2 n \rceil$ bits of precision plus one bit of sign about each $c_j$ and $s_j$, $j \geq 2$ (in addition to having full knowledge about his own $c_1$ and $s_1$, of course). This would be very expensive if all those bits had to be resent each time round the loop, with increasing values of $k$. Fortunately, this process works well if the parties send truncations of these values to the leader, because each truncation simply does one bit of precision to the previous one. Hence, it suffices for the leader to request $2(5 + \lceil \log_2 n \rceil)$ bits from each other party at the onset when $k = 1$, and only two additional bits per party are needed afterwards for each subsequent trip round the loop (one for $c_j$ and one for $s_j$). All counted, a total of $2(n - 1)(k + 4 + \lceil \log_2 n \rceil)$ bits will have been requested from all other participants by the time we have gone through the “repeat forever” loop $k$ times. Since the expected value of $k$ upon exiting this loop is bounded by $n + 4$, we expect the leader to need $2(n - 1)((n + 4) + 4 + \lceil \log_2 n \rceil)$ bits of communication from the other participants to complete von Neumann’s rejection algorithm (lines 5 to 27). This is $O(n^2)$ expected bits of communication. The additional amount of communication required to sample Bernoulli $B$ at step 4 (which is $(n - 1)(5 + \log_2 n)$ bits) and for the leader to broadcast to all participants the value of $S$ as well as synchronization bits by which he needs to inform the other parties of success or failure each time round the loops is negligible. All counted, Algorithm 5 needs $O(n)$ bits of randomness and $O(n^2)$ bits of communication in order to sample exactly the GHZ distribution under arbitrary von Neumann measurements.

3.4 Variations on the theme

We can modify Algorithm 5 in a variety of ways to improve different parameters at the expense of others. Here, we mention briefly three of these variations: the parallel model, bounding the number of rounds, and the simulation of equatorial measurements.

The parallel model: Let us use Algorithm 4 to replace step 17 in Algorithm 5. This allows the leader to obtain his required $(k + 2)$-approximations of $a_1(X)$ and $a_2(X)$ with no need for him to learn all the $(k + 3 + \lceil \log_2 n \rceil)$-approximations of $c_j$ and $s_j$ from each party $j \geq 2$. We have seen that $O(k \log n + \log^2 n)$ parallel time is expected to suffice to this task. Unfortunately, this improvement is incompatible with the idea of transmitting only one more bit of information for each $c_j$ and $s_j$ when $k$ is increased by 1, which was crucial in the efficiency of the basic version of Algorithm 5 studied in Section 3.3. The problem stems from the fact that the $k$-truncation of the product of the $k$-truncations of $x$ and $y$ can be entirely different from the $(k + 1)$-truncation of the product of the $(k + 1)$-truncations of the same numbers. This is illustrated with $x = 0.1111 \ldots$ and $y = 0.1001 \ldots$ (in binary, of course). If we take $k = 3$, the truncations of $x$ and $y$ are 0.111 and 0.100, respectively,
whose product is 0.011100. In contrast, with \( k = 4 \), the truncations of \( x \) and \( y \) are 0.1111 and 0.1001, respectively, whose product is 0.10000111. We see that the 3-truncation of the product of the 3-truncations is 0.011, whereas the 4-truncation of the product of the 4-truncations is 0.1000, which are different on each bit of the fractional part. This demonstrates the fact that the bits going up the binomial tree in Algorithm 4 can change drastically from one run to the next even if a single bit of precision is added to all nodes at the bottom level, and therefore that we have to start afresh for each new value of \( k \). As a consequence, the use Algorithm 4 to replace step 17 in Algorithm 5 results in an “improvement” in which we expect to have to transmit \( \Omega(n^3) \) bits, taking \( \Omega(n^2 \log n) \) parallel time to do so!

Fortunately, there is an easy cure to this problem, which we only sketch here. In addition to using Algorithm 4 to replace step 17 in Algorithm 5, we also change step 24 from “\( k \leftarrow k + 1 \)” to “\( k \leftarrow 2k \)”. Even though parties have to transmit the entire \((k + 3 + \lceil \log_2 n \rceil)\)-truncations of each \( c_j \) and \( s_j \) for each new value of \( k \), the work done each time round the loop is roughly equivalent to the sum of all the work done until then. Since we expect to succeed when \( k \) is roughly equal to \( n \), the expected total parallel time will be about twice \( O(k \log n + \log^2 n) \) with \( k \approx n \), which is simply \( O(n \log n) \). The expected total number of bits communicated with this approach will be slightly greater than with Algorithm 5 but will remain \( O(n^2) \).

