Quantum Random Access Codes
with Shared Randomness

Andris Ambainis[a], Debbie Leung[b],
Laura Mancinska[a,b], Maris Ozols[a,b]

[a] *Department of Computing, University of Latvia,*
Raina bulv. 19, Riga, LV-1586, Latvia

[b] *Department of Combinatorics and Optimization, and Institute for Quantum*
Computing, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada

October 2, 2008

Abstract

We consider a communication method, where the sender encodes n classical bits into 1 qubit and sends it to the receiver who performs a certain measurement depending on which of the initial bits must be recovered. This procedure is called $n \xmapsto{p} 1$ quantum random access code (QRAC) where $p > 1/2$ is its success probability. It is known that $2 \xmapsto{0.85} 1$ and $3 \xmapsto{0.79} 1$ QRACs (with no classical counterparts) exist and that $4 \xmapsto{p} 1$ QRAC with $p > 1/2$ is not possible.

We extend this model with shared randomness (SR) that is accessible to both parties. Then $n \xmapsto{p} 1$ QRAC with SR and $p > 1/2$ exists for any $n \geq 1$. We give an upper bound on its success probability (the known $2 \xmapsto{0.85} 1$ and $3 \xmapsto{0.79} 1$ QRACs match this upper bound). We discuss some particular constructions for several small values of n.

We also study the classical counterpart of this model where n bits are encoded into 1 bit instead of 1 qubit and SR is used. We give an optimal construction for such code and find its success probability exactly – it is less than in the quantum case.

Supplementary materials are available on-line at
http://home.lanet.lv/~sd20008/racs

Contents

1   Introduction

1.1   Random access codes

In general *random access code* (or simply RAC) stands for "encoding a long message into fewer bits and be able to recover (decode) any one of the initial bits (with some probability of success)". Random access code can be characterized by the symbol "$n \overset{p}{\mapsto} m$" meaning that n bits are encoded into m and any one of the initial bits can be recovered with probability at least p. In this paper we consider only the case when $m = 1$. So we have the following problem:

Problem (Classical)**.** *There are two parties – Alice and Bob. Alice is asked to encode some classical n-bit string into 1 bit and send this bit to Bob. We want Bob to be able to recover any one of the n initial bits with high success probability.*

Note that Alice does not know in advance which bit Bob will need to recover, so she cannot send only that bit. If they share a quantum channel then we have the quantum version of the previous problem:

Problem (Quantum)**.** *Alice must encode her classical n-bit message into 1 qubit (quantum bit) and send it to Bob. He performs some measurement on the received qubit to extract the required bit (the measurement that is used depends on which bit is needed).*

Both problems look similar, however the quantum version has an important feature. In the classical case the fact that Bob can recover any one of the initial bits implies that he can actually recover all of them – each with high probability of success. Surprisingly in quantum case this is not true, because after the first measurement the state of the qubit will be disturbed and further attempts to extract more information can fail.

It has been shown that there exist $2 \overset{0.85}{\mapsto} 1$ and $3 \overset{0.79}{\mapsto} 1$ *quantum random access codes* or QRACs (see [1] for the first code and [2] for both). It has also been shown [2] that it is impossible to construct a $4 \overset{p}{\mapsto} 1$ QRAC with $p > 1/2$. We will discuss these results more in Sect. 3.3.

We want to emphasize the setting in which the impossibility of $4 \overset{p}{\mapsto} 1$ QRAC was proved: Alice is allowed to perform a locally randomized encoding of the given string into a one-qubit state and Bob is allowed to perform different *positive operator–valued measure* (POVM) measurements to recover different bits. This is the most general setting when information is encoded into a one-qubit state and both parties are allowed to use randomized strategies, but are not allowed to cooperate. However, we can consider an even more general setting – when all kinds of classical randomness are allowed. This means that Alice and Bob are allowed to cooperate by using some shared source of randomness to agree on which strategy to use. We will refer to this source as a *shared random string* or *shared randomness* (SR). It turns out that in this new setting $4 \overset{p}{\mapsto} 1$ QRAC is possible with $p > 1/2$. In fact, $n \overset{p}{\mapsto} 1$ QRAC with $p > 1/2$ can be constructed for all $n \geq 1$.

1.2 Outline of results

In Sect. 2 we study classical $n \mapsto 1$ random access codes with shared randomness. In Sect. 2.2 we introduce Yao's principle that is useful for understanding both classical and quantum codes. Classical code that is optimal for all n is presented in Sect. 2.3.1 and the asymptotic behavior of its success probability is considered in Sect. 2.3.2.

In Sect. 3 we study quantum random access codes with shared randomness. In Sect. 3.3 we discuss what is known in the case when shared randomness is not allowed, i.e., $2 \mapsto 1$ and $3 \mapsto 1$ QRACs and the impossibility of $4 \mapsto 1$ QRAC. In Sect. 3.5 we give an upper bound of success probability of QRACs with SR and generalize it in Sect. 3.6 for POVM measurements. In Sect. 3.7 we construct $n \overset{p}{\mapsto} 1$ QRAC with SR and $p > 1/2$ for all $n \geq 2$.

In Sect. 4 we try to find optimal QRACs with SR for several small values of n. In particular, in Sect. 4.1 we discuss QRACs obtained by numerical optimization, but in Sect. 4.2 we consider symmetric constructions.

Finally, we conclude in Sect. 5 with the summary of the obtained results (Sect. 5.1), a list of open problems (Sect. 5.2) and possible generalizations (Sect. 5.3).

2 Classical random access codes

2.1 Types of classical encoding–decoding strategies

As a synonym for random access code we will use the term *strategy* to refer to the joint *encoding–decoding scheme* used by Alice and Bob. Two measures of how good the strategy is will be used: the *worst case success probability* and the *average success probability*. Both probabilities must be calculated over all possible pairs (x, i) where $x \in \{0, 1\}^n$ is the input and $i \in \{1, \ldots, n\}$ indicates which bit must be recovered. We are interested in the worst case success probability, but in our case according to Yao's principle (introduced in Sect. 2.2) the average success probability can be used to estimate it.

Depending on the computational model considered, different types of strategies are allowed. The simplest type corresponds to Alice and Bob acting deterministically and independently.

Definition. A *pure classical $n \mapsto 1$ encoding–decoding strategy* is an ordered tuple $(E, D_1, \ldots, D_n)$ that consists of *encoding function* $E : \{0, 1\}^n \mapsto \{0, 1\}$ and n *decoding functions* $D_i : \{0, 1\} \mapsto \{0, 1\}$.

These limited strategies yield RACs with poor performance. This is because Bob can replay all bits correctly for no more than two input strings, since he receives either 0 or 1 and acts deterministically in each case. For all other strings at least one bit will definitely be recovered incorrectly, therefore the worst case success probability is 0. If we allow Alice and Bob to act probabilistically but without cooperation, then we get mixed strategies.

Definition. A *mixed classical $n \mapsto 1$ encoding–decoding strategy* is an ordered tuple $(P_E, P_{D_1}, \ldots, P_{D_n})$ of probability distributions. P_E is a distribution over encoding functions and P_{D_i} over decoding functions.

It is obvious that in this setting we can obtain the worst case probability to be at least $1/2$. This is obtained by guessing – we output either 0 or 1 with probability $1/2$ regardless of the input. Formally it means that for each i, P_{D_i} is a uniform distribution over two constant decoding functions 0 and 1. It has been shown that in this setting for $2 \mapsto 1$ case one cannot do better than guessing, i.e., there is no $2 \overset{p}{\mapsto} 1$ RAC with the worst case success probability $p > 1/2$ [1].

However, we can allow the cooperation between Alice and Bob – they can use the shared random string to agree on some joint strategy.

Definition. A *classical $n \mapsto 1$ encoding–decoding strategy with shared randomness* is a probability distribution over pure classical strategies.

Note that this is the most general randomized setting, since both – randomized cooperation and local randomization are possible. It is demonstrated in the following example.

Example. Consider the following strategy: randomly agree on $i \in \{1, \ldots, n\}$ and send the ith bit; if ith bit is asked, replay the received bit, otherwise – guess. It can formally be specified as follows: uniformly choose a pure strategy from the set

$$\bigcup_{i \in \{1,\ldots,n\}} \left\{ (e_i, c_1, \ldots, c_{i-1}, d, g_1, \ldots, g_{n-i}) \mid c \in \{d_0, d_1\}^{i-1}, g \in \{d_0, d_1\}^{n-i} \right\},$$

where the encoding function e_i is given by $e_i(x) = x_i$ and decoding functions d_0, d_1, and d are given by $d_0(b) = 0$, $d_1(b) = 1$, and $d(b) = b$, where b is the received bit.

The amount of shared randomness used in the example is $n - 1 + \log n$ bits, because one out of $n \cdot 2^{n-1}$ pure strategies must be selected. Actually $\log n$ bits would suffice, if we had introduced the strategy with SR as a probability distribution over mixed strategies.

The classical strategies with SR is the model we are interested in, because they provide a classical analogue of QRACs with SR. However, in this setting the task to find the optimal strategy seems to be hard, therefore we will turn to Yao's principle for help.

2.2 Yao's principle

When dealing with randomized algorithms, it is hard to draw some general conclusions (like proving optimality of a certain randomized algorithm) because the possible algorithms may form a continuum. In such situation it is very helpful to apply Yao's principle [3]. It allows us to shift the randomness in the algorithm to the input and consider only deterministic algorithms.

Let $\mathcal{S}$ be a classical strategy with SR. One can think of it as a stochastic process consisting of applying the encoding map E to the input x, which is followed by applying the decoding map D_i of the ith bit. Both of these maps depend on the value of the shared random string. The result of $\mathcal{S}$ is $\mathcal{S}(x,i) = D_i(E(x))$, which is a stochastic variable over the set $\{0, 1\}$. Let $\Pr[\mathcal{S}(x,i) = v]$ denote the probability that the stochastic variable $\mathcal{S}(x,i)$ takes value v. Then the worst case success probability of the optimal classical strategy with SR is given by

$$\max_{\mathcal{S}} \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i]. \tag{1}$$

However, if we fix some distribution μ over the input set $\{0,1\}^n \times \{1, \ldots, n\}$, then the expected success probability of a pure (deterministic) strategy $\mathcal{P}$ is given by $\Pr_\mu[\mathcal{P}(x,i) = x_i]$. If the "hardest" input distribution is chosen as μ, then the expected success probability of the best pure strategy for this distribution is

$$\min_\mu \max_\mathcal{P} \Pr_\mu[\mathcal{P}(x,i) = x_i]. \tag{2}$$

Yao's principle states that the quantities given in (1) and (2) are equal [3]:

$$\max_\mathcal{S} \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i] = \min_\mu \max_\mathcal{P} \Pr_\mu[\mathcal{P}(x,i) = x_i]. \tag{3}$$

Thus Yao's principle provides us with an upper bound for the worst case probability (1). All we have to do is to choose an arbitrary input distribution μ_0 and find the best pure strategy $\mathcal{P}_0$ for it. Then according to Yao's principle we have:

$$\Pr_{\mu_0}[\mathcal{P}_0(x,i) = x_i] \geq \max_\mathcal{S} \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i], \tag{4}$$

with equality if and only if μ_0 is the "hardest" distribution. It turns out that for random access codes the uniform distribution η is the "hardest". To prove it, we must first consider the randomization lemma.

Lemma 1. $\forall \mathcal{P} \exists \mathcal{S} : \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i] = \Pr_\eta[\mathcal{P}(x,i) = x_i]$, *where η is the uniform distribution. In other words: the worst case success probability of $\mathcal{S}$ is the same as the average case success probability of $\mathcal{P}$ with uniformly distributed input.*

Proof. This can be achieved by randomizing the input with the help of shared random string. Alice's input can be randomized by XOR-ing it with an n-bit random string r. But Bob's input can be randomized by adding (modulo n) a random number $d \in \{0, \ldots, n-1\}$ to it (assume for now that bits are numbered from 0 to $n-1$). To obtain a consistent strategy, these actions must be identically performed on both sides, thus a shared random string of $n + \log n$ bits[1] is required. Assume that E and D_i are the encoding and decoding functions of the pure strategy $\mathcal{P}$, then the new strategy $\mathcal{S}$ is:

$$E'(x) = E(\mathrm{Shift}_d(x \oplus r)),$$
$$D_i'(b) = D_{i+d \bmod n}(b) \oplus r_i,$$

where $\mathrm{Shift}_d(s)$ substitutes $s_{i+d \bmod n}$ by s_i in string s. Due to input randomization, this strategy will have the same success probability for all inputs (x,i), namely

$$\Pr[\mathcal{S}(x,i) = x_i] = \sum_{x \in \{0,1\}} \sum_{i=0}^{n-1} \frac{1}{2^n \cdot n} \cdot \mathcal{P}(x,i) = \Pr_\eta[\mathcal{P}(x,i) = x_i],$$

that coincides with the average success probability of pure strategy $\mathcal{P}$. $\qquad \square$

Now we will show that inequality (4) becomes equality when $\mu_0 = \eta$, meaning that the uniform distribution η is the "hardest".

---

[1] We will not worry how Bob obtains a uniformly distributed d from a string of random bits when $n \neq 2^k$.

Lemma 2. *The minimum of (2) is reached at uniform distribution η, i.e.,*

$$\min_{\mu} \max_{\mathcal{P}} \Pr_{\mu}[\mathcal{P}(x,i) = x_i] = \max_{\mathcal{P}} \Pr_{\eta}[\mathcal{P}(x,i) = x_i]. \tag{5}$$

Proof. From the previous Lemma we know that there exists a strategy with SR $\mathcal{S}_0$ such that

$$\min_{x,i} \Pr[\mathcal{S}_0(x,i) = x_i] = \max_{\mathcal{P}} \Pr_{\eta}[\mathcal{P}(x,i) = x_i] \tag{6}$$

($\mathcal{S}_0$ is obtained from the best pure strategy by prepending it with input randomization). However, among all strategies with SR there might be one that is better than S_0, thus

$$\max_{\mathcal{S}} \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i] \geq \max_{\mathcal{P}} \Pr_{\eta}[\mathcal{P}(x,i) = x_i]. \tag{7}$$

But if we put $\mu_0 = \eta$ into inequality (4), we obtain

$$\max_{\mathcal{P}} \Pr_{\eta}[\mathcal{P}(x,i) = x_i] \geq \max_{\mathcal{S}} \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i], \tag{8}$$

which is the same as (7), but with reversed sign. It means that both sides are actually equal:

$$\max_{\mathcal{P}} \Pr_{\eta}[\mathcal{P}(x,i) = x_i] = \max_{\mathcal{S}} \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i]. \tag{9}$$

Applying the Yao's principle to the right hand side of (9) we obtain the desired equation (5). $\square$

Theorem 1. *For any pure strategy $\mathcal{P}$:*

$$\Pr_{\eta}[\mathcal{P}(x,i) = x_i] \leq \max_{\mathcal{S}} \min_{x,i} \Pr[\mathcal{S}(x,i) = x_i], \tag{10}$$

with equality if and only if $\mathcal{P}$ is optimal for the uniform distribution η.

Proof. To obtain the required inequality, do not maximize the left hand side of equation (9), but put an arbitrary $\mathcal{P}$. It is obvious that we will obtain equality if and only if $\mathcal{P}$ is optimal. $\square$

This theorem has important consequences – it allows us to consider pure strategies with uniformly distributed input rather than strategies with SR. If we manage to find the optimal pure strategy, then we can also construct an optimal strategy with SR using input randomization[2]. If the pure strategy is not optimal, then we get a lower bound for the strategy with SR.

2.3 Classical $n \mapsto 1$ RAC

Before considering $n \mapsto 1$ QRACs with shared randomness, we will find an optimal classical $n \mapsto 1$ RAC with shared randomness and derive bounds for it.

---

[2]If the encoding function depends only on the Hamming weight of the input string x (e.g., majority function) and the decoding function does not depend on i, there is no need to randomize over i, so n instead of $n + \log n$ shared random bits are enough.

2.3.1 Optimal strategy

According to Theorem 1 we can consider only pure strategies. As a pure strategy is deterministic, for each input it gives either a correct or a wrong answer. To maximize the average success probability we must find a pure strategy that gives correct answer for as many of the $n \cdot 2^n$ inputs as possible – such strategy we will call *optimal pure strategy*.

Let us first consider the problem of finding an optimal decoding strategy, when the encoding strategy is fixed. An encoding function $E : \{0,1\}^n \mapsto \{0,1\}$ divides the set of all strings into two parts:

$$X_0 = \{x \in \{0,1\}^n \mid E(x) = 0\},$$
$$X_1 = \{x \in \{0,1\}^n \mid E(x) = 1\}.$$

If Bob receives bit b, he knows that the initial string was definitely from set X_b, but there is no way for him to tell exactly which string it was. However, if he must recover only the ith bit, he can check, whether there are more zeros or ones among the ith bits of strings from set X_b. More formally, we can introduce the symbol $N_i^b(k)$ that denotes the number of strings from set X_b that have the bit k in ith position:

$$N_i^b(k) = |\{x \in X_b \mid x_i = k\}|, \tag{11}$$

Therefore the optimal decoding strategy $D_i : \{0,1\} \mapsto \{0,1\}$ for the ith bit is:

$$D_i(b) = \begin{cases} 0 & \text{if } N_i^b(0) \geq N_i^b(1), \\ 1 & \text{otherwise.} \end{cases} \tag{12}$$

Of course, if $N_i^b(0) = N_i^b(1)$, Bob can output 1 as well. For pure strategies there are only 4 possible decoding functions for each bit: 0, 1, b, or NOT b. But this is still quite a lot so we will consider the two following lemmas. The first lemma will rule out the *constant decoding functions* 0 and 1.

Lemma 3. *For any n there exists an optimal pure classical $n \mapsto 1$ RAC that does not use constant decoding functions 0 and 1 for any bits.*

Proof. We will show that if there exists an optimal strategy that contains constant decoding functions for some bits, then there also exists an optimal strategy that does not. Let us assume that there is an optimal strategy with constant decoding function 0 for the ith bit (the same argument goes through for 1 as well). Then according to equation (12) we have: $N_i^0(0) \geq N_i^0(1)$ and $N_i^1(0) \geq N_i^1(1)$. Note that $N_i^0(0)+N_i^1(0) = N_i^0(1)+N_i^1(1) = 2^{n-1}$, because $x_i = 0$ in exactly half of all 2^n strings. This means that actually $N_i^0(0) = N_i^0(1)$ and $N_i^1(0) = N_i^1(1)$. If we take a look at (12) again, we see that in such situation any decoding strategy is optimal and we can use any non–constant strategy instead. $\square$

Lemma 4. *For any n there exists an optimal pure classical $n \mapsto 1$ RAC that does not use decoding function NOT b for any bits.*

Proof. We will show that for each pure strategy $\mathcal{P}$ that uses negation as decoding function for the ith bit, there exists a pure strategy $\mathcal{P}'$ with the same average case success probability that does not. If $\mathcal{P}$ consists of encoding function E and

decoding functions D_j, then $\mathcal{P}'$ can be obtained from $\mathcal{P}$ by inverting the ith bit before encoding and after decoding:

$$E'(x) = E(\text{NOT}_i\, x),$$

$$D'_j(b) = \begin{cases} \text{NOT}\, D_j(b) & \text{if } j = i, \\ D_j(b) & \text{otherwise,} \end{cases}$$

where NOT_i inverts the ith bit of string. It is obvious that $\mathcal{P}$ and $\mathcal{P}'$ have the same average success probabilities, because if $\mathcal{P}$ gives correct answer for input (x, i) then $\mathcal{P}'$ gives correct answer for input $(\text{NOT}_i\, x, i)$. The same stands for wrong answers. $\qquad\square$

Theorem 2. *Pure classical $n \mapsto 1$ RAC with identity decoding functions and majority encoding function is optimal.*

Proof. According to Lemma 3 and Lemma 4, there exists an optimal pure classical $n \mapsto 1$ RAC with identity decoding function for all bits. Now we must consider the other part – finding an optimal encoding given a particular (identity) decoding function. It is obvious that in our case optimal encoding must return the majority of bits:

$$E'(x) = \begin{cases} 0 & \text{if } |x| < n/2, \\ 1 & \text{otherwise,} \end{cases}$$

where $|x|$ is the Hamming weight of string x (the number of ones in it). $\qquad\square$

2.3.2 Asymptotic bounds

Let us find the exact value of the average success probability for the optimal pure RAC suggested in Theorem 2. We will separately consider the even and odd cases.