**Reducing the number of rounds:** Algorithm 5 is efficient in terms of the number of bits of randomness as well as the number of bits of communication, but it requires an expected \( O(n) \) rounds, in which the leader and all other parties take turn at sending messages. This could be prohibitive if they are far apart and their purpose is to try to convince examiners that they are actually using true entanglement and quantum processes to produce their joint outputs, because it would prevent them from responding quickly enough to be credible.

The solution should be rather obvious at this point, and we leave the details to the reader. If we change step 24 from “\( k \leftarrow k + 1 \)” to “\( k \leftarrow 2k \)”, the expected number of rounds is decreased from \( O(n) \) to \( O(\log n) \). If in addition we start with “\( k \leftarrow n \)” instead of “\( k \leftarrow 1 \)”, the expected number of rounds becomes a constant. We should refrain from using the parallel model, however, which requires logarithmically more messages than the sequential model to be transmitted between pairs of parties.

**Equatorial measurements:** Recall that equatorial measurements are those for which \( \varphi_i = 0 \) for each party \( i \). In this case, the leader can sample according to \( p_1 \) or \( p_2 \) without any help or communication from the other parties since he has complete knowledge of their elevation angles. Therefore, he can run steps 5 to 27 of Algorithm 5 all by himself! However, he needs to know from which of \( p_1 \) or \( p_2 \) to sample, which is provided by step 1 in Algorithm 5. The only remaining needs for communication occurs in step 28 which has to be modified from “The leader informs all the other parties that the simulation is complete” to “The leader informs all the other parties of which value of \( X_i \in \{-1,+1\} \) he has chosen from them”.

Only step 1 requires significant communication since the new step 28 needs only the transmission of \( n - 1 \) bits. We have already seen at the end of Section 3.1 that step 1 requires an expected communication of \( O(n \log n) \) bits in the sequential model. This is an improvement over the previously best technique known to simulate the GHZ distribution under arbitrary equatorial von Neumann measurements [6], which required an expectation of \( O(n^2) \) bits of communication.

In the parallel model, it is easy to adapt Algorithm 4 to work with sums instead of products, which is the relevant operation to parallelize step 1 of algorithm 5. In fact, parallelizing sums is easier than products because if \( \hat{x} \) and \( \hat{y} \) are \( \ell \)-truncations of \( x \) and \( y \), respectively, then \( \hat{x} + \hat{y} \) is not only an
($\ell - 1$)-approximation of $x + y$, but so is its $\ell$-truncation. We expect about 4 rounds in the “repeat forever” loop of Algorithm 1, in which round $k$ requires the leader to obtain a $k$-approximation of $\cos^2(\sum_{i=1}^{n} \theta_i)$, which in turn requires the transmission of $(3 + k + \lceil \log_2 n \rceil)$ bits from each of the $n - 1$ half-angles $\theta_i/2$ owned by the various parties $i \geq 2$. The binomial tree construction makes it possible to percolate this sum up to the leader through $\lceil \log_2 n \rceil$ levels for an expected total of $O(\log^2 n)$ time. Ironically, this makes trivial step 28 dominate over step 1 in terms of parallel time. Even though the leader is able to obtain the global outcome $X \in \{-1, +1\}^n$ after only $O(\log^2 n)$ time, the problem at hand requires him to distribute their outputs to each participant. According to our parallel model, of communication, no shortcut is possible since the leader must send the output bits one by one. Consequently, we cannot do better than $O(n)$ time in the parallel model to sample the GHZ distribution under arbitrary equatorial measurements.

Of course, we could have parallelized step 28 even in the case of non-equatorial measurements. However, this would not have impacted significantly on the overall time complexity of our solution, which remains $O(n \log n)$.

4 Discussion

We have addressed the problem of simulating the effect of arbitrary independent von Neumann measurements on the qubits forming the general GHZ state $\frac{1}{\sqrt{2}} |0^n\rangle + \frac{1}{\sqrt{2}} |1^n\rangle$ distributed among $n$ parties. Rather than doing the actual quantum measurements, the parties must sample the exact GHZ probability distribution by purely classical means, which necessarily requires communication in view of Bell’s theorem. Our main objective was to find a protocol that solves this conundrum with a finite amount of expected communication, which had only been known previously to be possible when the von Neumann measurements are restricted to being equatorial (a severe limitation indeed). Our solution needs only $O(n^2)$ bits of communication, which can be dispatched in $O(n \log n)$ time if bits can be sent in parallel provided nobody has to send or receive more than one bit in any given step. We also improved on the former art in the case of equatorial measurements, with $O(n \log n)$ bits of communication and $O(n)$ parallel time.

Knuth and Yao [14] initiated the study of the complexity of generating random integers (or bit strings) with a given probability distribution $p(x)$, assuming only the availability of a source of unbiased identically independently distributed random bits. They showed that any sampling algorithm must use an expected number of bits at least equal to the entropy $\sum_x p(x) \log_2(1/p(x))$ of the distribution, and that the best algorithm does not need more than three additional bits. For further results on the bit model in random variate generation, see chapter XIV of Ref. [8].

The GHZ distribution has a binary entropy no larger than $n$, and therefore it could be simulated with at most $n + 3$ random bits if all the parameters were concentrated in a single place. Even though we have studied the problem of simulating this distribution in a setting in which the defining parameters (here the description of the von Neumann measurements) are distributed among $n$ parties, and despite the fact that our main purpose was to minimize communication between these parties, we were able to succeed with $6n + 14$ expected random bits, which is less than six times the bound of Knuth and Yao.

We leave for further research the problem of simulating arbitrary positive-operator-valued measurements (POVMs) on the single-qubit shares of GHZ states, as well as the problem of simulating the GHZ distribution with worst-case bounded classical communication.
Acknowledgements

We wish to thank Marc Kaplan and Nicolas Gisin for stimulating discussions about the simulation of entanglement. Furthermore, Marc has carefully read Ref. [11], in which the decomposition of the GHZ distribution as a convex combination of two sub-distributions was first accomplished.

G.B. is supported in part by Canada’s Natural Sciences and Engineering Research Council of Canada (NSERC), the Canada Research Chair program, and the Canadian Institute for Advanced Research (CIFAR). L. D. is supported in part by NSERC and Fonds de recherche du Québec – Nature et technologies (FRQNT).

References

[1] J.-D. Bancal, C. Branciard and N. Gisin, “Simulation of equatorial von Neumann measurements on GHZ states using nonlocal resources”, Advances in Mathematical Physics 2010:293245, 2010.

[2] J. S. Bell, “On the Einstein-Podolsky-Rosen paradox”, Physics 1:195–200, 1964.

[3] C. Branciard and N. Gisin, “Quantifying the nonlocality of Greenberger-Horne-Zeilinger quantum correlations by a bounded communication simulation protocol”, Physical Review Letters 107:020401, 2011.

[4] G. Brassard. “Quantum Communication Complexity”, Foundations of Physics 33(11):1593–1616, 2003.

[5] G. Brassard, R. Cleve and A. Tapp, “Cost of exactly simulating quantum entanglement with classical communication”, Physical Review Letters 83:1874–1877, 1999.

[6] G. Brassard and M. Kaplan, “Simulating equatorial measurements on GHZ states with finite expected communication cost”, Proceedings of 7th Conference on Theory of Quantum Computation, Communication, and Cryptography (TQC), pp. 65–73, 2012.

[7] N. Cerf, N. Gisin and S. Massar, “Classical teleportation of a quantum bit” Physical Review Letters 84(11):2521–2524, 2000.

[8] L. Devroye, Non-Uniform Random Variate Generation, Springer-Verlag, New York, 1986.

[9] A. Einstein, B. Podolsky and N. Rosen, “Can quantum-mechanical description of physical reality be considered complete?”, Physical Review 47:777–780, 1935.

[10] N. Gisin, personal communication, 2010.

[11] C. Gravel, “Structure de la distribution de probabilité de l’état GHZ sous l’action de mesures de von Neumann locales”, M.Sc. thesis, Department of Computer Science and Operations Research, Université de Montréal: https://papyrus.bib.umontreal.ca/jspui/handle/1866/5511, 2011.