In the odd case ($n = 2m + 1$) the average success probability is given by

$$p(2m + 1) = \frac{1}{(2m+1) \cdot 2^{2m+1}} \left(2 \sum_{i=m+1}^{2m+1} i \binom{2m+1}{i} \right), \qquad (13)$$

where the factor 2 stands for either zeros or ones being the majority, and $\binom{2m+1}{i}$ stands for the number of strings where the given symbol dominates and appears exactly i times.

In the even case ($n = 2m$) there are a lot of strings with the same number of zeros and ones. These strings are bad, because with majority encoding and identity decoding it is not possible to give the correct answer for more than a half of all bits. The corresponding average success probability is given by

$$p(2m) = \frac{1}{2m \cdot 2^{2m}} \left(2 \sum_{i=m+1}^{2m} i \binom{2m}{i} + m \binom{2m}{m} \right), \qquad (14)$$

where the last term stands for the bad strings.

In Appendix A we give a combinatorial interpretation of the sums in (13) and (14). Equations (122) and (123) derived in Appendix A can be used to
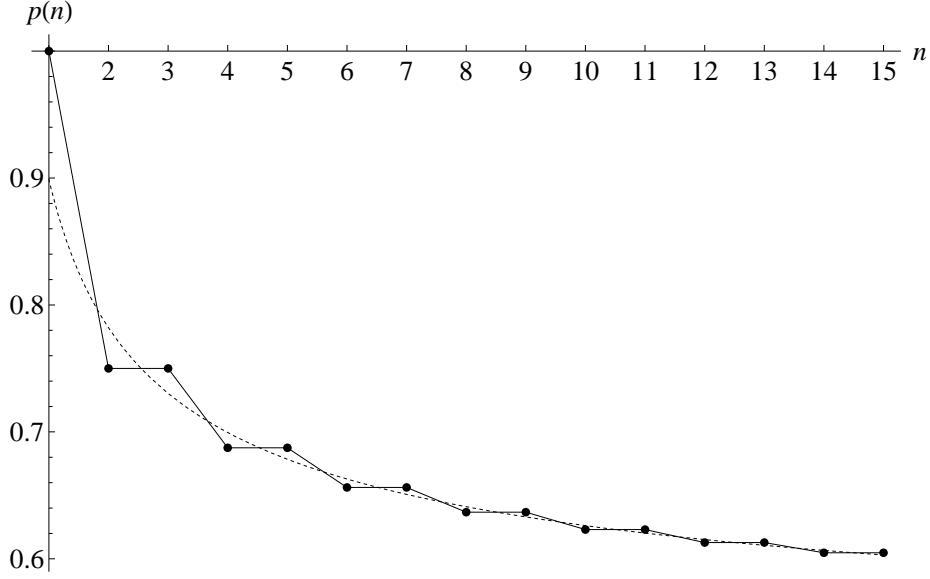
9

Figure 1: Exact probability of success $p(n)$ for optimal pure classical $n \mapsto 1$ RAC (solid line) and its approximate value (dotted line) according to (17).

simplify $p(2m+1)$ and $p(2m)$, respectively. It turns out that both probabilities are equal:

$$p(2m) = p(2m+1) = \frac{1}{2} + \frac{1}{2^{2m+1}}\binom{2m}{m}. \tag{15}$$

One can use (15) to compute the $p(n)$ exactly, by putting $m = \lfloor \frac{n}{2} \rfloor$. We can apply Stirling's approximation [5] $m! \approx \left(\frac{m}{e}\right)^m \sqrt{2\pi m}$ to (15) and obtain

$$p(2m) = p(2m+1) \approx \frac{1}{2} + \frac{1}{2\sqrt{\pi m}}. \tag{16}$$

If we put $m \approx \frac{n}{2}$, then (16) turns to

$$p(n) \approx \frac{1}{2} + \frac{1}{\sqrt{2\pi n}}. \tag{17}$$

Exact probability (15) and its approximation (17) is shown in Fig. 1.

For odd and even cases asymptotic upper and lower bounds of $p(n)$ can be obtained using the following inequality [5]:

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n+1}} < n! < \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}. \tag{18}$$

For odd case we have:

$$\frac{\exp\left(\frac{1}{12n-11} - \frac{2}{6n-6}\right)}{\sqrt{2\pi(n-1)}} < p(n) - \frac{1}{2} < \frac{\exp\left(\frac{1}{12n-12} - \frac{2}{6n-5}\right)}{\sqrt{2\pi(n-1)}}, \tag{19}$$
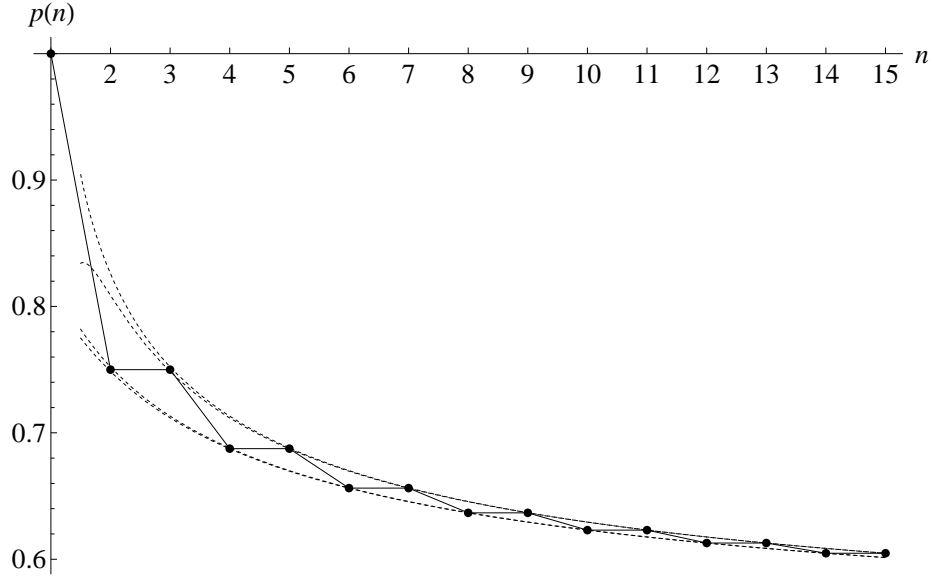
10

Figure 2: Dotted lines show upper and lower bounds of $p(n)$ for odd and even n according to inequalities (19) and (20).

but for even:

$$\frac{\exp\left(\frac{1}{12n} - \frac{2}{6n+1}\right)}{\sqrt{2\pi n}} < p(n) - \frac{1}{2} < \frac{\exp\left(\frac{1}{12n+1} - \frac{2}{6n}\right)}{\sqrt{2\pi n}}. \tag{20}$$

All four bounds are shown in Fig. 2.

3   Quantum random access codes

3.1   Visualizing a qubit

When dealing with quantum random access codes (at least it the qubit case), it is a good idea to try to visualize them. We provide two ways.

3.1.1   Bloch sphere representation

A *pure qubit state* is a column vector $|\psi\rangle \in \mathbb{C}^2$. It can be expressed as a linear combination $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$, where $|0\rangle = \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)$ and $|1\rangle = \left(\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right)$. The coefficients $\alpha, \beta \in \mathbb{C}$ must satisfy $|\alpha|^2 + |\beta|^2 = 1$. Since the physical state is not affected by the *phase factor* (i.e., $|\psi\rangle$ and $e^{i\phi} |\psi\rangle$ are the same states for any $\phi \in \mathbb{R}$), without the loss of generality one can write

$$|\psi\rangle = \begin{pmatrix} \cos\frac{\theta}{2} \\ e^{i\varphi} \sin\frac{\theta}{2} \end{pmatrix}, \tag{21}$$

where $0 \leq \theta \leq \pi$ and $0 \leq \varphi < 2\pi$ (the factor $1/2$ for θ in (21) is chosen so that these ranges resemble the ones for *spherical coordinates* in $\mathbb{R}^3$).
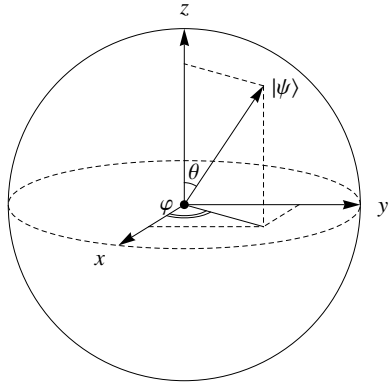
11

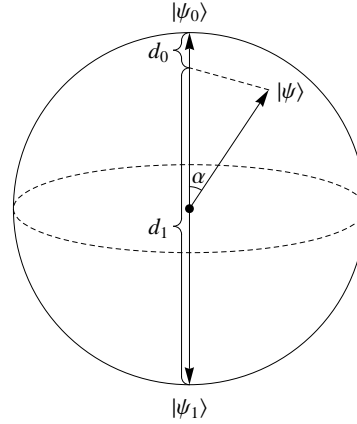Figure 3: Angles θ and φ of the Bloch vector corresponding to state $|\psi\rangle$.

Figure 4: Geometric interpretation of orthogonal measurement.

For almost all states $|\psi\rangle$ there is a unique way to assign the parameters θ and φ. The only exceptions are states $|0\rangle$ and $|1\rangle$, that correspond to $\theta = 0$ and $\theta = \pi$, respectively. In both cases φ does not affect the physical state. Note that the spherical coordinates with *latitude* θ and *longitude* φ have the same property, namely – the longitude is not defined at poles. This suggests that the state space of a single qubit is topologically a sphere.

Indeed, there is a one-to-one correspondence between pure qubit states and the points on a unit sphere in $\mathbb{R}^3$. This is called the *Bloch sphere representation* of a qubit state. The *Bloch vector* for state (21) is $\boldsymbol{r} = (x, y, z)$, where the coordinates (see Fig. 3) are given by

$$\begin{cases} x = \sin\theta\cos\varphi, \\ y = \sin\theta\sin\varphi, \\ z = \cos\theta. \end{cases} \tag{22}$$

Given the Bloch vector $\boldsymbol{r} = (x, y, z)$, the coefficients of the corresponding state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ can be found as follows [4, pp. 102]:

$$\alpha = \sqrt{\frac{z+1}{2}}, \quad \beta = \frac{x+iy}{\sqrt{2(z+1)}} \tag{23}$$

with the convention that $(0, 0, -1)$ corresponds to $\alpha = 0$ and $\beta = 1$.

The *density matrix* of a state $|\psi\rangle$ is defined as $\rho = |\psi\rangle\langle\psi|$. For the state $|\psi\rangle$ in (21) we have:

$$\rho = \frac{1}{2}\begin{pmatrix} 1+\cos\theta & e^{-i\varphi}\sin\theta \\ e^{i\varphi}\sin\theta & 1-\cos\theta \end{pmatrix} = \frac{1}{2}\left(I + x\sigma_x + y\sigma_y + z\sigma_z\right), \tag{24}$$

where (x, y, z) are the coordinates of the Bloch vector $\boldsymbol{r}$ given in (22) and

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{25}$$

are called *Pauli matrices*. We can write (24) more concisely as

$$\rho = \frac{1}{2}\left(I + \boldsymbol{r} \cdot \boldsymbol{\sigma}\right) \tag{26}$$

where $\boldsymbol{r} = (x, y, z)$ and $\boldsymbol{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$.

If $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are the Bloch vectors of two pure states $|\psi_1\rangle$ and $|\psi_2\rangle$, then

$$|\langle\psi_1|\psi_2\rangle|^2 = \mathrm{Tr}(\rho_1\rho_2) = \frac{1}{2}(1 + \boldsymbol{r}_1 \cdot \boldsymbol{r}_2). \tag{27}$$

This relates the inner product in $\mathbb{C}^2$ to the one in $\mathbb{R}^3$. Since $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are unit vectors, $\boldsymbol{r}_1 \cdot \boldsymbol{r}_2 = \cos\alpha$, where α is the angle between $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$.

In qubit case an *orthogonal measurement* M can be specified by a set of two orthonormal states: $M = \{|\psi_0\rangle, |\psi_1\rangle\}$. Orthonormality means that $\langle\psi_i|\psi_j\rangle = \delta_{ij}$. If we measure a qubit that is in state $|\psi\rangle$ with measurement M then the *outcome* will be either 0 or 1 and the state will collapse to $|\psi_0\rangle$ or $|\psi_1\rangle$ with probabilities $|\langle\psi_0|\psi\rangle|^2$ and $|\langle\psi_1|\psi\rangle|^2$, respectively. Observe that for orthogonal states equation (27) implies $\boldsymbol{r}_1 \cdot \boldsymbol{r}_2 = -1$, therefore they correspond to antipodal points on Bloch sphere. If we denote the angle between the Bloch vectors of $|\psi\rangle$ and $|\psi_0\rangle$ by α, then according to (27) the probabilities of outcomes are

$$\begin{cases} p_0 = \dfrac{1}{2}(1 + \cos\alpha), \\[2mm] p_1 = \dfrac{1}{2}(1 - \cos\alpha). \end{cases} \tag{28}$$

There is a nice geometrical interpretation of these probabilities. If we project the Bloch vector corresponding to $|\psi\rangle$ on the axes spanned by the Bloch vectors of $|\psi_0\rangle$ and $|\psi_1\rangle$ (see Fig. 4), then $p_0 = d_1/2$ and $p_1 = d_0/2$ (note the different indices), where d_0 is the distance between the projection and $|\psi_0\rangle$, but d_1 – between the projection and $|\psi_1\rangle$. Observe that vectors on the upper hemisphere have greater probability to collapse to $|\psi_0\rangle$, but on lower hemisphere – to $|\psi_1\rangle$. On the equator both probabilities are equal to $\frac{1}{2}$.

3.1.2 Unit disk representation

There is another way of visualizing a qubit. Unlike the Bloch sphere representation, this way of representing a qubit is not known to be used elsewhere. The idea is to use only one complex number to specify a pure qubit state $|\psi\rangle = \left(\begin{smallmatrix} \alpha \\ \beta \end{smallmatrix}\right) \in \mathbb{C}^2$. It is possible, since $|\psi\rangle$ can be written in the form (21) that is completely determined by its second component

$$\beta = e^{i\varphi}\sin\frac{\theta}{2}.$$

The first component is just $\sqrt{1 - |\beta|^2} = \alpha$. As $|\beta| \leq 1$, the set of all possible qubit states can be identified with a unit disk in the complex plane (the polar coordinates assigned to $|\psi\rangle$ are (r, φ), where $r = \sin\frac{\theta}{2}$). The origin corresponds to $|\psi\rangle = |0\rangle$, but all points on the unit circle $|\beta| = 1$ are identified with $|\psi\rangle = |1\rangle$. This corresponds to puncturing the Bloch sphere at its South pole and flattening it to a unit disk.
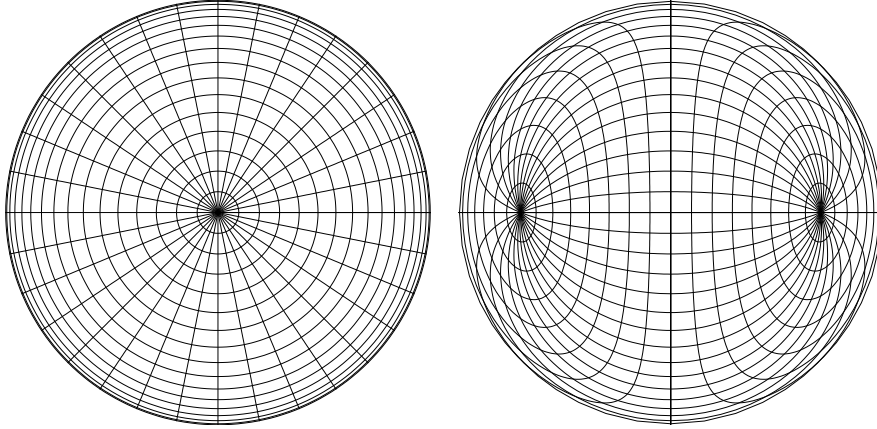
Figure 5: Hadamard transformation in the unit disk representation.

Example. Let us consider the action of the *Hadamard gate* $H = \frac{1}{\sqrt{2}} \left(\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix} \right)$ in the unit disk representation. It acts on the basis states as follows:

$$H \left| 0 \right\rangle = \frac{\left| 0 \right\rangle + \left| 1 \right\rangle}{\sqrt{2}} = \left| + \right\rangle, \quad H \left| 1 \right\rangle = \frac{\left| 0 \right\rangle - \left| 1 \right\rangle}{\sqrt{2}} = \left| - \right\rangle.$$

The way it transforms the curves of constant θ and φ is shown in Fig. 5. After this transformation the origin corresponds to $\left| + \right\rangle$, but the unit circle to $\left| - \right\rangle$ state. The states $\left| 1 \right\rangle$ and $\left| 0 \right\rangle$ correspond to the "left pole" and "right pole", respectively.

We will use this representation in Sect. 3.3 and Sect. 4.1 to describe the qubit states whose Bloch vectors are the vertices of certain polyhedra. It is surprising that the values of β for these states are the roots of certain polynomials with integer coefficients.

3.2 Types of quantum encoding–decoding strategies

Let us now consider the quantum analogue of a pure strategy.

Definition. A *pure quantum $n \mapsto 1$ encoding–decoding strategy* is an ordered tuple $(E, M_1, \ldots, M_n)$ that consists of encoding function $E : \{0,1\}^n \mapsto \mathbb{C}^2$ and n orthogonal measurements: $M_i = \left\{ \left| \psi_0^i \right\rangle, \left| \psi_1^i \right\rangle \right\}$.

If Alice encodes the string x with function E, she obtains a pure qubit state $\left| \psi \right\rangle = E(x)$. When Bob receives $\left| \psi \right\rangle$ and is asked to recover the ith bit of x, he performs the measurement M_i. The probability that Bob recovers x_i correctly is equal to

$$p(x, i) = \left| \left\langle \psi_{x_i}^i \middle| \psi \right\rangle \right|^2. \tag{29}$$

As in the classical setting, we can allow Alice and Bob to have probabilistic quantum strategies without cooperation. Though we will not need it, the mixed quantum strategy can be defined in complete analogy with the classical one.

14

Definition. A *mixed quantum $n \mapsto 1$ encoding–decoding strategy* is an ordered tuple $(P_E, P_{M_1}, \ldots, P_{M_n})$ of probability distributions. P_E is a distribution over encoding functions E and P_{M_i} are probability distributions over orthogonal measurements of qubit.

The main object of our research is the quantum strategy with cooperation, i.e., with shared randomness. It is defined in complete analogy with the classical one.

Definition. A *quantum $n \mapsto 1$ encoding–decoding strategy with shared randomness* is a probability distribution over pure quantum strategies.

We would like to point out two very important things about the quantum strategy with shared randomness. The first thing is that all statements about classical strategies with SR in Sect. 2.2 are valid for quantum strategies as well (the only difference is that *"pure strategy"* now means *"pure quantum strategy"* instead of *"pure classical strategy"* and *"strategy with SR"* means *"quantum strategy with SR"* instead of *"classical strategy with SR"*). The most important consequence of this observation is that Theorem 1 is valid also for quantum strategies with SR. This means that the same technique of obtaining the upper bound can be used in the quantum case, i.e., we can consider the average success probability of a pure quantum strategy instead of the worst case success probability of the quantum strategy with SR.

The second thing is that the quantum strategy with SR is the most powerful quantum encoding–decoding strategy if all kinds of classical randomness are allowed (though it is not the most general one). This issue is discussed in Sect. 3.6 and Appendix B.

3.3 Known quantum RACs

In [1] it has been shown that for $2 \mapsto 1$ classical RACs in the mixed setting the decoding party cannot do better than guessing, i.e., the worst case success probability cannot exceed $1/2$. However, if quantum states can be transmitted, there are pure quantum $2 \mapsto 1$ and $3 \mapsto 1$ schemes [1]. This clearly indicates the advantages of quantum RACs. A quantum $4 \mapsto 1$ scheme cannot exist [2]. We will review these results in the next three sections.

3.3.1 The $2 \mapsto 1$ QRAC

The $2 \mapsto 1$ QRAC was first introduced in [1]. The main idea is to use two mutually orthogonal pairs of antipodal Bloch vectors for measurement bases. For example, let M_1 and M_2 be the measurements along the x and y axes, respectively. The corresponding Bloch vectors are $\boldsymbol{v}_1 = (\pm 1, 0, 0)$ and $\boldsymbol{v}_2 = (0, \pm 1, 0)$. The measurement bases are:

$$M_1 = \left\{ \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right\}, \tag{30}$$

$$M_2 = \left\{ \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ i \end{pmatrix}, \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -i \end{pmatrix} \right\}. \tag{31}$$

The planes orthogonal to the x and y axes cut the Bloch sphere into four parts. Note that in each part only one definite string can be encoded (otherwise the
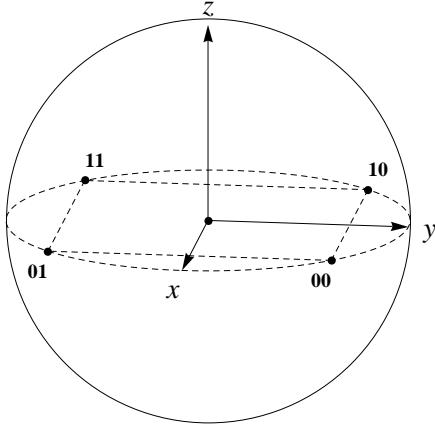
Figure 6: Bloch sphere representation of encoding for $2 \mapsto 1$ quantum random access code.
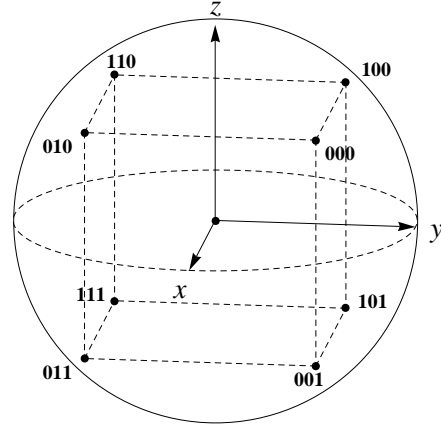
Figure 7: Bloch sphere representation of encoding for $3 \mapsto 1$ quantum random access code.

worst case success probability will be less than $\frac{1}{2}$). According to equations in (28) all encoding points must be as far from both planes as possible in order to maximize the worst case success probability (recall the geometrical interpretation of the measurement shown in Fig. 4). In our case the best encoding states are the vertices of a square $\frac{1}{\sqrt{2}}(\pm 1, \pm 1, 0)$ inscribed in the unit circle on the xy plane (see Fig. 6). Given a string $x = x_1 x_2$, the Bloch vector of the encoding state can be found as follows:

$$\boldsymbol{r}(x) = \frac{1}{\sqrt{2}} \begin{pmatrix} (-1)^{x_1} \\ (-1)^{x_2} \\ 0 \end{pmatrix}. \tag{32}$$

The corresponding encoding function is:

$$E(x_1, x_2) = \frac{1}{\sqrt{2}} |0\rangle + \frac{(-1)^{x_1} + i(-1)^{x_2}}{2} |1\rangle. \tag{33}$$

The success probability is the same for all input strings and all bits to be recovered:

$$p = \frac{1}{2} \left(1 + \cos \frac{\pi}{4} \right) = \frac{1}{2} + \frac{1}{2\sqrt{2}} \approx 0.8535533906. \tag{34}$$

3.3.2 The $3 \mapsto 1$ QRAC

It is not hard to generalize the $2 \mapsto 1$ QRAC to a $3 \mapsto 1$ code – just take three mutually orthogonal pairs of antipodal Bloch vectors, i.e., the vertices of an *octahedron*. The third pair is $\boldsymbol{v}_3 = (0, 0, \pm 1)$ and the corresponding measurement basis is:

$$M_3 = \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\}. \tag{35}$$

In this case we have three orthogonal planes that cut the sphere into eight parts and only one string can be encoded into each part. In this case the optimal

encoding states correspond to the vertices of a cube $\frac{1}{\sqrt{3}}(\pm 1, \pm 1, \pm 1)$ inscribed in the Bloch sphere (see Fig. 7). The Bloch vector of the encoding state of string $x = x_1 x_2 x_3$ is:

$$\boldsymbol{r}(x) = \frac{1}{\sqrt{3}} \begin{pmatrix} (-1)^{x_1} \\ (-1)^{x_2} \\ (-1)^{x_2} \end{pmatrix}. \tag{36}$$

The corresponding encoding function is $E(x_1, x_2, x_3) = \alpha \left|0\right\rangle + \beta \left|1\right\rangle$ with coefficients α and β explicitly given by

$$\begin{cases} \alpha = \sqrt{\dfrac{1}{2} + \dfrac{(-1)^{x_3}}{2\sqrt{3}}}, \\ \beta = \dfrac{(-1)^{x_1} + i(-1)^{x_2}}{\sqrt{6 + 2\sqrt{3}(-1)^{x_3}}}. \end{cases} \tag{37}$$

In fact, coefficients β are exactly the eight roots of the polynomial

$$36\beta^8 + 24\beta^4 + 1 \tag{38}$$

(recall the unit disk representation in Sect. 3.1.2). This code also has the same success probability in all cases:

$$p = \frac{1}{2} + \frac{1}{2\sqrt{3}} \approx 0.7886751346. \tag{39}$$

3.3.3 Impossibility of the $4 \mapsto 1$ QRAC

It has been shown that $2 \mapsto 1$ and $3 \mapsto 1$ codes discussed above cannot be generalized for 4 (and hence more) encoded bits [2]. The reason is simple – it is not possible to cut the Bloch sphere into 16 parts with 4 great circles. Thus the number of strings will exceed the number of parts, hence at least two strings must be encoded in the same part. This will decrease the worst case success probability below $\frac{1}{2}$.

Let us consider how many parts can be obtained by cutting a sphere with 4 great circles. Without loss of generality we can assume that the first great circle coincides with the equator. We use the *gnomonic projection* (from the center of the sphere) to project the remaining three circles to a plane tangent to the south pole. Note that great circles are transformed into lines and vice versa, thus we will obtain three lines. Also note that each region in the plane corresponds to two (diametrically opposite) regions on the sphere. It is simple to verify that three lines cannot cut the plane into more than 7 parts. Thus the sphere cannot be cut into more than 14 parts with four great circles (an example achieving the upper bound is shown in Figs. 22 and 23).

In general, if we have n great circles, the maximal number of parts we can obtain is twice what we can obtain by cutting the plane with $n - 1$ lines. If each line we draw intersects all previous lines and no three lines intersect at the same point, we get $n(n - 1) + 2$ parts.

3.4 Optimal encoding for given decoding strategy

An orthogonal measurement for the ith bit is specified by antipodal points on the Bloch sphere: $M_i = \{\boldsymbol{v}_i, -\boldsymbol{v}_i\}$. Let $\boldsymbol{r}_x$ be the Bloch vector that corresponds

to the quantum state in which string $x \in \{0,1\}^n$ is encoded. According to equations in (28) the success probability for input (x,i) is

$$p(x,i) = \frac{1}{2}\big(1 + (-1)^{x_i}\boldsymbol{v}_i \cdot \boldsymbol{r}_x\big) \tag{40}$$

and the average success probability is given by

$$
\begin{aligned}
p &= \frac{1}{2^n \cdot n} \sum_{x \in \{0,1\}^n} \sum_{i=1}^{n} \frac{1}{2}\big(1 + (-1)^{x_i}\boldsymbol{v}_i \cdot \boldsymbol{r}_x\big) \\
&= \frac{1}{2}\left(1 + \frac{1}{2^n \cdot n} \underbrace{\sum_{x \in \{0,1\}^n} \boldsymbol{r}_x \cdot \sum_{i=1}^{n}(-1)^{x_i}\boldsymbol{v}_i}_{S_{\boldsymbol{v},\boldsymbol{r}}}\right).
\end{aligned}
\tag{41}
$$

In order to maximize the probability p, we only need to maximize $S_{\boldsymbol{v},\boldsymbol{r}}$ in equation (41) over all possible measurements $\boldsymbol{v}_i$ and encodings $\boldsymbol{r}_x$ (in total $n + 2^n$ unit vectors in $\mathbb{R}^3$). We will denote the maximum of $S_{\boldsymbol{v},\boldsymbol{r}}$ by $S(n)$:

$$S(n) = \max_{\{\boldsymbol{v}_i\}_i, \{\boldsymbol{r}_x\}_x} S_{\boldsymbol{v},\boldsymbol{r}} = \max_{\{\boldsymbol{v}_i\}_i} \sum_{x \in \{0,1\}^n} \max_{\boldsymbol{r}_x} \boldsymbol{r}_x \cdot \sum_{i=1}^{n}(-1)^{x_i}\boldsymbol{v}_i. \tag{42}$$

If we define

$$\boldsymbol{v}_x = \sum_{i=1}^{n}(-1)^{x_i}\boldsymbol{v}_i, \tag{43}$$

then it is obvious that the scalar product $\boldsymbol{r}_x \cdot \boldsymbol{v}_x$ in (42) will be maximized when $\boldsymbol{r}_x$ is chosen along the same direction as $\boldsymbol{v}_x$, i.e. $\boldsymbol{r}_x = \boldsymbol{v}_x / \|\boldsymbol{v}_x\|$ when $\|\boldsymbol{v}_x\| \neq 0$. In this case we have $\boldsymbol{r}_x \cdot \boldsymbol{v}_x = \|\boldsymbol{v}_x\|$ and

$$S(n) = \max_{\{\boldsymbol{v}_i\}_i} \sum_{x \in \{0,1\}^n} \left\| \sum_{i=1}^{n}(-1)^{x_i}\boldsymbol{v}_i \right\|. \tag{44}$$

Therefore we only need to maximize over all possible measurements succinctly represented by n unit vectors $\boldsymbol{v}_i \in \mathbb{R}^3$, because the optimal encoding is already determined by measurements (see Sect. 4.1 for some numerical results obtained in this way). When the value of $S(n)$ is found, then according to (41) the corresponding probability is:

$$p(n) = \frac{1}{2}\left(1 + \frac{S(n)}{2^n \cdot n}\right). \tag{45}$$

We can observe a connection between quantum and classical RACs with SR. Assume that Alice and Bob have to implement $n \mapsto 1$ QRAC with SR and are deciding what strategies to use – Bob is responsible for choosing the measurements, but Alice has to choose how to encode the input string. Once they have decided, they have to follow the agreement and cannot cheat. Unfortunately, Bob is corrupt and wants to propose the worst measurements – measuring all bits in the same basis. Luckily Alice is clever enough to choose the optimal encoding for Bob's measurements. According to the discussion above, she has

to use the majority encoding function. Thus the obtained QRAC is as good as an optimal classical RAC discussed in Sect. 2.3.1, Theorem 2.

It looks plausible that Bob cannot do worse than to use the same measurement for all bits. However, we have not proved it. Thus we can state this observation as a conjecture:

Conjecture. *For any measurements there is an encoding such that the obtained $n \mapsto 1$ quantum RAC with SR is at least as good as optimal $n \mapsto 1$ classical one.*

3.5 Upper bound

In this section we will derive an upper bound for $S(n)$. For this purpose we rewrite the equation (44) in the following form:

$$S(n) = \max_{\{\boldsymbol{v}_i\}_i} S_{\boldsymbol{v}} \tag{46}$$

where

$$S_{\boldsymbol{v}} = \sum_{a \in \{1,-1\}^n} \left\| \sum_{i=1}^{n} a_i \boldsymbol{v}_i \right\| \tag{47}$$

(for convenience we take the sum over the set $\{1, -1\}^n$ instead of $\{0, 1\}^n$).

Lemma 5. *For any unit vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ we have:*

$$\sum_{a_1,\ldots,a_n \in \{1,-1\}} \| a_1 \boldsymbol{v}_1 + \cdots + a_n \boldsymbol{v}_n \|^2 = n \cdot 2^n. \tag{48}$$

Proof. For $n = 1$ we have:

$$\sum_{a_1 \in \{1,-1\}} \| a_1 \boldsymbol{v}_1 \|^2 = \| \boldsymbol{v}_1 \|^2 + \| -\boldsymbol{v}_1 \|^2 = 2.$$

Let us assume that equation (48) holds for $n = k$. Then for $n = k + 1$ we have

$$\sum_{a_1,\ldots,a_k,a_{k+1} \in \{1,-1\}} \| a_1 \boldsymbol{v}_1 + \cdots + a_k \boldsymbol{v}_k + a_{k+1} \boldsymbol{v}_{k+1} \|^2.$$

If we write out the sum over a_{k+1} explicitly, we obtain

$$\sum_{a_1,\ldots,a_k \in \{1,-1\}} \left(\| a_1 \boldsymbol{v}_1 + \cdots + a_k \boldsymbol{v}_k + \boldsymbol{v}_{k+1} \|^2 + \| a_1 \boldsymbol{v}_1 + \cdots + a_k \boldsymbol{v}_k - \boldsymbol{v}_{k+1} \|^2 \right).$$

We can use the parallelogram identity

$$\| \boldsymbol{u}_1 + \boldsymbol{u}_2 \|^2 + \| \boldsymbol{u}_1 - \boldsymbol{u}_2 \|^2 = 2 \left(\| \boldsymbol{u}_1 \|^2 + \| \boldsymbol{u}_2 \|^2 \right),$$

that holds for any two vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$, to simplify the sum as follows:

$$\sum_{a_1,\ldots,a_k \in \{1,-1\}} 2 \left(\| a_1 \boldsymbol{v}_1 + \cdots + a_k \boldsymbol{v}_k \|^2 + \| \boldsymbol{v}_{k+1} \|^2 \right). \tag{49}$$

We know that $\boldsymbol{v}_{k+1}$ is a unit vector and we have assumed that (48) holds for $n = k$, therefore (49) simplifies to: $2 \left(k \cdot 2^k + 2^k \right) = (k+1) \cdot 2^{k+1}$. $\quad\square$

We will use the previous lemma to obtain an upper bound for $S_{\boldsymbol{v}}^2$ defined in (47). According to (46) this will give us an upper bound for $S(n)$ as well.

Lemma 6. *For any set of unit vectors $\{\boldsymbol{v}_i\}_{i=1}^n$ inequality $S_{\boldsymbol{v}} \le \sqrt{n} \cdot 2^n$ holds.*

Proof. According to (47) we have:

$$S_{\boldsymbol{v}}^2 \;=\; \sum_{a,a' \in \{1,-1\}^n} \left\| \sum_{i=1}^n a_i \boldsymbol{v}_i \right\| \cdot \left\| \sum_{i=1}^n a_i' \boldsymbol{v}_i \right\|. \tag{50}$$

For any real x and y we have $(x-y)^2 \ge 0$. It means $xy \le \frac{1}{2}(x^2 + y^2)$. If we apply this inequality to each term in (50), we obtain:

$$S_{\boldsymbol{v}}^2 \;\le\; \sum_{a,a' \in \{1,-1\}^n} \frac{1}{2} \left(\left\| \sum_{i=1}^n a_i \boldsymbol{v}_i \right\|^2 + \left\| \sum_{i=1}^n a_i' \boldsymbol{v}_i \right\|^2 \right). \tag{51}$$

We can replace the double sum in (51) with two single sums, because the first norm depends only on a, but the second only on a':

$$S_{\boldsymbol{v}}^2 \;\le\; 2^n \, \frac{1}{2} \sum_{a \in \{1,-1\}^n} \left\| \sum_{i=1}^n a_i \boldsymbol{v}_i \right\|^2 \;+\; 2^n \, \frac{1}{2} \sum_{a' \in \{1,-1\}^n} \left\| \sum_{i=1}^n a_i' \boldsymbol{v}_i \right\|^2. \tag{52}$$

If we apply Lemma 5 to the sums over a and a', we obtain

$$S_{\boldsymbol{v}}^2 \le 2^n \frac{1}{2}(n \cdot 2^n) + 2^n \frac{1}{2}(n \cdot 2^n) = n \cdot 2^{2n}. \tag{53}$$

After taking the square root from both sides, we get $S_{\boldsymbol{v}} \le \sqrt{n} \cdot 2^n$. $\qquad\square$

Theorem 3. *For any $n \overset{p}{\mapsto} 1$ QRAC with shared randomness: $p \le \frac{1}{2} + \frac{1}{2\sqrt{n}}$.*

Proof. From Lemma 6 we have $S_{\boldsymbol{v}} \le \sqrt{n} \cdot 2^n$. From equation (46) we see that the same upper bound applies to $S(n)$. Putting it into (45) we get:

$$p \le \frac{1}{2} + \frac{1}{2\sqrt{n}}. \qquad\square$$

In particular, this means that the known $2 \mapsto 1$ and $3 \mapsto 1$ QRACs discussed in Sect. 3.3 cannot be improved even if shared randomness is allowed.

The intuition behind this upper bound is as follows. If instead of $\mathbb{R}^3$ the Bloch vector of a qubit state would be in $\mathbb{R}^n$, we could choose all n measurements to be orthogonal. Vectors forming these measurement bases would be the vertices of the *cross polytope*, i.e., all permutations of $(\pm 1, 0, \ldots, 0)$, and the optimal encoding would be the vertices of a *hypercube*, i.e., points $(\pm 1, \pm 1, \ldots, \pm 1)$. Then all terms in equation (47) would be equal to $\sqrt{n}$ and sum to $2^n \sqrt{n}$, thus the probability (45) would be equal to $\frac{1}{2}(1 + \frac{2^n \sqrt{n}}{2^n n}) = \frac{1}{2} + \frac{1}{2\sqrt{n}}$. Since we have only three dimensions, the actual probability should be smaller.

3.6 General upper bound

Let us prove an analogue of Theorem 3 for a more general model, because quantum mechanics allows us to consider more general quantum states and measurements. Namely, Alice can encode her message into a *mixed state* instead of a pure state and Bob can use a POVM measurement instead of an orthogonal measurement to recover information. A mixed state is just a probability distribution over pure states therefore it does not provide a more general encoding model. In contrast, a POVM measurement provides a more general decoding model. In Appendix B we show that in the qubit case a POVM can be replaced by a probability distribution over orthogonal measurements and *constant decoding functions* (0 or 1). This suggests that the definition of the pure quantum encoding–decoding strategy given in Sect. 3.2 should be extended allowing constant decoding functions as well. In fact, there is another reason to extend the definition.

Example. It is not possible to construct a pure QRAC (as defined in Sect. 3.2) that simulates the following pure classical $2 \mapsto 1$ RAC:

- *encoding:* encode the first bit,

- *decoding:* if the first bit is asked, replay the received bit, if the second one is asked – say 0 no matter what is received.

To recover the first bit with certainty, Alice and Bob have to agree on two antipodal points on the Bloch sphere, where the information is encoded. Unfortunately the second bit will cause a problem – it is not possible to choose an orthogonal measurement of a qubit in an unknown state, so that the result is always the same.

To resolve this problem, we must allow Bob either to perform a measurement or to use a constant decoding function.

Definition. An *enhanced orthogonal measurement* is either an orthogonal measurement or one that always gives the same answer.

Definition. An *enhanced pure quantum $n \mapsto 1$ encoding–decoding strategy* is an ordered tuple $(E, M_1, \ldots, M_n)$ consisting of encoding function $E : \{0,1\}^n \mapsto \mathbb{C}^2$ and n decoding functions M_i that are enhanced orthogonal measurements.

Definition. An *enhanced quantum encoding–decoding strategy with SR* is a probability distribution over enhanced pure quantum strategies.

Now it is straightforward to construct a pure quantum RAC for the previous example. In fact, now any classical RAC (either pure, mixed or with SR) can be simulated by the corresponding type of a quantum RAC.