[12] C. Gravel, “Structure of the probability distribution for the GHZ quantum state under local von Neumann measurements”, Quantum Physics Letters 1(3):87–96, 2012.
[13] D.M. Greenberger, M.A. Horne and A. Zeilinger, “Going beyond Bell’s theorem”, in *Bell’s Theorem, Quantum Theory and Conceptions of the Universe*, edited by M. Kafatos (Kluwer Academic, Dordrecht), pp. 69–72, 1989.

[14] D.E. Knuth and A.C. Yao, “The complexity of nonuniform random number generation”, in: *Algorithms and Complexity*, edited by J.E. Traub, pp. 357–428, Academic Press, New York, 1976.

[15] S. Massar, D. Bacon, N. Cerf and R. Cleve, “Classical simulation of quantum entanglement without local hidden variables”, *Physical Review A* 63(5):052305, 2001.

[16] T. Maudlin, “Bell’s inequality, information transmission, and prism models”, *PSA: Proceedings of the Biennial Meeting of the Philosophy of Science Association*, pp. 404–417, 1992.

[17] J. von Neumann, “Various techniques used in connection with random digits. Monte Carlo methods”, *National Bureau Standards* 12:36–38, 1951. Reprinted in *Collected Works*, 5:768–770, Pergamon Press, 1963.

[18] O. Regev and B. Toner, “Simulating quantum correlations with finite communication”, *SIAM Journal on Computing* 39(4):1562–1580, 2009.

[19] M. Steiner, “Towards quantifying non-local information transfer: finite-bit non-locality”, *Physics Letters A* 270:239–244, 2000.

[20] B. Toner and D. Bacon, “Communication cost of simulating Bell correlations”, *Physical Review Letters* 91:187904, 2003.
A Convex decomposition of the GHZ distribution

Our simulation of GHZ distribution hinges upon its decomposition into a convex combination of two sub-distributions, as stated at the beginning of Section 2,

\[ p(x) = \cos^2\left(\frac{\theta}{2}\right) p_1(x) + \sin^2\left(\frac{\theta}{2}\right) p_2(x), \]

in which the coefficients \( \cos^2(\theta/2) \) and \( \sin^2(\theta/2) \) depend only on the equatorial part of the measurements, whereas the sub-distributions depend only on their real part. This decomposition was obtained by one of us [11, 12], albeit in the usual computer science language in which von Neumann measurements are presented as a unitary transformation followed by a measurement in the computational basis. For completeness, we give in this Appendix a rough sketch of how this decomposition can be obtained directly in the language of von Neumann measurements.

First, we remind the reader of some facts, including some already given in Section 2. We begin with a \( 2 \times 2 \) von Neumann measurement, which is a Hermitian idempotent operator \( M \) that can be written as

\[ M = x\sigma_1 + y\sigma_2 + z\sigma_3 = x\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} + y\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} + z\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} z & x - iy \\ x + iy & -z \end{pmatrix}. \]

The fact that \( M \) is idempotent implies that \( x^2 + y^2 + z^2 = 1 \). Thus, using the spherical coordinates \((\theta, \varphi) \in [0, 2\pi) \times [-\pi/2, \pi/2]\), the parameters \((x, y, z)\) can be written as

\[ x = \cos(\theta) \cos(\varphi) \]
\[ y = \sin(\theta) \cos(\varphi) \]
\[ z = \sin(\varphi) \]

so that

\[ M = \begin{pmatrix} \sin(\varphi) & \cos(\varphi) \exp(-i\theta) \\ \cos(\varphi) \exp(i\theta) & -\sin(\varphi) \end{pmatrix}. \]

The spectra (set of eigenvalues) of \( M \) is \( \{-1, +1\} \) and the unitary operator \( U \) that diagonalizes \( M \) is given by

\[ U = \begin{pmatrix} \alpha & -\bar{\beta} \\ -\beta & -\bar{\alpha} \end{pmatrix} \]

with

\[ \alpha = \cos\left(\frac{\varphi}{2} - \frac{\pi}{4}\right) \]
\[ \beta = \exp(i\theta) \sin\left(\frac{\varphi}{2} - \frac{\pi}{4}\right). \]

In other words, we have \( M = U \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} U^\dagger \).

Also, we have that the density matrix representing the GHZ state can be decomposed as

\[ \rho = |\Psi_n\rangle\langle\Psi_n| = \frac{1}{2} \left( \otimes_{j=1}^{n} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \otimes_{j=1}^{n} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \otimes_{j=1}^{n} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + \otimes_{j=1}^{n} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \right). \]
Before analyzing the joint probability function, we also remind the reader that $|+1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|-1\rangle = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$ and that

$$|b\rangle\langle b| = \begin{pmatrix} \delta_{+1}(b) & 0 \\ 0 & \delta_{-1}(b) \end{pmatrix},$$

where $\delta$ is the Kronecker delta function $\delta_{x}(y) = 1$ iff $x = y$ and $\delta_{x}(y) = 0$ iff $x \neq y$. We also invite the reader to verify that

$$\text{Trace}\left( |b\rangle\langle b| U \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} U^{\dagger}\right) = |\alpha|^2 \delta_{+1}(b) + |\beta|^2 \delta_{-1}(b)$$

$$= \cos^2\left(\frac{1}{2}\left(\varphi - \frac{\pi}{2} b\right)\right)$$

$$\text{Trace}\left( |b\rangle\langle b| U \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} U^{\dagger}\right) = -\alpha \beta \delta_{+1}(b) + \alpha \beta \delta_{-1}(b)$$

$$= -\exp(i\theta) \sin\left(\frac{1}{2}\left(\varphi - \frac{\pi}{2} b\right)\right) \cos\left(\frac{1}{2}\left(\varphi - \frac{\pi}{2} b\right)\right)$$

$$\text{Trace}\left( |b\rangle\langle b| U \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} U^{\dagger}\right) = -\alpha \beta \delta_{+1}(b) + \alpha \beta \delta_{-1}(b)$$

$$= -\exp(-i\theta) \sin\left(\frac{1}{2}\left(\varphi - \frac{\pi}{2} b\right)\right) \cos\left(\frac{1}{2}\left(\varphi - \frac{\pi}{2} b\right)\right)$$

$$\text{Trace}\left( |b\rangle\langle b| U \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} U^{\dagger}\right) = |\alpha|^2 \delta_{+1}(b) + |\beta|^2 \delta_{-1}(b)$$

$$= \sin^2\left(\frac{1}{2}\left(\varphi - \frac{\pi}{2} b\right)\right).$$

From now on, we write $E_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, $E_2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$, $E_3 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ and $E_4 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$ in order to save space.

Given $n$ von Neumann measurements $M_j$, or equivalently the $n$ unitary transformations $U_j$ for $j \in \{1, \ldots, n\}$ that diagonalize them, i.e. $M_j = U_j \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} U_j^{\dagger}$, and also their joint eigenvectors $|b\rangle = \otimes_{j=1}^{n}|b_j\rangle$ and the joint spectral operator $U = \otimes_{j=1}^{n}U_j$, the joint probability function of the
**eigenvalues** is given by the axioms of quantum mechanics:

\[
P\{b_1, \ldots, b_n\} = |\langle b | U | \Psi_n \rangle|^2 = \text{Trace} \left( (b | U | \Psi_n) (\langle b | U | \Psi_n \rangle)^\dagger \right) = \text{Trace} \left( \langle b | U | \Psi_n \rangle (\Psi_n | U^\dagger b \rangle \right) = \text{Trace} \left( (|b\rangle \langle b|) U \rho U^\dagger \right) = \frac{1}{2} \text{Trace} \left( \bigotimes_{j=1}^n |b_j\rangle \langle b_j| \left( \bigotimes_{j=1}^n U_j E_1 U_j^\dagger + \bigotimes_{j=1}^n U_j E_2 U_j^\dagger + \bigotimes_{j=1}^n U_j E_3 U_j^\dagger + \bigotimes_{j=1}^n U_j E_4 U_j^\dagger \right) \right) = \frac{1}{2} \sum_{i=1}^4 \prod_{j=1}^n \text{Trace} \left( \begin{pmatrix} \delta_{j+1}(b_j) & 0 \\ 0 & \delta_{j-1}(b_j) \end{pmatrix} U_j E_i U_j^\dagger \right).
\]