There is no need to further extend the model of enhanced QRACs with SR by adding some other types of classical randomness. For example, a probabilistic combination of POVMs does not provide a more general measurement, because it can be simulated by a probabilistic combination of enhanced orthogonal measurements. The same stands for a probabilistic post-processing of the measurement result (it can be simulated by a probabilistic combination of enhanced orthogonal measurements as shown in Appendix B). Therefore the

enhanced QRACs with SR is the most general model when all kinds of classical randomness are allowed.

It might be possible that the upper bound obtained in Theorem 3 does not hold for this model, but this is not the case.

Theorem 4. *For $n \overset{p}{\mapsto} 1$ QRAC with any classical randomness: $p \leq \frac{1}{2} + \frac{1}{2\sqrt{n}}$.*

Proof. As discussed above, the enhanced QRACs with SR is the most general model. According to Yao's principle and Theorem 1, we can consider the average success probability of pure enhanced QRACs instead. It suffices to show that the constant decoding functions can be ruled out. More precisely – that QRACs having a constant decoding function for some bit give a smaller upper bound than those without it. In fact, we have to prove a quantum analogue of Lemma 3 in Sect. 2.3.1.

We will use induction on n. The case $n = 1$ is trivial – a pure enhanced QRAC with a constant decoding function has average success probability $\frac{1}{2} \leq 1$. Let us assume that for some $n = k - 1 \geq 1$ the constant decoding functions do not give any benefit. We now prove that the same holds for $n = k$. Let us assume that the constant decoding function 0 is used for the kth bit. The average case success probability is

$$p(k) = \frac{1}{2^k \cdot k} \sum_{x \in \{0,1\}^k} \left(\sum_{i=1}^{k-1} p(x, i) + \delta_{0, x_k} \right), \tag{54}$$

where $p(x, i)$ is the success probability (29) for the input (x, i) where $i \leq k - 1$ and δ_{0, x_k} is the probability that the decoding function 0 gives a correct answer for the kth bit. The last bit can be ignored during the encoding and decoding of other bits:

$$p(k) = \left(\frac{1}{2^k \cdot k} \sum_{x \in \{0,1\}^{k-1}} 2 \sum_{i=1}^{k-1} p(x, i) \right) + \frac{1}{2k} \tag{55}$$

$$= \frac{k-1}{k} \left(\frac{1}{2^{k-1} \cdot (k-1)} \sum_{x \in \{0,1\}^{k-1}} \sum_{i=1}^{k-1} p(x, i) \right) + \frac{1}{2k}. \tag{56}$$

Note that the bracketed expression in (56) is the success probability $p(k-1)$ of a shorter QRAC. Therefore

$$p(k) = \frac{k-1}{k} \cdot p(k-1) + \frac{1}{2k}. \tag{57}$$

Now we can apply the inductive hypothesis:

$$p(k) \leq \frac{k-1}{k} \left(\frac{1}{2} + \frac{1}{2\sqrt{k-1}} \right) + \frac{1}{2k} = \frac{1}{2} + \frac{\sqrt{k-1}}{2k} < \frac{1}{2} + \frac{1}{2\sqrt{k}}, \tag{58}$$

completing the proof. Thus the upper bound obtained in Theorem 3 holds for the general model as well. □

Observe that for $n = 2$ and $n = 3$ this upper bound matches equations (34) and (39), respectively. It means, the known $2 \mapsto 1$ and $3 \mapsto 1$ quantum random access codes with pure encoding–decoding strategies (see Sects. 3.3.1 and 3.3.2, respectively) are optimal even if all kinds of randomness are allowed. For $n = 4$ we get $p \leq \frac{3}{4}$. We now turn to lower bound for p.

3.7   Lower bound

A lower bound for QRACs with shared randomness can be obtained by randomized encoding. Alice and Bob can use the shared random string to agree on some random orthogonal measurement for each bit. Each of these measurement bases can be specified by antipodal points on the Bloch sphere (see Sect. 3.1.1). These points can be sampled by using some sphere point picking method [6], near uniformly given enough shared randomness. The chosen measurements determine the optimal encoding scheme (see Sect. 3.4) which is known to both sides.

The expected success probability of randomized $n \mapsto 1$ QRAC similarly to (45) is given by

$$\mathbb{E}(p) = \frac{1}{2}\left(1 + \frac{1}{2^n \cdot n} \mathop{\mathbb{E}}_{\{v_i\}_i}(S_v) \right) \tag{59}$$

where according to equation (47)

$$\mathop{\mathbb{E}}_{\{v_i\}_i}(S_v) = \mathop{\mathbb{E}}_{\{v_i\}_i}\left(\sum_{x \in \{0,1\}^n} \left\| \sum_{i=1}^n (-1)^{x_i} v_i \right\| \right) = \sum_{x \in \{0,1\}^n} \mathop{\mathbb{E}}_{\{v_i\}_i}\left\| \sum_{i=1}^n (-1)^{x_i} v_i \right\|.$$

Each string $x \in \{0,1\}^n$ influences the direction of some vectors v_i, but the distribution of resultant vectors is still uniform. Therefore we have:

$$\mathop{\mathbb{E}}_{\{v_i\}_i}(S_v) = 2^n \mathop{\mathbb{E}}_{\{v_i\}_i}\left\| \sum_{i=1}^n v_i \right\|. \tag{60}$$

This expression has a very nice geometrical interpretation – it is the average distance traveled by a particle that performs n steps of unit length each in a random direction.

Theorem 5 (by S.Chandrasekhar [7, pp. 14]). *The probability density to arrive at point $\boldsymbol{R}$ after performing $n \gg 1$ steps of random walk is*

$$W(\boldsymbol{R}) = \left(\frac{3}{2\pi n} \right)^{3/2} \exp\left(-\frac{3\|\boldsymbol{R}\|^2}{2n} \right). \tag{61}$$

Theorem 6. *For every $n \gg 1$ there exists an $n \overset{p}{\mapsto} 1$ QRAC with expected success probability $p = \frac{1}{2} + \sqrt{\frac{2}{3\pi n}}$.*

Proof. Because of the spherical symmetry of the probability density in formula (61), the average distance traveled after $n \gg 1$ steps of random walk is given by:

$$\mathop{\mathbb{E}}_{\{v_i\}_i}\left\| \sum_{i=1}^n v_i \right\| = \int_0^\infty R \cdot W(R) \cdot 4\pi R^2 dR = 2\sqrt{\frac{2n}{3\pi}}. \tag{62}$$

From (60) we have

$$\mathop{\mathbb{E}}_{\{v_i\}_i}(S_v) = 2^n \cdot 2\sqrt{\frac{2n}{3\pi}}.$$

By putting it back into (59) we obtain

$$\mathbb{E}(p) = \frac{1}{2} + \sqrt{\frac{2}{3\pi n}}. \qquad \square$$

4 Constructions of QRACs with SR

It is reasonable that one can do better than the lower bound obtained above using random measurements. In this section we will consider several constructions of quantum random access codes with shared randomness for some particular values of n. First, in Sect. 4.1 we will describe numerically obtained QRACs. Then, in Sect. 4.2 we will construct new QRACs with high degree of symmetry. In Sect. 4.3 we will compare both kinds of codes and draw some conclusions.

4.1 Numerical results

n	Section	Probability
2	4.1.1	0.8535533906
3	4.1.1	0.7886751346
4	4.1.2	0.7414814566
5	4.1.3	0.7135779205
6	4.1.4	0.6940463870
7		0.6786376322
8		0.6666334980
9	4.1.5	0.6568927813
10		0.6482001458
11		0.6410507577
12		0.6348711533

Table 1: The success probabilities of numerical $n \mapsto 1$ QRACs.

In this section we will discuss some particular $n \mapsto 1$ QRACs with shared randomness for several small values of n. These codes were obtained using numerical optimization. The optimization must be performed only over all possible measurements, because in Sect. 3.4 we showed that the choice of measurements in a simple way determines the optimal encoding. Each measurement is specified by a unit vector $\boldsymbol{v}_i \in \mathbb{R}^3$. For $n \mapsto 1$ QRAC there are n such vectors and one needs two angles to specify each of them. Without loss of generality we can assume that $\boldsymbol{v}_1 = (0, 0, 1)$ due to the rotational symmetry of the Bloch sphere. Thus only $2(n-1)$ real parameters are required to specify all $\boldsymbol{v}_i$ and therefore an $n \mapsto 1$ QRAC. To find the best configuration of measurements $\boldsymbol{v}_i$, one needs to maximize $S_{\boldsymbol{v}}$ given by (47). According to (45) the success probability of the corresponding QRAC is given by

$$p_{\boldsymbol{v}} = \frac{1}{2} \left(1 + \frac{S_{\boldsymbol{v}}}{2^n \cdot n} \right). \tag{63}$$

Once the measurements $\boldsymbol{v}_i$ are found, one can easily obtain the Bloch vector $\boldsymbol{r}_x$ of the qubit state that must be used to optimally encode the string x. We showed (see Sect. 3.4) that $\boldsymbol{r}_x$ is a unit vector in direction $\boldsymbol{v}_x$, where $\boldsymbol{v}_x$ is given by (43). For almost all QRACs that we have found using numerical optimization, points $\boldsymbol{r}_x$ form a symmetric pattern on the surface of the Bloch ball. Thus we were able to guess the exact values of $\boldsymbol{r}_x$ and $\boldsymbol{v}_i$. However, as in any numerical optimization, optimality of the resulting codes is not guaranteed.

In order to make the resulting codes more understandable, we depict them in three-dimensions using the following conventions:
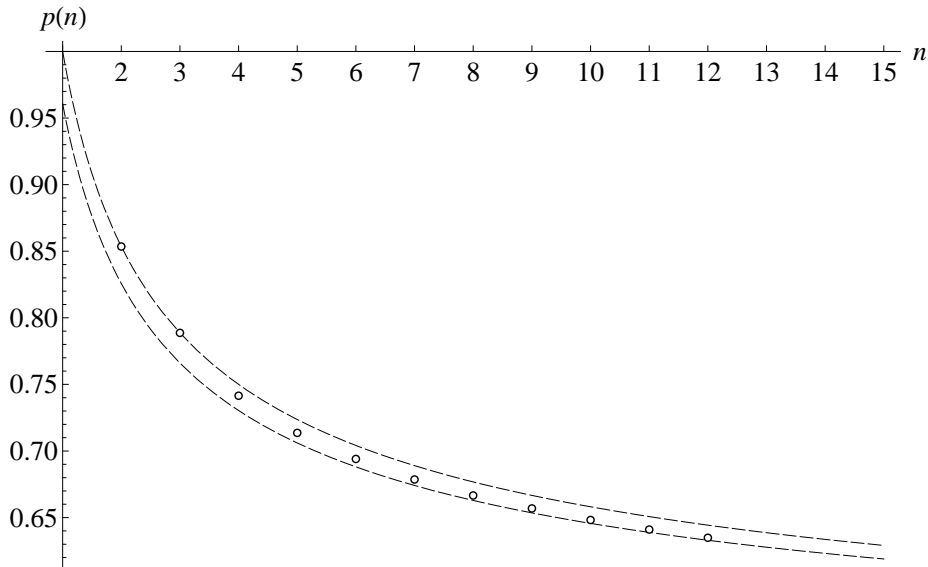
Figure 8: Success probabilities $p(n)$ of numerical $n \mapsto 1$ QRACs with SR from Table 1. Upper bound $\frac{1}{2} + \frac{1}{2\sqrt{n}}$ and lower bound $\frac{1}{2} + \sqrt{\frac{2}{3\pi n}}$ correspond to dashed lines (see Sects. 3.6 and 3.7, respectively).

- each *red point* encodes the string indicated,

- each *blue point* defines the axis of the measurement when the indicated bit is to be output,

- for each measurement there is a corresponding (unlabeled) *blue great circle* containing states yielding 0 and 1 equiprobably.

More precisely, the blue point with label i defines the basis vector $\left|\psi_0^i\right\rangle$ corresponding to the outcome 0 of the ith measurement (see Sect. 3.2). Note that the blue circles and blue points come in pairs – the vector $\left|\psi_0^i\right\rangle$ defined by the blue point is orthogonal to the corresponding circle. As a cautionary note, occasionally, the blue point for one measurement falls on the great circle of a different measurement (for example, blue points 1 and 2 in Fig. 9 lie on one another's corresponding circles). If there are too many red points, we omit the string labels for clarity.

Usually the codes have some symmetry, for example, the encoding points are the vertices of a polyhedron. In such cases we show the corresponding polyhedron instead of the Bloch sphere. We do not discuss $7 \mapsto 1$ and $8 \mapsto 1$ QRACs since the best numerical results have almost no symmetry. We also do not discuss the numerical results for $n \geq 10$ (see Table 1 for success probabilities). Numerical $10 \mapsto 1$ code is symmetric and resembles $6 \mapsto 1$ code discussed in Sect. 4.1.4, but numerical $11 \mapsto 1$ and $12 \mapsto 1$ codes again have almost no symmetry. Success probabilities of all numerical $n \mapsto 1$ QRACs with SR are summarized in Table 1 and Fig. 8.
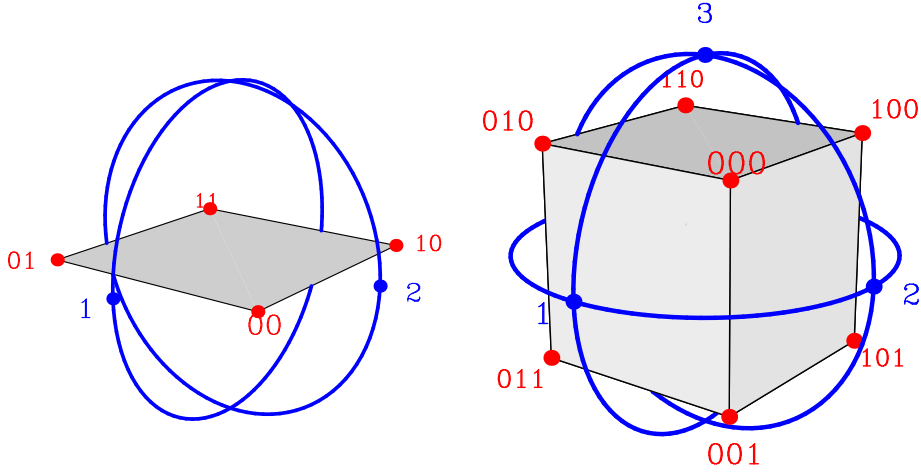
Figure 9: The $2 \mapsto 1$ QRAC with SR.   Figure 10: The $3 \mapsto 1$ QRAC with SR.

4.1.1   The $2 \mapsto 1$ and $3 \mapsto 1$ QRACs with SR

We used numerical optimization as described above to find $2 \mapsto 1$ and $3 \mapsto 1$ QRACs with shared randomness and obtained the optimal codes discussed in Sects. 3.3.1 and 3.3.2.

The codes are shown in Fig. 9 and 10, respectively. In the first case the encoding points are the vertices of a square and the success probability is

$$p = \frac{1}{2} + \frac{1}{2\sqrt{2}} \approx 0.8535533906. \tag{64}$$

In the second case they are the vertices of a cube. The success probability is

$$p = \frac{1}{2} + \frac{1}{2\sqrt{3}} \approx 0.7886751346. \tag{65}$$

4.1.2   The $4 \mapsto 1$ QRAC with SR

In Sect. 3.3.3 we discussed the impossibility of a $4 \mapsto 1$ QRAC when Alice and Bob are not allowed to cooperate. However, a $4 \mapsto 1$ QRAC can be obtained if they have shared randomness. The particular $4 \mapsto 1$ QRAC with SR discussed here was found by a numerical optimization. It is a hybrid of the $2 \mapsto 1$ and $3 \mapsto 1$ codes discussed in Sects. 3.3.1 and 3.3.2, respectively.

The measurements are preformed in the bases (M_1, M_2, M_3, M_3), where M_1, M_2, and M_3 are the same as in the $3 \mapsto 1$ case (note that the last two bits are measured in the same basis, namely M_3). These bases are given by (30), (31), and (35), respectively. The points that correspond to an optimal encoding for these bases are the vertices of a regular square $\frac{1}{\sqrt{2}}(\pm 1, \pm 1, 0)$ in the xy plane and a cube $\frac{1}{\sqrt{6}}(\pm 1, \pm 1, \pm 2)$ that is stretched in the z direction (see the Bloch sphere in Fig. 11). The Bloch vector for the string $x = x_1 x_2 x_3 x_4$ is explicitly given by

$$\boldsymbol{r}(x) = \frac{1}{\sqrt{6}} \begin{pmatrix} (-1)^{x_1}\big(1 - (1 - \sqrt{3})\,|x_3 - x_4|\big) \\ (-1)^{x_2}\big(1 - (1 - \sqrt{3})\,|x_3 - x_4|\big) \\ (-1)^{x_3} + (-1)^{x_4} \end{pmatrix}. \tag{66}$$
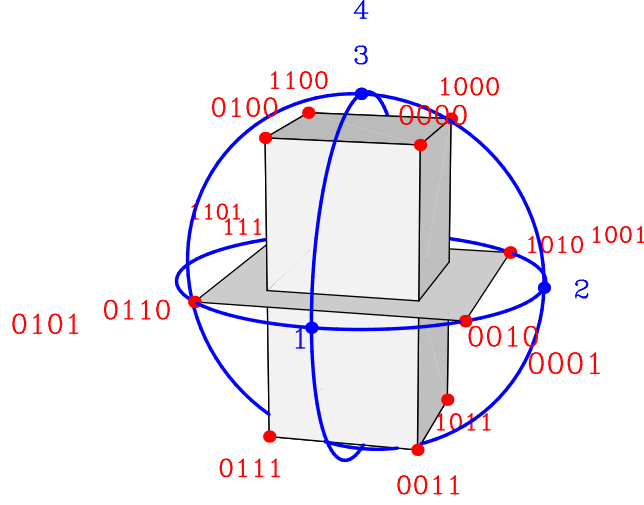
26

Figure 11: The $4 \mapsto 1$ QRAC with SR.

The encoding function can be described as follows:

- if $x_3 = x_4$, use the usual $3 \mapsto 1$ QRAC with an emphasis on x_3 to encode the string $x_1 x_2 x_3$,

- if $x_3 \neq x_4$ – encode only $x_1 x_2$ using the usual $2 \mapsto 1$ QRAC.

In the $3 \mapsto 1$ scheme the probability to recover x_3 must be increased by stretching the cube along the z axis, because x_3 equals x_4 and therefore it is of greater value than x_1 or x_2.

This $4 \mapsto 1$ QRAC can also be seen as a combination of two $3 \mapsto 1$ QRACs: the string $x_1 x_2 x_3$ is encoded into the vertices of a smaller cube inscribed in a half of the Bloch ball (the vertices that lie within the sphere are projected to its surface). The last bit x_4 indicates in which half the smaller cube lies (the upper and lower hemispheres correspond to $x_4 = 0$ and 1, respectively).