Putting these equations together, we have:

\[
\prod_{j=1}^n \text{Trace} \left( \begin{pmatrix} \delta_{j+1}(b_j) & 0 \\ 0 & \delta_{j-1}(b_j) \end{pmatrix} U_j E_1 U_j^\dagger \right) = \prod_{j=1}^n \cos^2 \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) = f_1,
\]

\[
\prod_{j=1}^n \text{Trace} \left( \begin{pmatrix} \delta_{j+1}(b_j) & 0 \\ 0 & \delta_{j-1}(b_j) \end{pmatrix} U_j E_2 U_j^\dagger \right) = \prod_{j=1}^n - \exp(n \theta_j) \sin \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) \cos \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) = f_2,
\]

\[
\prod_{j=1}^n \text{Trace} \left( \begin{pmatrix} \delta_{j+1}(b_j) & 0 \\ 0 & \delta_{j-1}(b_j) \end{pmatrix} U_j E_3 U_j^\dagger \right) = \prod_{j=1}^n - \exp(-n \theta_j) \sin \left( \frac{1}{2} \left( \varphi_j + \frac{\pi}{2} b_j \right) \right) \cos \left( \frac{1}{2} \left( \varphi_j + \frac{\pi}{2} b_j \right) \right) = f_3.
\]

\[
\prod_{j=1}^n \text{Trace} \left( \begin{pmatrix} \delta_{j+1}(b_j) & 0 \\ 0 & \delta_{j-1}(b_j) \end{pmatrix} U_j E_4 U_j^\dagger \right) = \prod_{j=1}^n \sin^2 \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) = f_4.
\]

Thus, \( P\{b_1, \ldots, b_n\} = \frac{1}{2} (f_1 + f_2 + f_3 + f_4). \)

Keeping in mind that

\[
f_2 + f_3 = 2 \cos \left( \sum_{j=1}^n \theta_j \right) \prod_{j=1}^n - \sin \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) \cos \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right)
\]

and

\[
\cos^2(t) = \cos^2(t/2) - \sin^2(t/2),
\]

it follows that

\[
P\{b_1, \ldots, b_n\} = \cos^2 \left( \frac{1}{2} \sum_{j=1}^n \theta_j \right) \left( \prod_{j=1}^n \cos \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) + \prod_{j=1}^n - \sin \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) \right)^2 + \sin^2 \left( \frac{1}{2} \sum_{j=1}^n \theta_j \right) \left( \prod_{j=1}^n \cos \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) - \prod_{j=1}^n - \sin \left( \frac{1}{2} \left( \varphi_j - \frac{\pi}{2} b_j \right) \right) \right)^2.
\]

23
If we define

\[ a_1(b) = \prod_{j=1}^{n} \cos \left( \frac{1}{2} (\phi_j - \frac{\pi}{2} x_j) \right), \quad a_2(b) = \prod_{j=1}^{n} - \sin \left( \frac{1}{2} (\phi_j - \frac{\pi}{2} x_j) \right), \]

\[ p_1(b) = \frac{1}{2} (a_1(b) + a_2(b))^2, \quad p_2(b) = \frac{1}{2} (a_1(b) - a_2(b))^2, \]

this is precisely the convex decomposition

\[ p(b) = \cos^2 \left( \frac{\theta}{2} \right) p_1(b) + \sin^2 \left( \frac{\theta}{2} \right) p_2(b), \]

where \( \theta = \sum_{j=1}^{n} \theta_j \), given at the beginning of Section 2.

As a “reality check”, it is elementary to analyse this formula for the special case of equatorial measurements, in which \( \phi_j = 0 \) for all \( j \). In particular, we obtain the well-known facts that all the marginal probability distributions obtained by tracing out one or more of the parties are uniform and that the expected value of the product of the \( b_j \)'s is equal to

\[ \cos^2 \left( \frac{1}{2} \sum_{j=1}^{n} \theta_j \right) - \sin^2 \left( \frac{1}{2} \sum_{j=1}^{n} \theta_j \right) = \cos \left( \sum_{j=1}^{n} \theta_j \right). \]

Those were indeed the formulas used in the prior art of simulating equatorial measurements on GHZ states [1, 3, 6].