The qubit state is explicitly given by $E(x_1, x_2, x_3, x_4) = \alpha \left|0\right\rangle + \beta \left|1\right\rangle$, where

$$\begin{cases} \alpha = \sqrt{\dfrac{1}{2} + \dfrac{(-1)^{x_3} + (-1)^{x_4}}{2\sqrt{6}}}, \\[2mm] \beta = \dfrac{(-1)^{x_1} + i(-1)^{x_2}}{\sqrt{4\big(3 - 2\left|x_3 - x_4\right|\big) + 2\sqrt{6}\big((-1)^{x_3} + (-1)^{x_4}\big)}}. \end{cases} \tag{67}$$

The 16 values for β are exactly the sixteen roots of the polynomial (recall Sect. 3.1.2)

$$2304\beta^{16} + 3072\beta^{12} + 1120\beta^8 + 128\beta^4 + 1. \tag{68}$$

If the shared random string is not available, the worst case success probability of this QRAC is $\frac{1}{2}$. However, if the shared randomness is available, the input randomization (as in Lemma 1) can be used and we will get the same success
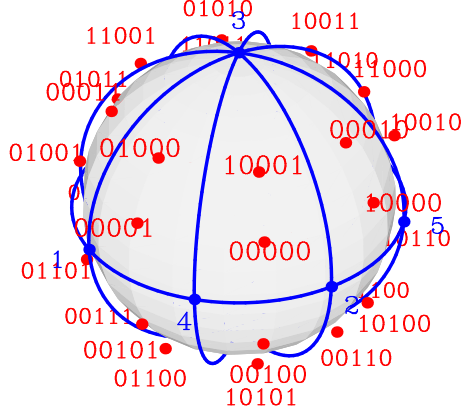
Figure 12: The $5 \mapsto 1$ QRAC with SR.

probability for all inputs, namely

$$p = \frac{1}{2} + \frac{1 + \sqrt{3}}{8\sqrt{2}} \approx 0.7414814566. \tag{69}$$

4.1.3 The $5 \mapsto 1$ QRAC with SR

To obtain a $5 \mapsto 1$ QRAC, we take the bases M_1, M_2, and M_3, given by (30), (31), and (35), respectively, and also:

$$M_4 = \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2} \\ i+1 \end{pmatrix}, \frac{1}{2} \begin{pmatrix} -\sqrt{2} \\ i+1 \end{pmatrix} \right\}, \tag{70}$$

$$M_5 = \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2} \\ i-1 \end{pmatrix}, \frac{1}{2} \begin{pmatrix} -\sqrt{2} \\ i-1 \end{pmatrix} \right\}. \tag{71}$$

The Bloch vectors $\boldsymbol{v}_3 = (0, 0, \pm 1)$ for the basis M_3 are along the z axis, but the Bloch vectors of the other four bases form a regular octagon in the xy plane (shown in Fig. 12): $\boldsymbol{v}_1 = (\pm 1, 0, 0)$, $\boldsymbol{v}_2 = (0, \pm 1, 0)$, $\boldsymbol{v}_4 = \pm \frac{1}{\sqrt{2}}(1, 1, 0)$, $\boldsymbol{v}_5 = \pm \frac{1}{\sqrt{2}}(-1, 1, 0)$. The Bloch vector encoding the string $x = x_1 x_2 x_3 x_4 x_5$ is

$$\boldsymbol{r}(x) = \frac{1}{\sqrt{10 + s(x)4\sqrt{2}}} \begin{pmatrix} \sqrt{2}(-1)^{x_1} + (-1)^{x_4} - (-1)^{x_5} \\ \sqrt{2}(-1)^{x_2} + (-1)^{x_4} + (-1)^{x_5} \\ \sqrt{2}(-1)^{x_3} \end{pmatrix}, \tag{72}$$

where $s(x) \in \{-1, 1\}$ and is given by

$$s(x) = \frac{(-1)^{x_1} + (-1)^{x_2}}{2}(-1)^{x_4} - \frac{(-1)^{x_1} - (-1)^{x_2}}{2}(-1)^{x_5}. \tag{73}$$

The great circles with equiprobable outcomes of the measurements partition the Bloch sphere into 16 equal spherical triangles. There are two strings encoded into each triangle. The idea how to locate the right point for the given string x is as follows. Observe that the strings with $x_3 = 0$ and $x_3 = 1$ are encoded into
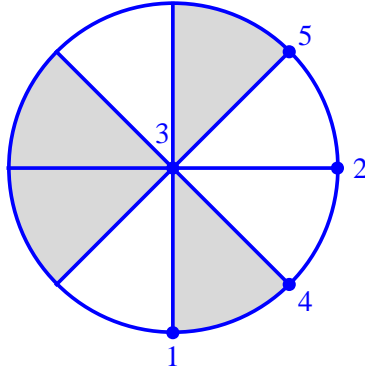
Figure 13: The "preferable regions" of the measurement M_2 (only the upper hemisphere is shown, the other half is symmetric). For each of the measurements the direction of the Bloch vector $|\psi_0\rangle$ is indicated by the corresponding number. The white triangles correspond to $x_2 = 0$, but the gray ones to $x_2 = 1$.

the upper and lower hemisphere, respectively (this means that for all strings the probability that the measurement M_3 gives the correct value of x_3 is greater than $\frac{1}{2}$). Next observe that a half of all strings have $s(x) = 1$, but the other half has $s(x) = -1$ (in fact, the two strings in the same triangle have distinct values of s).

Let us first consider the case $s(x) = 1$. We call such string *compatible* with the measurements, because it can be encoded in such a way that every measurement gives the correct value of the corresponding bit with probability greater than $\frac{1}{2}$. For the ith bit of x we can define the "preferable region" on the Bloch sphere as the hemisphere where M_i recovers x_i with probability greater than $\frac{1}{2}$. The intersection of these five regions is one sixteenth of the Bloch sphere – the triangle where x must be encoded. The point with the smallest absolute value of the z coordinate in this triangle must be chosen (it has smaller probability to recover x_3 correctly, but the probabilities for the other four bits are larger).

If $s(x) = -1$, the string x is *incompatible* with the measurements, because the intersection of the "preferable regions" is empty. Thus, no matter where the string is encoded, at least one bit will differ from the most probable outcome of the corresponding measurement. We can take this into account and modify the definition of the "preferable region" for the ith bit ($i \neq 3$). It is a union of eight triangles: four triangles, where the most probable outcome of M_i equals x_i, and four triangles where it does not equal x_i (in either case the triangles with maximal probability of correct outcome of M_i must be taken). For example, the "preferable regions" for x_2 are shown in Fig. 13. The regions for x_3 remain the same as in the previous case. The intersection of all five regions for the given string x is the triangle, where the string must be encoded. The point with the largest absolute value of z coordinate in the triangle must be chosen. As a result, three of the measurements will give the correct value of the corresponding bit of string x with probability greater than $\frac{1}{2}$.

The corresponding qubit state is given by $E(x_1, x_2, x_3, x_4, x_5) = \alpha |0\rangle + \beta |1\rangle$
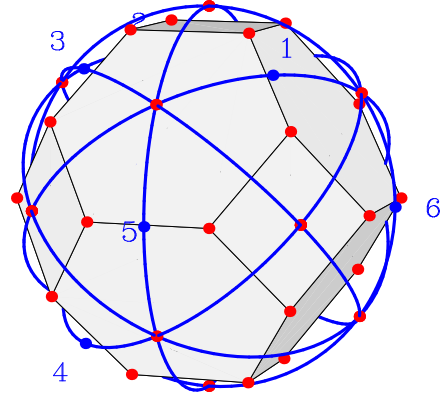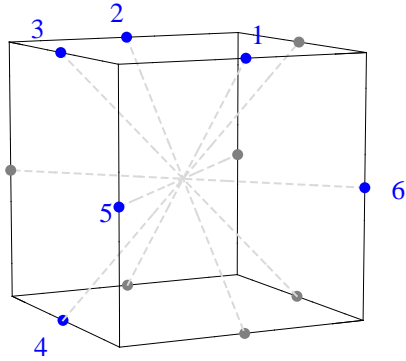
Figure 14: The measurements for the $6 \mapsto 1$ QRAC shown on the right.

Figure 15: The $6 \mapsto 1$ QRAC with SR.

with coefficients α and β defined as follows:

$$\begin{cases} \alpha = \sqrt{\dfrac{1}{2} + \dfrac{(-1)^{x_3}}{2\sqrt{5 + s(x)2\sqrt{2}}}}, \\[4mm] \beta = \dfrac{(-1)^{x_1} + i(-1)^{x_2} + \frac{i+1}{\sqrt{2}}(-1)^{x_4} + \frac{i-1}{\sqrt{2}}(-1)^{x_5}}{\sqrt{10 + s(x)4\sqrt{2} + 2(-1)^{x_3}\sqrt{5 + s(x)2\sqrt{2}}}}. \end{cases} \tag{74}$$

The coefficients β are the roots of the polynomial

$$1336336\beta^{32} + 961792\beta^{24} + 151432\beta^{16} + 1600\beta^8 + 1. \tag{75}$$

Again, using input randomization we obtain the same success probability for any input, namely

$$p = \frac{1}{2} + \frac{1}{20}\sqrt{2(5 + \sqrt{17})} \approx 0.7135779205. \tag{76}$$

4.1.4 The $6 \mapsto 1$ QRAC with SR

The Bloch vectors corresponding to the 6 measurements are as follows:

$$\begin{aligned} \boldsymbol{v}_1 &= \pm(0, +1, +1)/\sqrt{2}, \\ \boldsymbol{v}_2 &= \pm(0, -1, +1)/\sqrt{2}, \\ \boldsymbol{v}_3 &= \pm(+1, 0, +1)/\sqrt{2}, \\ \boldsymbol{v}_4 &= \pm(+1, 0, -1)/\sqrt{2}, \\ \boldsymbol{v}_5 &= \pm(+1, +1, 0)/\sqrt{2}, \\ \boldsymbol{v}_6 &= \pm(-1, +1, 0)/\sqrt{2}. \end{aligned} \tag{77}$$

They correspond to the 12 vertices of the *cuboctahedron* (or the midpoints of the 12 edges of the cube) and are shown in Fig. 14. The great circles orthogo-

nal to these vectors form the projection of the edges of a normalized[3] *tetrakis hexahedron* and partition the Bloch sphere into 24 parts (see Fig. 15). Each of these parts contains one vertex of a *truncated octahedron* – the dual of tetrakis hexahedron. It is inscribed in the Bloch sphere shown in Fig. 15.

The measurement bases corresponding to v_i can be found using (23):

$$
\begin{aligned}
M_1 &= \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2+\sqrt{2}} \\ i\sqrt{2-\sqrt{2}} \end{pmatrix}, \frac{1}{2} \begin{pmatrix} \sqrt{2-\sqrt{2}} \\ -i\sqrt{2+\sqrt{2}} \end{pmatrix} \right\}, \\
M_2 &= \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2+\sqrt{2}} \\ -i\sqrt{2-\sqrt{2}} \end{pmatrix}, \frac{1}{2} \begin{pmatrix} \sqrt{2-\sqrt{2}} \\ i\sqrt{2+\sqrt{2}} \end{pmatrix} \right\}, \\
M_3 &= \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2+\sqrt{2}} \\ \sqrt{2-\sqrt{2}} \end{pmatrix}, \frac{1}{2} \begin{pmatrix} \sqrt{2-\sqrt{2}} \\ -\sqrt{2+\sqrt{2}} \end{pmatrix} \right\}, \\
M_4 &= \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2-\sqrt{2}} \\ \sqrt{2+\sqrt{2}} \end{pmatrix}, \frac{1}{2} \begin{pmatrix} \sqrt{2+\sqrt{2}} \\ -\sqrt{2-\sqrt{2}} \end{pmatrix} \right\}, \\
M_5 &= \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2} \\ i+1 \end{pmatrix}, \frac{1}{2} \begin{pmatrix} \sqrt{2} \\ -i-1 \end{pmatrix} \right\}, \\
M_6 &= \left\{ \frac{1}{2} \begin{pmatrix} \sqrt{2} \\ i-1 \end{pmatrix}, \frac{1}{2} \begin{pmatrix} \sqrt{2} \\ -i+1 \end{pmatrix} \right\}.
\end{aligned}
\tag{78}
$$

Note that M_5 and M_6 are the same as (70) and (71) for the $5 \mapsto 1$ QRAC described in the previous section. Another way to describe these 6 bases is to consider the β coefficients for 12 vectors that form them. It turns out that these coefficients are exactly the roots of the polynomial

$$
256\beta^{12} - 128\beta^8 - 44\beta^4 + 1.
\tag{79}
$$

Let us consider how to determine the point where a given string should be encoded. According to (43) we have to find the sum of vectors v_i defined in (77), each taking with either plus or minus sign. These vectors correspond to six pairs of opposite edges of a cube and the signs determine which edge from each pair we are taking (see Fig. 14). There are only three distinct kinds of ways of doing this (see Fig. 16). Regardless of which way it is, for each of the chosen edges there is exactly one other that shares a common face and is parallel to it. Thus we can partition the chosen edges into three pairs (in Fig. 16 such pairs are joined with a thick blue line). The sum of the vectors v_i for edges in a pair is always parallel to one of the axis and its direction is indicated with an arrow in Fig. 16. From these arrows one can see where the encoding point should lie.

Now let us classify all $2^6 = 64$ strings of length 6 into 3 types according to the location of the encoding point on the Bloch sphere. Each type of strings is encoded into a vertex of a specific polyhedron (see Fig. 17). These polyhedra are *cube*, *truncated octahedron*, and *octahedron* and the number of strings of each type are 16, 24, and 24, respectively. Let us consider them case by case:

---

[3] The vertices of the *tetrakis hexahedron* are not all at the same distance from the origin (the ones forming an octahedron are $2/\sqrt{3}$ times closer than those forming a cube). So the polyhedron has to be *normalized* to fit inside the Bloch sphere (the vectors pointing to the vertices have to be rescaled to have a unit norm).
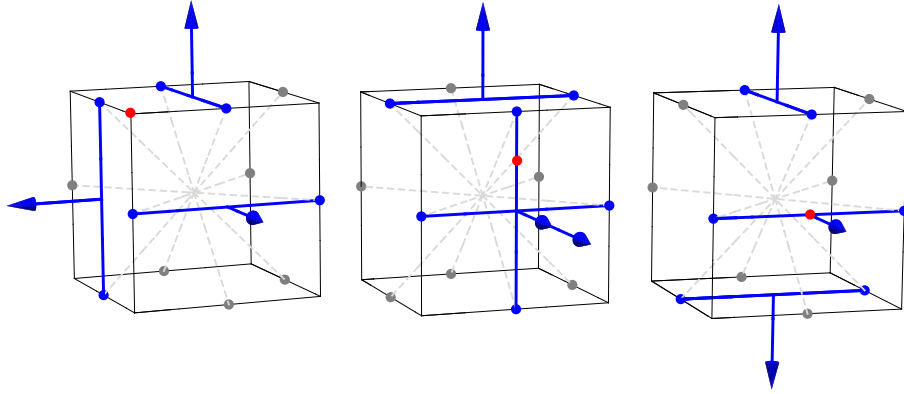
Figure 16: Three distinct kinds of ways of choosing one edge from each pair of opposite edges of a cube. The chosen edges are marked with blue points. Points lying on opposite edges of the same face are connected and the direction of the sum of the corresponding vectors is indicated with an arrow. The corresponding encoding point is shown in red. The red points obtained from all possible choices of the same kind are the vertices of a cube, a truncated octahedron, and an octahedron, respectively (see Fig. 17).

- *Cube* has 8 vertices:

$$\frac{1}{\sqrt{3}}(\pm 1, \pm 1, \pm 1) \tag{80}$$

  and there are 2 strings encoded into each vertex. These 16 strings are exactly those $x_1 x_2 \ldots x_6 \in \{0, 1\}^6$ that satisfy:

$$|x_1 - x_2| + |x_3 - x_4| + |x_5 - x_6| \in \{0, 3\}. \tag{81}$$

  This condition ensures that the three arrows in Fig. 16 are orthogonal.

- *Truncated octahedron* has 24 vertices and their coordinates are obtained by all permutations of the components of

$$\frac{1}{\sqrt{5}}(0, \pm 1, \pm 2). \tag{82}$$

Truncated octahedron	Octahedron
$**1110$	$**1101$
$**0001$	$**0010$
$10**11$	$01**11$
$01**00$	$10**00$
$1110**$	$1101**$
$0001**$	$0010**$

Table 2: Patterns of strings corresponding to the vertices of truncated octahedron and octahedron ("$*$" stands for any value).
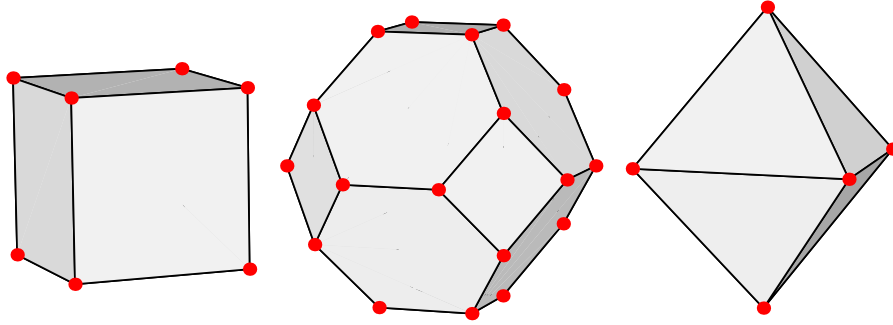
Figure 17: Three polyhedra (cube, truncated octahedron, and octahedron) corresponding to three different types of strings for $6 \mapsto 1$ QRAC with SR. The red points in Fig. 15 are obtained by superimposing these three polyhedra.

There is just 1 string encoded into each vertex. In this case there will be two pairs of chosen edges that belong to the same face (note the "cross" in the Fig. 16 formed by pairs whose arrows are pointing outwards of the page). The third pair (with the arrow pointing up) can be rotated around this face to any of the four possible positions. This corresponds to fixing four bits of the string and choosing the remaining two bits in an arbitrary way. Since the "cross" can be on any of the six faces of the cube, one can easily describe all 24 strings of this type (they are listed in the first column of Table 2).

- *Octahedron* has 6 vertices:

$$(\pm 1, 0, 0) \cup (0, \pm 1, 0) \cup (0, 0, \pm 1) \tag{83}$$

and there are 4 strings encoded into each vertex. In this case two arrows in Fig. 16 are pointing to opposite directions (up and down). If we fix these arrows, we can rotate the third one (pointing outwards) in any of the four positions. Hence we can describe all 24 strings of this type in a similar way (see the second column of Table 2).

The coefficients β of the encoding states are the 64 roots of the polynomial

$$\beta^4(\beta - 1)^4(4\beta^4 - 1)^4(36\beta^8 + 24\beta^4 + 1)^2$$
$$(25\beta^8 - 15\beta^4 + 1)(400\beta^8 - 360\beta^4 + 1)(400\beta^8 + 56\beta^4 + 25). \tag{84}$$

The obtained success probability using input randomization is

$$p = \frac{1}{2} + \frac{2 + \sqrt{3} + \sqrt{15}}{16\sqrt{6}} \approx 0.6940463870. \tag{85}$$
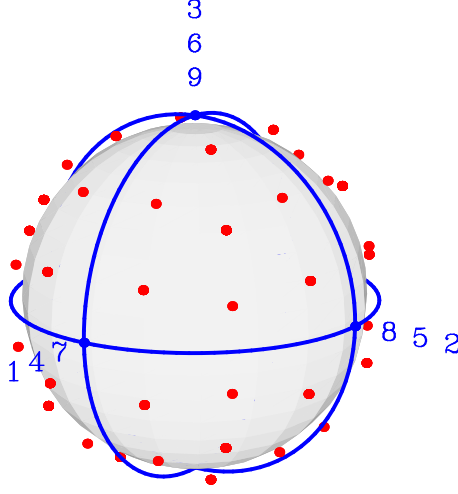
Figure 18: The $9 \mapsto 1$ QRAC with SR.

4.1.5 The $9 \mapsto 1$ QRAC with SR

This QRAC is a combination of three $3 \mapsto 1$ QRACs described in Sect. 3.3.2. It has three measurements along each axis:

$$
\begin{aligned}
\boldsymbol{v}_1 = \boldsymbol{v}_4 = \boldsymbol{v}_7 &= \pm(1,0,0), \\
\boldsymbol{v}_2 = \boldsymbol{v}_5 = \boldsymbol{v}_8 &= \pm(0,1,0), \\
\boldsymbol{v}_3 = \boldsymbol{v}_6 = \boldsymbol{v}_9 &= \pm(0,0,1).
\end{aligned}
\tag{86}
$$

The measurement bases M_1, M_2, and M_3 corresponding to the Bloch vectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, and $\boldsymbol{v}_3$ are given by (30), (31), and (35), respectively.

The encoding points can be characterized as a $4 \times 4 \times 4$ *cubic lattice* formed by vectors (43) projected on the surface of the Bloch ball. Note that this lattice consists of vertices of 8 equal cubes each lying in a different octant. Then the 7 points inside of each spherical triangle in Fig. 18 are the projection of the vertices of the corresponding cube.

All $2^9 = 512$ strings can be classified into 3 types. First consider a string $a_1 a_2 a_3 \in \{0,1\}^3$ and define

$$
s(a_1, a_2, a_3) = \frac{|a_1 - a_2| + |a_2 - a_3| + |a_3 - a_1|}{2}.
\tag{87}
$$

Notice that $s(a_1, a_2, a_3) \in \{0, 1\}$. Now for $x = x_1 x_2 \ldots x_9 \in \{0,1\}^9$ define

$$
t(x) = s(x_1, x_4, x_7) + s(x_2, x_5, x_8) + s(x_3, x_6, x_9).
\tag{88}
$$

Then the type of the string x can be determined as follows:

$$
t(x) = \begin{cases} 0, 3 & \text{cube}, \\ 1 & \text{truncated cube}, \\ 2 & \text{small rhombicuboctahedron}. \end{cases}
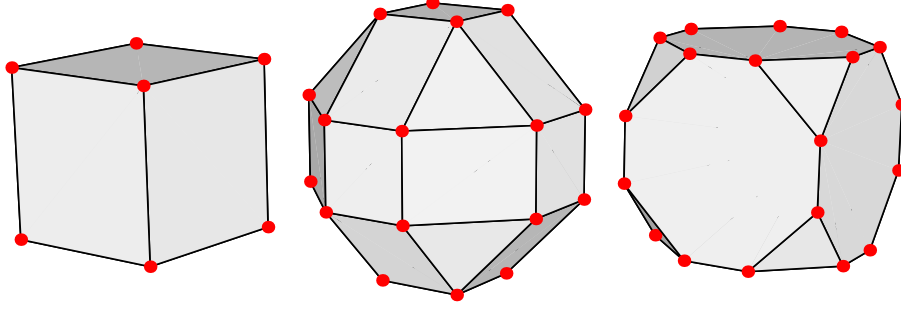\tag{89}
$$

34

Figure 19: Three polyhedra (cube, small rhombicuboctahedron, and truncated cube) corresponding to three different types of strings for $9 \mapsto 1$ QRAC with SR. The red points in Fig. 18 are obtained by superimposing these three polyhedra.

These types are named after polyhedra, since each type of string is encoded into the vertices of the corresponding polyhedron (see Fig. 19):

- *Cube* has 8 vertices and there are 28 strings encoded into each vertex. These vertices are:

$$\frac{1}{\sqrt{3}}(\pm 1, \pm 1, \pm 1). \tag{90}$$

- Deformed[4] *truncated cube* has 24 vertices and there are 3 strings encoded into each vertex. These vertices are:

$$\frac{1}{\sqrt{19}}(\pm 1, \pm 3, \pm 3) \cup \frac{1}{\sqrt{19}}(\pm 3, \pm 1, \pm 3) \cup \frac{1}{\sqrt{19}}(\pm 3, \pm 3, \pm 1). \tag{91}$$

- Deformed[5] *small rhombicuboctahedron* also has 24 vertices and there are 9 strings encoded into each vertex. These vertices are:

$$\frac{1}{\sqrt{11}}(\pm 3, \pm 1, \pm 1) \cup \frac{1}{\sqrt{11}}(\pm 1, \pm 3, \pm 1) \cup \frac{1}{\sqrt{11}}(\pm 1, \pm 1, \pm 3). \tag{92}$$

The coefficients β for the corresponding qubit states $\alpha \ket{0} + \beta \ket{1}$ are the roots of the following polynomial:

$$(36\beta^8 + 24\beta^4 + 1)^{28}(1444\beta^8 + 760\beta^4 + 81)^3(484\beta^8 + 440\beta^4 + 1)^9$$
$$(52128400\beta^{16} - 21509824\beta^{12} + 26780424\beta^8 - 372400\beta^4 + 15625)^3$$
$$(5856400\beta^{16} - 1788864\beta^{12} + 1232264\beta^8 - 92400\beta^4 + 15625)^9. \tag{93}$$

Using input randomization we get success probability

$$p = \frac{1}{2} + \frac{192 + 10\sqrt{3} + 9\sqrt{11} + 3\sqrt{19}}{384} \approx 0.6568927813. \tag{94}$$

---

[4]The edges of the *truncated cube* are of the same length. In our case the eges forming triangles are $\sqrt{2}$ times longer than the other edges.

[5]The edges of the *small rhombicuboctahedron* are also of the same length, but in our case the edges forming triangles again are $\sqrt{2}$ times longer.

4.2 Symmetric constructions

In Sect. 4.1 we have discussed in great detail $n \mapsto 1$ quantum random access codes with shared randomness for some particular values of n. Since these codes were obtained using numerical optimization, there are still some questions left open. Most importantly, are the codes for $n \geq 4$ discussed in Sect. 4.1 optimal? If this is the case, do these codes (see Figs. 9, 10, 11, 12, 15, and 18) have anything in common that makes them so good?

The purpose of this section is to shed some light on these two questions. We will explore the possibility that *symmetry* is the property that makes QRACs with SR good. In Sect. 4.2.1 we will explore what symmetries do the codes found by numerical optimization have and what are other possible symmetries that we could use. In several subsequent sections we will use these symmetries to construct new codes to see how they compare with the numerically obtained ones (the success probabilities of the obtained codes are summarized in Table 3). In Sect. 4.3 we will conclude that symmetric codes are *not* necessarily optimal and speculate what else potentially could be used to construct good QRACs.

n	Section	Probability
4	4.2.2	0.7332531755
6	4.2.3	0.6940418856
9	4.2.4	0.6563927998
15	4.2.5	0.6201829084

Table 3: The success probabilities of symmetric $n \mapsto 1$ QRACs with SR.

4.2.1 Symmetric great circle arrangements

If we want to construct a QRAC with SR that has some sort of symmetry, we have to choose the directions of measurements in a symmetric way. In other words, we have to symmetrically arrange the great circles that are orthogonal to the measurement directions.

In this section we will discuss two ways how great circles can be arranged on sphere in a symmetric way. These arrangements come from *quasiregular polyhedra* and *triangular symmetry groups*, respectively. The first kind of arrangement cannot be directly observed in numerically obtained examples, despite its high symmetry. However, the second one can be observed in almost all numerically obtained codes. Since our approach is more or less empiric, we will not justify when an arrangement is "symmetric enough"[6] to be of interest. We will use the term *symmetric codes* to refer to the codes constructed below. This is just to distinguish them from numerically obtained codes in Sect. 4.1, not because they satisfy some formal criterion of "being symmetric".

Quasiregular polyhedra

---

[6]Several possible criteria are: (a) any great circle can be mapped to any other by a rotation from the symmetry group of the arrangement, (b) the sphere is cut into pieces that are regular polygons, (c) the sphere is cut into pieces of the same form. However, not all examples we will give satisfy these three conditions. In fact, each condition is violated by at least one of the examples we will consider.
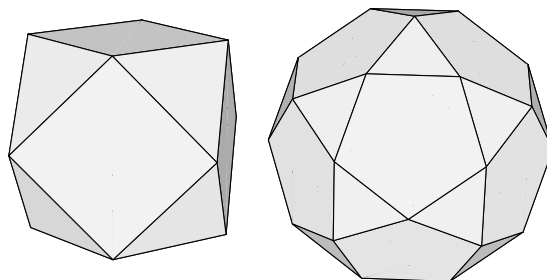
Figure 20: Quasiregular polyhedra: cuboctahedron and icosidodecahedron.

A (convex) *quasiregular polyhedron* is the intersection of a Platonic solid with its dual. There are only three possibilities:

$$\text{octahedron} = \text{tetrahedron} \cap \text{tetrahedron},\tag{95}$$

$$\text{cuboctahedron} = \text{cube} \cap \text{octahedron},\tag{96}$$

$$\text{icosidodecahedron} = \text{icosahedron} \cap \text{dodecahedron}.\tag{97}$$

Usually octahedron is not considered to be quasiregular, since it is Platonic. Thus there are only two convex quasiregular polyhedra (see Fig. 20).

These polyhedra have several nice properties. For example, all their edges are equivalent and there are exactly two types of faces (both regular polygons), each completely surrounded by the faces of the other type. The most relevant property for us is that their edges form great circles. Since the arrangements of great circles formed by the edges of *cuboctahedron* and *icosidodecahedron* do not appear in the numerical codes, we will use them in Sects. 4.2.2 and 4.2.3 to construct new (symmetric) $4 \mapsto 1$ and $6 \mapsto 1$ QRACs with SR, respectively.

Triangular symmetry groups

Consider a spherical triangle – it is enclosed by three planes that pass through its edges and the center of the sphere. Let us imagine that these planes are mirrors that reflect our triangle. These three reflections generate a *reflection group* [8, 9]. For some specific choices of the triangle this group is finite and the images of the triangle under different group operations do not overlap. Hence they form a *tiling* of the sphere. This tiling can also be seen as several (most likely more than three) great circles cutting the sphere into equal triangles.

We can choose any of the triangles in the tiling and repeatedly reflect it along its edges so that it moves around one of its vertices. It means, the angles of the corners that meet at any vertex of the tiling must be equal. Moreover, we do not want the triangle to intersect with any of the mirrors, thus only an even number of triangles can meet at a vertex.[7]

Hence the angles of the spherical triangle must be $(\frac{\pi}{p}, \frac{\pi}{q}, \frac{\pi}{r})$ for some integers $p, q, r \geq 2$. The sum of the angles of a spherical triangle is at least π, thus the

---

[7] Fore example, the tiling that arises by projecting the edges of an icosahedron onto the sphere has five triangles meeting at a vertex. This is not good, since when any of the edges in the tiling is extended to a great circle, it will intersect other triangles.
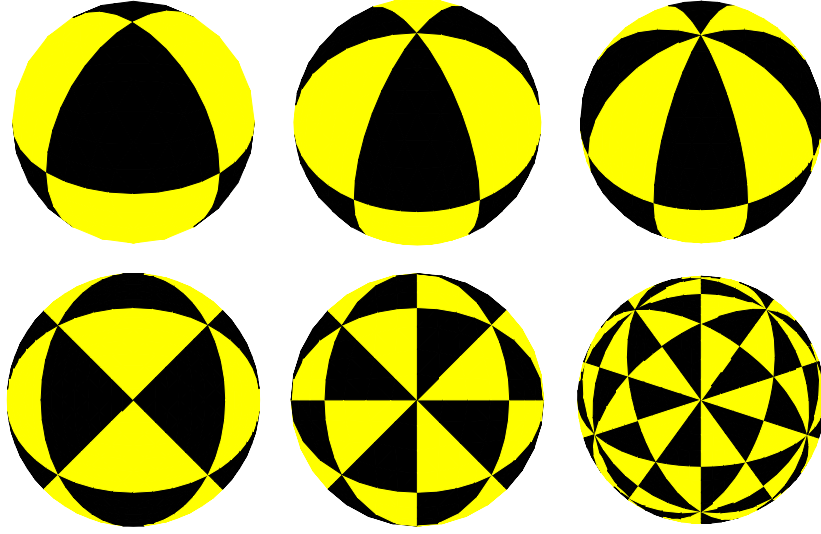
Figure 21: Triangular symmetry groups. First row: $(2, 2, 2)$, $(2, 2, 3)$, $(2, 2, 4)$. Second row: $(2, 3, 3)$, $(2, 3, 4)$, $(2, 3, 5)$.

numbers p, q, r must satisfy:

$$\frac{1}{p} + \frac{1}{q} + \frac{1}{r} > 1. \tag{98}$$

If $p \leq q \leq r$, the only solutions are: $(2, 2, k)$ for any $k \geq 2$, $(2, 3, 3)$, $(2, 3, 4)$, and $(2, 3, 5)$. The tilings corresponding to these solutions are shown in Fig. 21. The symmetry group of such tiling is called *triangular symmetry group* [9, pp. 158] and is denoted by (p, q, r).

We can observe these tilings in almost all numerically obtained QRACs discussed in Sect. 4.1. They are formed when the great circles corresponding to measurements partition the Bloch sphere into equal triangles. All such cases are summarized in Table 4. Tilings appearing in $2 \mapsto 1$ and $4 \mapsto 1$ QRACs that are not mentioned in the table can be seen as degenerate cases.

n	(p, q, r)	Polyhedron	Section and figure
3	$(2, 2, 2)$	octahedron	Sect. 4.1.1, Fig. 10
5	$(2, 2, 4)$	normalized octagonal dipyramid	Sect. 4.1.3, Fig. 12
6	$(2, 3, 3)$	normalized tetrakis hexahedron	Sect. 4.1.4, Fig. 15
9	$(2, 2, 2)$	octahedron	Sect. 4.1.5, Fig. 18

Table 4: Triangular symmetry groups of numerical $n \mapsto 1$ QRACs.

The tilings corresponding to triangular symmetry groups $(2, 3, 4)$ and $(2, 3, 5)$ do not appear in numerically obtained codes. Thus they will be used to construct new (symmetric) $9 \mapsto 1$ and $15 \mapsto 1$ QRACs with SR in Sects. 4.2.4 and 4.2.5, respectively. To each tiling one can associate a corresponding polyhedron with equal triangular faces. The polyhedra corresponding to $(2, 3, 4)$ and
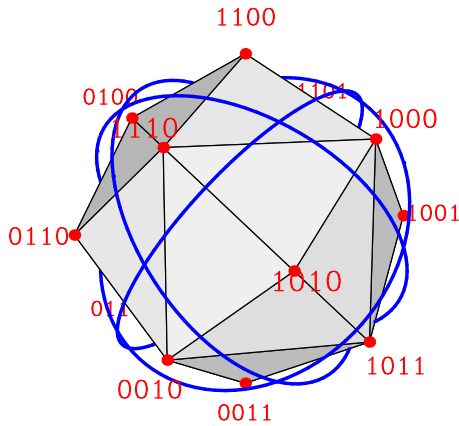
Figure 22: Symmetric $4 \mapsto 1$ QRAC with SR.

$(2, 3, 5)$ are called normalized[8] *disdyakis dodecahedron* and normalized *disdyakis triacontahedron*, respectively.

Polyhedra arising from both constructions of symmetric great circle arrangements are summarized in Table 5. Since the great circle arrangements corresponding to the four marked polyhedra do not appear in numerically obtained codes, we will use them construct new (symmetric) QRACs with SR.

n	Faces		(p, q, r)	Polyhedron
3	8	8	$(2, 2, 2)$	octahedron
4	14	14	QR	cuboctahedron ✓
6	32	32	QR	icosidodecahedron ✓
6	24	32	$(2, 3, 3)$	normalized tetrakis hexahedron
9	48	74	$(2, 3, 4)$	normalized disdyakis dodecahedron ✓
15	120	212	$(2, 3, 5)$	normalized disdyakis triacontahedron ✓

Table 5: The list of polyhedra whose edges form great circles. The first column indicates the number of great circles. The next two indicate, respectively, the number of faces of the polyhedron and the maximal number of pieces achievable by cutting the sphere with n great circles (see Sect. 3.3.3). The fourth column indicates the triangular symmetry group (QR means quasiregular). The name of the polyhedron is provided in the last column. Four marked polyhedra will be used in subsequent sections to construct symmetric QRACs with SR.

4.2.2   Symmetric $4 \mapsto 1$ QRAC with SR

Recall that in Sect. 3.3.3 we proved that four planes passing through the center of the Bloch sphere partition its surface into at most 14 parts. The most symmetric way to obtain 14 parts is to use the four planes parallel to the four faces of a

---

[8]*Normalized* means that all vectors pointing from the origin to the vertices of the polyhedron are rescaled to have unit norm.
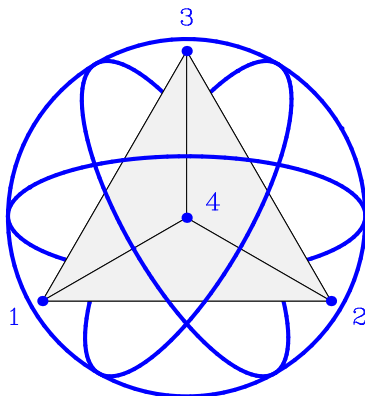
Figure 23: A regular tetrahedron and four great circles parallel to its faces. The circles are determined by the measurements in the direction of the vertices of the tetrahedron. The numbers at the vertices indicate the Bloch vectors of basis states $|\psi_0\rangle$ of the measurements for the $4 \mapsto 1$ QRAC shown in Fig. 22.

regular *tetrahedron*. The measurements are along the four directions given by the vertices (see Fig. 23).

The simplest way to construct a regular tetrahedron is to choose four specific vertices of a cube $\frac{1}{\sqrt{3}}(\pm 1, \pm 1, \pm 1)$. For example, the ones with an odd number of positive coordinates. They provide us with the following pairs of antipodal Bloch vectors as the measurement bases:

$$
\begin{aligned}
\boldsymbol{v}_1 &= \pm(+1, -1, -1)/\sqrt{3}, \\
\boldsymbol{v}_2 &= \pm(-1, +1, -1)/\sqrt{3}, \\
\boldsymbol{v}_3 &= \pm(-1, -1, +1)/\sqrt{3}, \\
\boldsymbol{v}_4 &= \pm(+1, +1, +1)/\sqrt{3}.
\end{aligned}
\tag{99}
$$

The qubit states corresponding to these Bloch vectors are as follows:

$$
\begin{aligned}
M_1 &= M(+1, +1), \\
M_2 &= M(+1, -1), \\
M_3 &= M(-1, +1), \\
M_4 &= M(-1, -1),
\end{aligned}
\tag{100}
$$

where

$$
M(s_1, s_2) = \left\{ \frac{1}{2}\sqrt{1 + \frac{s_1}{\sqrt{3}}} \begin{pmatrix} \sqrt{3} - s_1 \\ s_2(s_1 - i) \end{pmatrix}, \frac{1}{2}\sqrt{1 - \frac{s_1}{\sqrt{3}}} \begin{pmatrix} \sqrt{3} + s_1 \\ s_2(i - s_1) \end{pmatrix} \right\}. \tag{101}
$$

The great circles determined by these measurements partition the Bloch ball into 14 parts. In fact, the grid formed by these circles is a projection of the edges of a *cuboctahedron* (see the part on quasireglar polyhedra in Sect. 4.2.1) on the surface of the Bloch ball (see Figs. 22 and 23).

In each of the 14 parts of the Bloch sphere a definite string can be encoded so that each bit can be recovered with a probability greater than $\frac{1}{2}$. Strange as
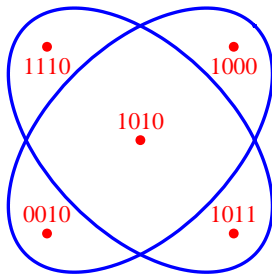
Figure 24: The relationship between the strings encoded into the spherical square and the adjacent spherical triangles according to the $4 \mapsto 1$ QRAC shown in Fig. 22.

it may seem, the remaining 2 strings ($x = 0000$ and $x = 1111$) can be encoded anywhere without affecting the success probability of this QRAC. This is not a surprise, if we recall from Sect. 3.4 that the optimal encoding $\boldsymbol{r}_x$ of the string x is a unit vector in the direction of $\boldsymbol{v}_x$ given by equation (43). In our case the Bloch vectors of the measurement bases point to the vertices of a regular tetrahedron centered at the origin. They clearly sum to zero, therefore $\boldsymbol{v}_{0000} = \boldsymbol{v}_{1111} = 0$. Thus the scalar product $\boldsymbol{r}_x \cdot \boldsymbol{v}_x$ in (42) is also zero and the success probability does not depend on the vectors $\boldsymbol{r}_{0000}$ and $\boldsymbol{r}_{1111}$. So we will ignore these two strings in the further discussion.

The other 14 strings are encoded into the vertices of a normalized *tetrakis hexahedron* (the *convex hull* of the *cube* and *octahedron*). The string $x = x_1 x_2 x_3 x_4$ is encoded into the Bloch vector $\boldsymbol{r}(x) = \boldsymbol{r}_w(x)$, where

$$w = x_1 \oplus x_2 \oplus x_3 \oplus x_4 \in \{0, 1\} \tag{102}$$

is the parity of the input. In the case $w = 0$ the encoding points are the vertices $(\pm 1, 0, 0) \cup (0, \pm 1, 0) \cup (0, 0, \pm 1)$ of an *octahedron*:

$$\boldsymbol{r}_0(x) = (-1)^{x_4} \begin{pmatrix} 1 - |x_1 - x_4| \\ 1 - |x_2 - x_4| \\ 1 - |x_3 - x_4| \end{pmatrix}. \tag{103}$$

But for $w = 1$ we get the vertices $(\pm 1, \pm 1, \pm 1)/\sqrt{3}$ of a *cube*:

$$\boldsymbol{r}_1(x) = \frac{(-1)^{x_1 x_2 + x_3 x_4}}{\sqrt{3}} \begin{pmatrix} (-1)^{x_1 + x_4} \\ (-1)^{x_2 + x_4} \\ (-1)^{x_3 + x_4} \end{pmatrix}. \tag{104}$$

Note that the Bloch vectors $\boldsymbol{r}_1(x)$ are the vertices of the same cube as the Bloch vectors of the $3 \mapsto 1$ QRAC discussed in Sect. 3.3.2.