### B Approximations and truncations in the parallel model

Hereinafter, \( |t| \) denotes the integer part of \( t \) for any \( t \in \mathbb{R} \). For any \( t \in \mathbb{R} \) and positive integer \( \ell \), we have that

\[ \left| \frac{|t|2^\ell}{2^\ell} - t \right| \leq \frac{1}{2^\ell}. \]

**Remark 1.** What parties are sending to each other are really quantities \( |t|2^\ell \) with the constraint that \( |t| \leq 1 \) so that it fits on a string of length \( \ell + 1 \) bits, taking account of the sign.

Suppose now that we have two numbers \( x_j, y_j \) at level \( j \) in the binomial tree inherent to Algorithm 4 such that \( |x_j| \leq 1 \) and \( |y_j| \leq 1 \). \( x_j \) and \( y_j \) can be expressed recursively using the numbers \( x_{j-1,1}, x_{j-1,2}, y_{j-1,1}, \) and \( y_{j-1,2} \) as

\[
\begin{align*}
x_j &= \left[ \frac{x_{j-1,1}2^\ell}{2^\ell} \frac{x_{j-1,2}2^\ell}{2^\ell} \right] \frac{1}{2^\ell} \\
y_j &= \left[ \frac{y_{j-1,1}2^\ell}{2^\ell} \frac{y_{j-1,2}2^\ell}{2^\ell} \right] \frac{1}{2^\ell}
\end{align*}
\]

We use \( \varepsilon_j \) for the error at level \( j \) on the product \( x_jy_j \); in other words,

\[
\left| \left[ \frac{x_j2^\ell}{2^\ell} \frac{y_j2^\ell}{2^\ell} \right] \frac{1}{2^\ell} - x_jy_j \right| = \varepsilon_j \iff \left| \left[ \frac{x_j2^\ell}{2^\ell} \frac{y_j2^\ell}{2^\ell} \right] \frac{1}{2^\ell} - \frac{x_{j-1,1}2^\ell}{2^\ell} \frac{y_{j-1,2}2^\ell}{2^\ell} \right| = \varepsilon_j
\]
Before upper bounding $\varepsilon_j$, we notice that,

$$\left| x_j - \frac{x_j 2^\ell}{2^\ell} \right| = \left| \frac{x_{j-1} 2^\ell}{2^\ell} - \frac{x_{j-1,2} 2^\ell}{2^\ell} + \frac{1}{2^\ell} \right| = \varepsilon_{j-1}$$

and the same for $y_j$.

Before upper bounding $\varepsilon_j$, we also establish the following inequality

$$\left| \frac{x_j 2^\ell}{2^\ell} \left| \frac{y_j 2^\ell}{2^\ell} \right| - x_j y_j \right| = \left| \frac{x_j 2^\ell}{2^\ell} \left| \frac{y_j 2^\ell}{2^\ell} \right| - x_j \frac{y_j 2^\ell}{2^\ell} + x_j \frac{y_j 2^\ell}{2^\ell} - x_j y_j \right| \leq \left| \frac{x_j 2^\ell}{2^\ell} \left| \frac{y_j 2^\ell}{2^\ell} \right| - x_j \frac{y_j 2^\ell}{2^\ell} \right| + \frac{1}{2^\ell} = \varepsilon_{j-1} + \varepsilon_{j-1} = 2 \varepsilon_{j-1}$$

Now we have that

$$\varepsilon_j = \left| \frac{x_j 2^\ell}{2^\ell} \left| \frac{y_j 2^\ell}{2^\ell} \right| - x_j y_j \right| \leq \left| \frac{x_j 2^\ell}{2^\ell} \left| \frac{y_j 2^\ell}{2^\ell} \right| - x_j \frac{y_j 2^\ell}{2^\ell} \right| + \frac{1}{2^\ell} = \frac{1}{2^\ell} + 2 \varepsilon_{j-1}$$

If there are $m = \lceil \log_2 n \rceil$ levels,

$$\varepsilon_m \leq \frac{1}{2^\ell} \sum_{j=0}^{m} 2^j < 2^{\lceil \log_2 n \rceil} = \frac{1}{2^\ell}.$$ 

It follows that a $k$-approximation of the product of $n$ numbers, which corresponds to $\varepsilon_m \leq \frac{1}{2^\ell}$, will be obtained if we truncate each intermediate subproduct to $\ell = k + \lceil \log_2 n \rceil + 1$ bits.