One can observe the following properties of this encoding. The surface of the Bloch ball is partitioned into 6 *spherical squares* and 8 *spherical triangles*. Strings with $w = 0$ and $w = 1$ are encoded into squares and triangles, respectively. If $w = 1$ ($x = 1000$ or $x = 0111$ and their permutations), the string has one bit that differs from the other three. Such string is encoded into the basis state of the corresponding measurement so that this bit can be recovered with

41

certainty. If $w = 0$, the string is encoded into a square and has the following property: each of its bits takes the value that occurs more frequently at the same position in the strings of the four neighboring triangles (see Fig. 24 as an example).

The corresponding encoding function is $E(x) = \alpha_w |0\rangle + \beta_w |1\rangle$ with coefficients α_0, β_0 and α_1, β_1 explicitly given by

$$
\begin{cases}
\alpha_0 = \sqrt{\dfrac{1}{2} + (-1)^{x_4} \dfrac{1 - |x_3 - x_4|}{2}}, \\
\beta_0 = x_3 x_4 + (-1)^{x_4} \dfrac{1 - |x_1 - x_4| + i(1 - |x_2 - x_4|)}{\sqrt{2}},
\end{cases}
\tag{105}
$$

and

$$
\begin{cases}
\alpha_1 = \sqrt{\dfrac{1}{2} + \dfrac{s(x)}{2\sqrt{3}}}, \\
\beta_1 = (-1)^{x_3} s(x) \dfrac{(-1)^{x_1} + i(-1)^{x_2}}{\sqrt{6 + s(x)2\sqrt{3}}},
\end{cases}
\tag{106}
$$

where $s(x) \in \{-1, 1\}$ is given by

$$
s(x) = (-1)^{x_1 x_2 + x_3 x_4 + x_3 + x_4}.
\tag{107}
$$

The 14 coefficients β_0 and β_1 are the roots of the polynomial

$$
\beta(\beta - 1)(4\beta^4 - 1)(36\beta^8 + 24\beta^4 + 1).
\tag{108}
$$

Using input randomization we get the same success probability for any input:

$$
p = \frac{1}{2} + \frac{2 + \sqrt{3}}{16} \approx 0.7332531755.
\tag{109}
$$

It is surprising that despite higher symmetry (compare Fig. 11 and Fig. 22) it has a lower success probability than the $4 \mapsto 1$ QRAC discussed in Sect. 4.1.2.

4.2.3 Symmetric $6 \mapsto 1$ QRAC with SR

According to the discussion in Sect. 3.3.3, six great circles can cut the sphere into maximum 32 parts. It turns out that there is a symmetric way to do this. Observe that *dodecahedron* has 12 faces and diametrically opposite ones are parallel. For each pair of parallel faces we can draw a plane through the origin parallel to both faces. These six planes intersect the sphere in six great circles that define our measurements. They are the projections of the edges of *icosidodecahedron* (see Fig. 20), which is one of the quasiregular polyhedra discussed in Sect 4.2.1.

There is another way to describe these measurements. Notice that *icosahedron* (the dual of dodecahedron) has 12 vertices (six antipodal pairs). Our measurements are along the six directions defined by these pairs. The coordinates of the vertices of the icosahedron are as follows:

$$
\frac{1}{\sqrt{1 + \tau^2}}(0, \pm\tau, \pm1) \cup \frac{1}{\sqrt{1 + \tau^2}}(\pm1, 0, \pm\tau) \cup \frac{1}{\sqrt{1 + \tau^2}}(\pm\tau, \pm1, 0),
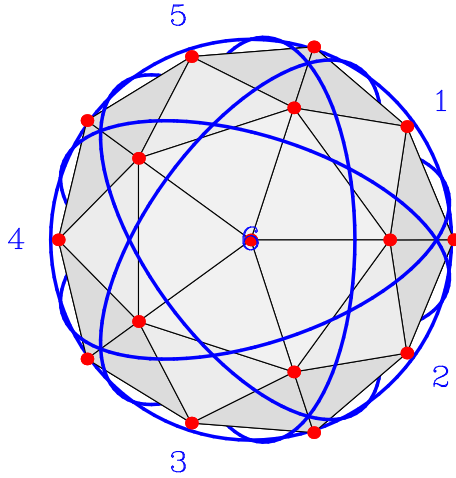\tag{110}
$$

Figure 25: Symmetric $6 \mapsto 1$ QRAC with SR.

where $\tau = \frac{1+\sqrt{5}}{2}$ is the *golden ratio* (the positive root of $x^2 = x + 1$).

Each of the 64 strings is encoded either in a vertex of an icosahedron or dodecahedron. They have 12 and 20 vertices, respectively, so there are two strings encoded in each vertex. The union of icosahedron and dodecahedron is called *pentakis dodecahedron* (see the polyhedron in Fig. 25).

The success probability of this code is

$$p = \frac{1}{2} + \frac{\sqrt{5}}{32} + \frac{1}{96}\sqrt{75 + 30\sqrt{5}} \approx 0.6940418856. \tag{111}$$

4.2.4 Symmetric $9 \mapsto 1$ QRAC with SR

This code is based on the tiling of sphere whose triangular symmetry group is $(2, 3, 4)$. The great circles corresponding to measurements coincide with the projection of the edges of *normalized disdyakis dodecahedron*. We can think of this QRAC as the union of $3 \mapsto 1$ and $6 \mapsto 1$ codes. The first three measurements are along the coordinate axis as in the $3 \mapsto 1$ QRAC discussed in Sect. 3.3.2. The remaining six measurements are exactly the same as for the $6 \mapsto 1$ code discussed in Sect. 4.1.4 (see Figs. 14 and 15), i.e., they are along the six antipodal pairs of 12 vertices of the cuboctahedron shown in Fig. 20. Note that a great circle of the first kind cannot be transformed to a great circle of the second kind via an operation from the symmetry group of the code (for the other three symmetric codes we can transform any circle to any other).

The resulting QRAC is shown in Fig. 26 and its success probability is:

$$p \approx 0.6563927998. \tag{112}$$

4.2.5 Symmetric $15 \mapsto 1$ QRAC with SR

The triangular symmetry group of this code is $(2, 3, 5)$ and the great circles coincide with the projection of the edges of *normalized disdyakis triacontahedron*. To
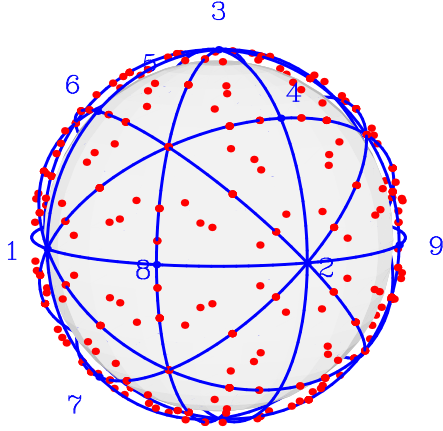
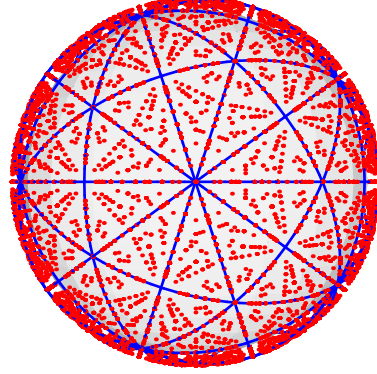Figure 26: Symmetric $9 \mapsto 1$ QRAC with SR.

Figure 27: Symmetric $15 \mapsto 1$ QRAC with SR.

understand what the measurements are in this case, note that *icosidodecahedron* (see Fig. 20) has 30 vertices. Their coordinates are:

$$(\pm 1, 0, 0) \cup (0, \pm 1, 0) \cup (0, 0, \pm 1), \tag{113}$$

$$\frac{1}{2\tau}(\pm 1, \pm\tau, \pm\tau^2) \cup \frac{1}{2\tau}(\pm\tau^2, \pm 1, \pm\tau) \cup \frac{1}{2\tau}(\pm\tau, \pm\tau^2, \pm 1). \tag{114}$$

The measurement directions are given by 15 antipodal pairs of these vertices.

The obtained QRAC is shown in Fig. 27. Its success probability is:

$$p \approx 0.6201829084. \tag{115}$$

4.3 Discussion

In this section we will compare and analyze the numerical and symmetric QRACs with SR described in Sects. 4.1 and 4.2, respectively. Hopefully these observations can be used to find new $n \mapsto 1$ QRACs with SR or to generalize the existing ones (see Sect. 5.3 for possible generalizations).

The success probabilities of numerical and symmetric QRACs with SR are given in Tables 1 and 3, respectively (see Table 6 for the comparison[9]). We see that none of the symmetric codes discussed in Sect. 4.2 is optimal. However, the success probabilities of numerical and symmetric codes do not differ much. Moreover, recall that there are two more symmetric codes ($3 \mapsto 1$ and $6 \mapsto 1$) that coincide with the numerically obtained ones (see Table 5). Concerning these two codes we can have more optimistic conclusion: $3 \mapsto 1$ QRAC is optimal (see Sect. 3.5) and possibly $6 \mapsto 1$ QRAC (see Sect. 4.1.4) is as well, since we did not manage to improve it in Sect. 4.2.3.

We just saw that symmetric QRACs are not necessarily optimal. One could ask if there are other heuristic methods that potentially could be used to construct good QRACs with SR. We will give a few speculations in the remainder

---

[9]For $n = 15$ we do not have numerical results, so we just use five measurements along each coordinate axis like for the $9 \mapsto 1$ QRAC discussed in Sect. 4.1.5.

n	Section	Probability
4	4.1.2	0.7414814566
	4.2.2	> 0.7332531755
6	4.1.4	0.6940463870
	4.2.3	> 0.6940418856
9	4.1.5	0.6568927813
	4.2.4	> 0.6563927998
15	9	0.6203554614
	4.2.5	> 0.6201829084

Table 6: Comparison of the success probabilities of $n \mapsto 1$ QRACs with SR. For each n the first probability corresponds to a numerical code, but the second one to a symmetric code.

of this section. In particular, we will discuss some special kinds of measurements that could be useful. To make the discussion more general, we will not restrict ourselves to the case of a single qubit.

Definition. Two orthonormal bases $\mathcal{B}_1$ and $\mathcal{B}_2$ of $\mathbb{C}^d$ are called *mutually unbiased bases* (MUBs) if $|\langle\psi_1|\psi_2\rangle|^2 = \frac{1}{d}$ for all $|\psi_1\rangle \in \mathcal{B}_1$ and $|\psi_2\rangle \in \mathcal{B}_2$. The maximal number of pairwise mutually unbiased bases in $\mathbb{C}^d$ is $d + 1$. [10]

When $d = 2$, equation (27) implies that Bloch vectors corresponding to basis vectors of *different* mutually unbiased bases are orthogonal[10]. There are three such bases in $\mathbb{C}^2$ and their Bloch vectors correspond to the vertices of an octahedron. For example, bases M_1, M_2, and M_3 defined in Sects. 3.3.1 and 3.3.2 are MUBs (they correspond to measuring along x, y, and z axis).

Note that the measurements for numerical $2 \mapsto 1$, $3 \mapsto 1$, $4 \mapsto 1$, and $9 \mapsto 1$ QRACs are performed entirely using MUBs and three out of five measurement bases for numerical $5 \mapsto 1$ QRAC are also MUBs.

There is another very special measurement that appears in our QRACs.

Definition. A set of d^2 unit vectors $|\psi_i\rangle \in \mathbb{C}^d$ is called *symmetric, informationally complete POVM* (SIC-POVM) if $|\langle\psi_i|\psi_j\rangle|^2 = \frac{1}{d+1}$ for any i, j. [11]

For $d = 2$ there are four such quantum states. Again, from equation (27) we see that the inner product between any two Bloch vectors corresponding to these states is $-\frac{1}{3}$. Such equiangular Bloch vectors are exactly the vertices of a tetrahedron, e.g., $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{v}_3$, $\boldsymbol{v}_4$ defined in (99). They were used in Sect. 4.2.2 to construct a symmetric $4 \mapsto 1$ QRAC.

Let us compare numerical and symmetric $4 \mapsto 1$ QRACs from Sects. 4.1.2 and 4.2.2, respectively. The first one is based on MUBs and is not very symmetric. Moreover, it looks like we are wasting one out of four bits, since two measurements are along the same direction. However, all measurement directions in the Bloch sphere are mutually orthogonal, except the ones that coincide. The second $4 \mapsto 1$ code is based on a SIC-POVM and is very symmetric. However, it looks like that in this case we are wasting two out of 16 strings, since the way we encode them does not influence the success probability.

---

[10]The notion of the Bloch vector can be generalized for $d \geq 2$ (see [12]). Then similar duality holds as well (see equation (121) in Sect. 5.3): mutually unbiased quantum states correspond to orthogonal Bloch vectors, but orthogonal quantum states correspond to "mutually unbiased" Bloch vectors, i.e., equiangular vectors pointing to the vertices of a regular simplex.

Now, if we compare the success probabilities of both $4 \mapsto 1$ codes (see Table 6), we see that the first one is clearly better. Hence we conclude that

> *orthogonality* of the measurement Bloch vectors
> seems to be more important than *symmetry*.

One can come to a similar conclusion when comparing $9 \mapsto 1$ and $15 \mapsto 1$ codes. Thus it looks like using roughly $\frac{n}{3}$ measurements along each coordinate axis is a quite good heuristic for constructing $n \mapsto 1$ QRAC with SR (see Sect. 5.2).

5   Conclusion

5.1   Summary

We study the *worst* case success probability of random access codes with shared randomness. Yao's principle (see equation (3) in Sect. 2.2) and input randomization (see Theorem 1) is applied to consider the *average* case success probability instead (this works in both classical and quantum cases).

In Sect. 2.3.2 we construct an optimal *classical* $n \mapsto 1$ RAC with SR as follows (see Theorem 2): Alice XORs the input string with n random bits she shares with Bob, computes the majority and sends it to Bob; if the ith bit is asked, Bob outputs the ith bit of the shared random string XORed with the received bit. The asymptotic success probability of this code is given by equation (17) in Sect. 2.3.2:

$$p(n) \approx \frac{1}{2} + \frac{1}{\sqrt{2\pi n}}. \tag{116}$$

The worst case success probability of an optimal *quantum* RAC with SR satisfies the following inequalities:

$$\frac{1}{2} + \sqrt{\frac{2}{3\pi n}} \leq p(n) \leq \frac{1}{2} + \frac{1}{2\sqrt{n}}. \tag{117}$$

These upper and lower bounds are obtained in Sects. 3.5 and 3.7, respectively.

Success probabilities of classical and quantum RACs are compared in Fig. 28.

5.2   Open problems on $n \mapsto 1$ QRACs

Improve the quantum lower bound. The known $2 \mapsto 1$ and $3 \mapsto 1$ QRACs and our numerical $4 \mapsto 1$ and $9 \mapsto 1$ QRACs with SR suggest that measurements in MUBs can be used to obtain good codes (see Sect. 4.3). In general we take roughly one third of measurements along each coordinate axis. Let $\boldsymbol{e}_1 = (1, 0, 0)$, $\boldsymbol{e}_2 = (0, 1, 0)$, $\boldsymbol{e}_3 = (0, 0, 1)$, and $\forall i : \boldsymbol{e}_{i+3} \equiv \boldsymbol{e}_i$. According to equation (63) the success probability of the corresponding $n \mapsto 1$ QRAC with SR is:

$$p(n) = \frac{1}{2}\left(1 + \frac{1}{2^n \cdot n} \sum_{a \in \{1, -1\}^n} \left\|\sum_{i=1}^{n} a_i \boldsymbol{e}_i\right\|\right). \tag{118}$$

Note that the sum over $\{1, -1\}^n$ is equal to

$$\sum_{i=0}^{x} \sum_{j=0}^{y} \sum_{k=0}^{z} \binom{x}{i}\binom{y}{j}\binom{z}{k} \sqrt{(x - 2i)^2 + (y - 2j)^2 + (z - 2j)^2}, \tag{119}$$
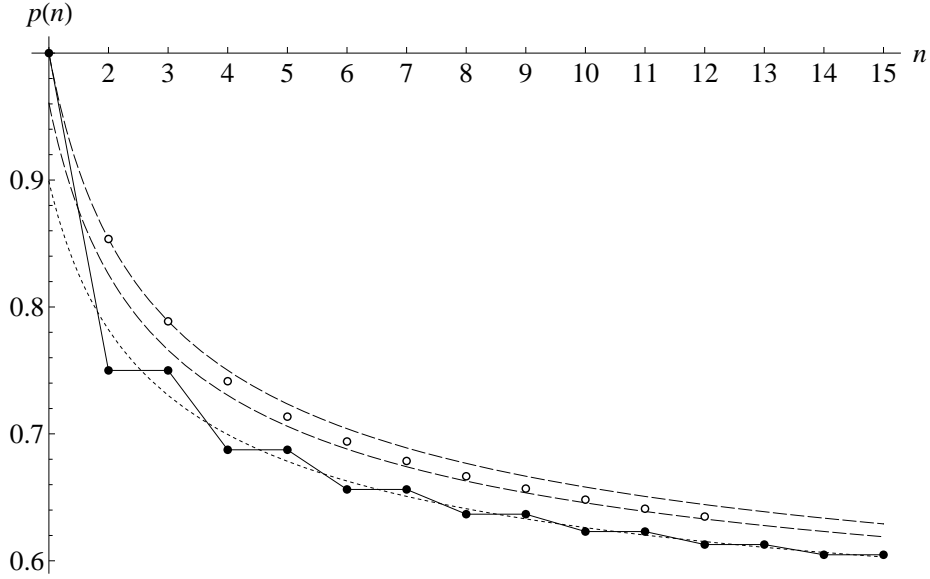
Figure 28: Comparison of success probabilities of classical and quantum RACs. Black dots correspond to optimal classical RAC and dotted line shows the asymptotic behavior. Circles correspond to numerical QRACs and dashed lines to quantum upper and lower bounds, respectively.

where $x + y + z = n$ and each of them is roughly $\frac{n}{3}$. This gives a better lower bound and also requires less shared randomness than approximations of random measurements (see Sect. 3.7). The difference of both lower bounds is shown in Fig. 29. In Fig. 30 we show how close both lower bounds and the success probabilities of numerical QRACs are to the upper bound from Sect. 3.5. Assume that Alice and Bob are given a point in the light gray region in Fig. 30 and asked to construct a QRAC with SR whose success probability is at least as good. Then they can use measurements along coordinate axis. If the point is in the dark gray region, they can use one of the numerical codes from Sect. 4.1. However, if it is in the white region, they have to solve the next open problem.

Optimality of numerical codes. Prove the optimality of any of the numerically obtained $n \mapsto 1$ QRACs with SR for $n \geq 4$ discussed in Sect. 4.1.

Prove the conjecture that quantum RACs with SR are at least as good their classical counterparts in the sense discussed at the end of Sect. 3.4.

5.3   Possible generalizations

There are several ways how our setting can be generalized both in classical and quantum case. The simplest generalization is to consider $n \overset{p}{\mapsto} 1$ RACs with SR for "d-valued bits" (called *qudits* in the quantum case) instead of the traditional bits with $d = 2$. Another direction is $n \overset{p}{\mapsto} m$ codes with $m > 1$. Of course, one can go both ways at the same time and consider $m > 1$ and $d > 2$ simultaneously. In fact, one can even use numbers in two different bases and consider $n_k \overset{p}{\mapsto} m_l$ codes, where the encoding function is $E : \{0, 1, \dots, k-1\}^n \mapsto \{0, 1, \dots, l-1\}^m$.
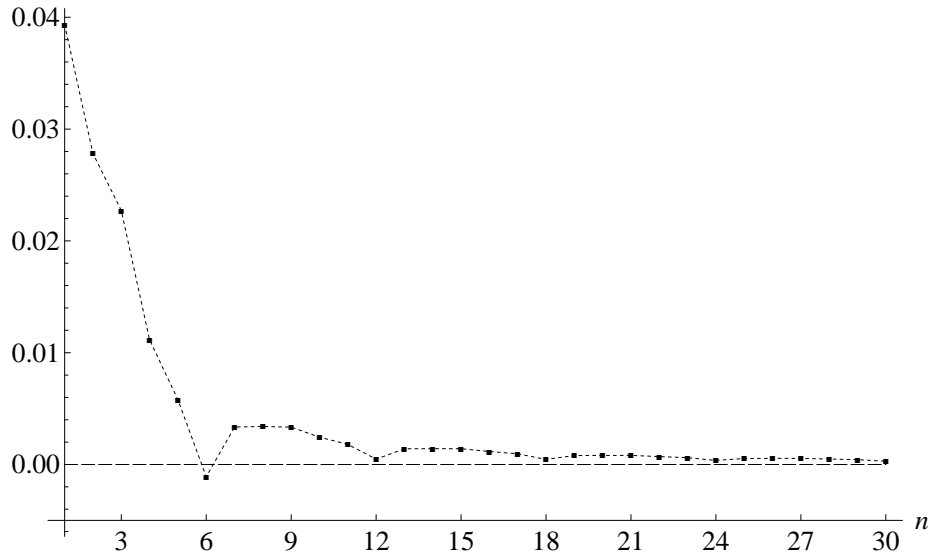
Figure 29: The *difference* of both lower bounds for QRACs with SR. Black squares and the horizontal line correspond to the bounds obtained using measurements along coordinate axes and random measurements, respectively. The first bound is better, except for $n = 6$ (notice a periodic pattern of length 6).
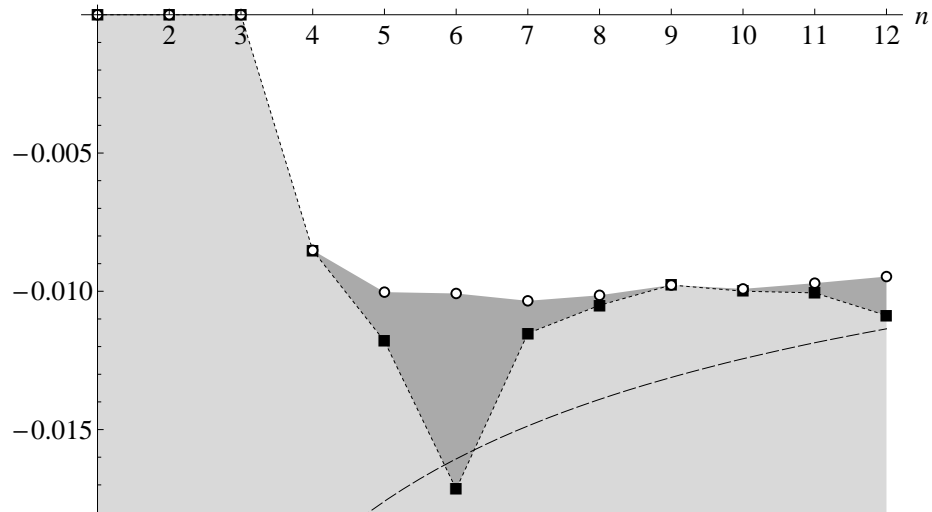

Figure 30: Close-up of the narrow region in Fig. 28 between the quantum upper and lower bound (everything is shown relative to the upper bound that corresponds to the horizontal axis). Circles indicate the gap between the upper bound and numerical QRACs with SR. Black squares show the gap between the upper bound and the lower bound by measurements along coordinate axes (see Fig. 29). Dashed line corresponds to the gap between the quantum upper bound and the lower bound by random measurements.

The notion of the Bloch vector can be generalized for $d > 2$. For example, to write down the density matrix for $d = 3$ one uses eight *Gell-Mann matrices* denoted by λ_i instead of three Pauli matrices σ_i defined in equation (25). In general one needs $d^2 - 1$ matrices λ_i that span the set of all trecaless $d \times d$ Hermitian matrices. A convenient choice of λ_i are the so called *generalized Gell-Mann matrices*, also known as the *generators of the Lie algebra of $SU(d)$*, given in [15]. We can use them to generalize equation (26):

$$\rho = \frac{1}{d}\left(I + \sqrt{\frac{d(d-1)}{2}}\boldsymbol{r} \cdot \boldsymbol{\lambda}\right), \tag{120}$$

where $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_{d^2-1})$ and $\boldsymbol{r} \in \mathbb{R}^{d^2-1}$ is the *generalized Bloch vector*[11] or *coherence vector* [12, 14]. Since λ_i are chosen so that $\mathrm{Tr}\,\lambda_i = 0$ and $\mathrm{Tr}(\lambda_i\lambda_j) = 2\delta_{ij}$, equation (27) generalizes to

$$|\langle\psi_1|\psi_2\rangle|^2 = \mathrm{Tr}(\rho_1\rho_2) = \frac{1}{d}\big(1 + (d-1)\boldsymbol{r}_1 \cdot \boldsymbol{r}_2\big). \tag{121}$$

If we want to recover a d-valued bit, we perform a measurement in an orthonormal basis $\{|\psi_1\rangle, \ldots, |\psi_d\rangle\}$ of $\mathbb{C}^d$. Since $|\langle\psi_i|\psi_j\rangle|^2 = 0$ for any pair $i \neq j$, the corresponding Bloch vectors must satisfy $\boldsymbol{r}_i \cdot \boldsymbol{r}_j = -\frac{1}{d-1}$. It means, they are the vertices of a regular simplex that belongs to a $(d-1)$-dimensional subspace and is centered at the origin (for $d = 2$ this is just a line segment).

On the other hand, in Sect. 4.3 we observed that it might be advantageous to perform measurements along orthogonal directions in the Bloch sphere to recover different bits. Let $\boldsymbol{r}_i \perp \boldsymbol{s}_j$ be two orthogonal Bloch vectors. Then the corresponding quantum states $|\psi_i\rangle$ and $|\varphi_j\rangle$ must satisfy $|\langle\psi_i|\varphi_j\rangle|^2 = \frac{1}{d}$. This is exactly the case when $|\psi_i\rangle$ and $|\varphi_j\rangle$ belong to *different* mutually unbiased bases (see Sect. 4.3). This suggests that distinct bits should be recovered using mutually unbiased measurements. Note that the Bloch vectors of the states from two MUBs correspond to the vertices of two regular simplices in mutually orthogonal subspaces. In general, the Bloch vectors of the states from all $d+1$ MUBs are the vectices of the so called *complementarity polytope* [16], which is just the octahedron when $d = 2$.

The conclusion of Sect. 4.3 and our discussion above suggests the use of MUBs to construct QRACs also for $d > 2$. However, there is a significant difference between the qubit and qudit case. Recall that for $d = 2$ the optimal way to encode the message x is to use a unit vector in the direction of $\boldsymbol{v}_x$ (see equation (43) in Sect. 3.4). Similar expression for $\boldsymbol{v}_x$ can be obtained when $d > 2$, but then the matrix ρ assigned to $\boldsymbol{r} = \boldsymbol{v}_x/\|\boldsymbol{v}_x\|$ according to equation (120) is not necessarily *positive semidefinite* and hence not a valid density matrix. However, it is known that for small enough value of $\|\boldsymbol{r}\|$ (in our case[11] $\|\boldsymbol{r}\| \leq \frac{1}{d-1}$), *all* Bloch vectors correspond to valid density matrices [13]. Hence, if we cannot use the pure state corresponding to $\boldsymbol{v}_x/\|\boldsymbol{v}_x\|$, we can always use the mixed state corresponding to $\frac{1}{d-1}\boldsymbol{v}_x/\|\boldsymbol{v}_x\|$. If one knows more about the shape of the region corresponding to valid quantum states, one can make a better choice and use a longer vector, possibly in a slightly different direction. Unfortunately, apart from being convex, not much is known about

---

[11]Our normalization follows [14], where the generalized Bloch sphere has radius 1. Another widely used convention is to assume radius $\sqrt{2(d-1)/d}$, e.g., see [12, 13].

this shape. Already for $d = 3$ it is rather involved [12, 13]. In general the conditions (in terms of the coordinates of the generalized Bloch vector $\boldsymbol{r}$) for ρ to have non-negative eigenvalues are given in [14, 12].

Finally, another way of generalizing QRACs with SR is to add other resources. A good candidate is *shared entanglement*.

A Combinatorial interpretation of sums

In this appendix we give a combinatorial interpretation of the sums in equations (13) and (14) from Sect. 2.3.2. This interpretation is formalized in the form of equations (122) and (123). We referred to these equations in Sect. 2.3.2 to obtain an exact formula (16) for the average success probability of an optimal classical RAC.

Let us consider a set of n distinct elements and count *the number of ways how to choose more that a half of n elements and mark one of them as special.* There are two approaches: first choose the elements and then mark the special one or first choose the special one and then choose the others.

In the first scenario there are $i\binom{n}{i}$ ways to choose exactly i elements and mark one of them as special. If we have to choose more than a half, we obtain the sum $\sum_{i=m+1}^{n} i\binom{n}{i}$ where $m = \lfloor \frac{n}{2} \rfloor$.

In the second scenario there are n ways to choose the special element. Then there are $l = n - 1$ elements left and at least m of them must be taken to have more than a half of n elements in total. The number of ways to do it corresponds to the number of subsets of size at least m of a set of l distinct elements. Let us consider the cases when l is odd and even separately.

If $n = 2m$ then $l = 2m - 1$ is odd. To each "large" subset of size i ($m \le i \le l$) we can assign a unique "small" subset (the complement set) of size $l - i$ ($0 \le l - i \le m - 1$), and vice versa. Each subset is either "large" or "small" thus the number of "large" and "small" subsets is the same – it is a half of the number of all subsets, i.e., $2^l/2 = 2^{l-1} = 2^{2m-2}$.

If $n = 2m + 1$ then $l = 2m$ is even. The "large" subsets have $m + 1 \le i \le l$ elements, but the "small" ones: $0 \le l - i \le m - 1$. Let us call the remaining $\binom{2m}{m}$ subsets of size m "balanced". In this case the bijection between the "large" and "small" subsets holds as well, but it maps the "balanced" subsets to themselves. Thus the total number of all subsets is "large"+"small"+$\binom{2m}{m} = 2^l$. The number of "large" subsets equals to $\left(2^l + \binom{2m}{m}\right)/2 = 2^{2m-1} + \frac{1}{2}\binom{2m}{m}$.

Both counting methods must give the same results, therefore for odd and even n we obtain, respectively:

$$\sum_{i=m+1}^{2m+1} i\binom{2m+1}{i} = (2m+1) \cdot \left(2^{2m-1} + \frac{1}{2}\binom{2m}{m}\right), \tag{122}$$

$$\sum_{i=m+1}^{2m} i\binom{2m}{i} = 2m \cdot 2^{2m-2}. \tag{123}$$

We would like to acknowledge Juris Smotrovs for providing this interpretation.
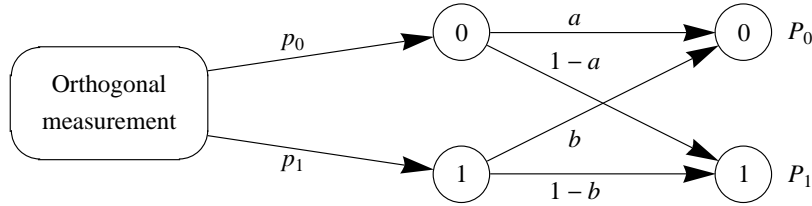
Figure 31: A simulation of the POVM measurement $\{E_0, E_1\}$ on a qubit using an orthogonal measurement and a post-processing of the measurement result.

B   POVMs versus orthogonal measurements

Orthogonal (or *von Neumann's*) measurement is not the most general type of measurement of a quantum system. In general a POVM measurement [17, 18] extracts more information. In this appendix we show that in the qubit case POVMs can be simulated using a probabilistic combination of *enhanced orthogonal measurements* defined in Sect. 3.6 (it is either an orthogonal measurement or a constant function). To define a POVM we have to introduce the notion of a positive semidefinite matrix [19].

Definition. Matrix E is called *positive semidefinite* if $\langle\psi| E |\psi\rangle \geq 0$ for all $|\psi\rangle$.

An equivalent definition is that E is diagonalizable and all eigenvalues of E are real and non-negative. Thus E is Hermitian.

Definition. *Positive operator–valued measure* (POVM) is a set $\{E_1, \ldots, E_m\}$ of positive semidefinite matrices such that $\sum_{i=1}^{m} E_i = I$. [17, 18]

POVM measurements can have arbitrary number of outcomes, but in the case of $n \mapsto 1$ QRACs we have to consider only qubit POVMs. Moreover, it is enough to consider POVMs with just two outcomes: 0 and 1. Such a POVM can be specified by $\{E_0, E_1\}$, where E_0 is positive semidefinite and $E_1 = I - E_0$. Since E_0 is also Hermitian, we can find a basis $\mathcal{B} = \{|\psi_0\rangle, |\psi_1\rangle\}$ in which E_0 is diagonal, i.e., $E_0 = \left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right)$. In this basis $E_1 = \left(\begin{smallmatrix} 1-a & 0 \\ 0 & 1-b \end{smallmatrix}\right)$. Since both E_0 and E_1 are positive semidefinite, $0 \leq a \leq 1$ and $0 \leq b \leq 1$. An arbitrary pure qubit state $|\psi\rangle$ in basis $\mathcal{B}$ can be specified by (21). When it is measured, the probabilities of outcomes are

$$
\begin{cases}
P_0 = \langle\psi| E_0 |\psi\rangle = a\cos^2\dfrac{\theta}{2} + b\sin^2\dfrac{\theta}{2}, \\
P_1 = \langle\psi| E_1 |\psi\rangle = (1-a)\cos^2\dfrac{\theta}{2} + (1-b)\sin^2\dfrac{\theta}{2}.
\end{cases}
\tag{124}
$$

Let us consider the following process (see Fig. 31) that simulates the POVM measurement $\{E_0, E_1\}$:

1. perform an orthogonal measurement in basis $\mathcal{B} = \{|\psi_0\rangle, |\psi_1\rangle\}$,

2. perform the following post-processing of the outcome of the measurement:

51

- *if the outcome was* 0: output 0 with probability a, output 1 with probability $1 - a$,

- *if the outcome was* 1: output 0 with probability b, output 1 with probability $1 - b$.

To see why this process is equivalent to the POVM measurement $\{E_0, E_1\}$, consider a pure qubit state $|\psi\rangle$ given by (21) in basis $\mathcal{B}$. When it is measured on the basis vectors $\{|\psi_0\rangle, |\psi_1\rangle\}$ of base $\mathcal{B}$, the probabilities of outcomes 0 and 1 are as follows (see also equation (28) in Sect. 3.1.1):

$$\begin{cases} p_0 = |\langle\psi_0|\psi\rangle|^2 = \cos^2\dfrac{\theta}{2}, \\ p_1 = |\langle\psi_1|\psi\rangle|^2 = \sin^2\dfrac{\theta}{2}. \end{cases} \tag{125}$$

Now it is simple to verify that the process shown in Fig. 31 has the same outcome probabilities (124) as the POVM measurement. However, this process cannot be considered as a probabilistic combination of enhanced orthogonal measurements, since it involves a probabilistic post-processing of the measurement result. To obtain the desired result, we have to modify it. The key idea is that with a certain probability the output can be produced without performing an actual measurement.

Let $\mu = \min\{a, b\}$. Whatever state is input to the process shown in Fig. 31, the probability P_0 to output 0 is at least μ, because

$$P_0 = p_0 a + p_1 b \geq (p_0 + p_1)\mu = \mu. \tag{126}$$

Note that μ does not depend on the state being measured. It means, one can output 0 with probability μ without performing an actual measurement. A similar lower bound holds for P_1 as well:

$$P_1 = p_0(1 - a) + p_1(1 - b) \geq (p_0 + p_1)(1 - M) = 1 - M, \tag{127}$$

where $M = \max\{a, b\} = a + b - \mu$. Let us consider the following probabilistic combination of four decoding strategies:

- *with probability* c_0: output 0 without performing a measurement,

- *with probability* c_1: output 1 without performing a measurement,

- *with probability* c_{01}: measure in the basis $\{|\psi_0\rangle, |\psi_1\rangle\}$,

- *with probability* c_{10}: measure in the opposite basis $\{|\psi_1\rangle, |\psi_0\rangle\}$.

The resulting probabilities of outcomes for this process are

$$\begin{cases} P_0 = c_0 + c_{01}p_0 + c_{10}p_1, \\ P_1 = c_1 + c_{01}p_1 + c_{10}p_0. \end{cases} \tag{128}$$

We can use the lower bounds (126) and (127) for P_0 and P_1, respectively, to assign the probabilities c_0, c_1, c_{01}, and c_{10} in the following way:

$$\begin{cases} c_0 = \mu, \\ c_1 = 1 - (a + b) + \mu, \\ c_{01} = a - \mu, \\ c_{10} = b - \mu \end{cases} \tag{129}$$

(note that at least one of probabilities c_{01} or c_{10} will be zero). It is not hard to verify that after the assignment (129) the probabilities P_0 and P_1 in (128) will match the probabilities of outcomes (124) of the POVM measurement.

Thus for each qubit POVM given by a and b one can find a probabilistic combination of enhanced orthogonal measurements given by c_0, c_1, c_{01}, and c_{10}, such that in both cases the probabilities of outcomes are the same.

Example. For $a = b = 1/2$ we have $c_0 = c_1 = 1/2$ and $c_{01} = c_{10} = 0$ that corresponds to a random guessing (observe that $E_0 = E_1$ in this case).

Example. However, $a = 1$ and $b = 0$ corresponds to a projective measurement in basis $\{|\psi_0\rangle, |\psi_1\rangle\}$, because $c_{01} = 1$ and $c_{10} = c_0 = c_1 = 0$.

Example. Finally, $a = 1$ and $b = 1$ corresponds to a constant function 0, because $c_0 = 1$ and $c_{01} = c_{10} = c_1 = 0$.

References

[1] Andris Ambainis, Ashwin Nayak, Amnon Ta-Shma, Umesh Vazirani, "Dense Quantum Coding and Quantum Finite Automata," *Journal of the ACM*, vol. 49, no. 4, pp. 496–511, 2002. arXiv:quant-ph/9804043v2

[2] Masahito Hayashi, Kazuo Iwama, Harumichi Nishimura, Rudy Raymond, Shigeru Yamashita, "$(4, 1)$-Quantum Random Access Coding Does Not Exist," *New J. Phys.*, vol. 8, 129, 2006. arXiv:quant-ph/0604061v1

[3] Yao A.C., "Probabilistic computations: towards a unified measure of complexity," Proceedings 18th Annual IEEE Symposium on Foundations of Computer Science, October 1977, 222–227.

[4] Giuliano Benenti, Giulio Casati, Giuliano Strini, "Principles of Quantum Computation and Information," vol. 1, World Scientific.

[5] Eric W. Weisstein, "Stirling's Approximation," MathWorld. http://mathworld.wolfram.com/StirlingsApproximation.html

[6] Eric W. Weisstein, "Sphere Point Picking," MathWorld. http://mathworld.wolfram.com/SpherePointPicking.html

[7] Chandrasekhar S., "Stochastic Problems in Physics and Astronomy," *Reviews of Modern Physics*, vol. 15, no. 1, 1943.

[8] Larry C. Grove, Clark T. Benson, "Finite Reflection Groups", 2nd Ed., Springer, 1985.

[9] Rouse Ball W.W., Coxeter H.S.M., "Mathematical Recreations and Essays," 13th Ed., Courier Dover Publications, 1987.

[10] William K. Wootters, Brian D. Fields, "Optimal State-Determination by Mutually Unbiased Measurements," *Annals of Physics*, vol. 191, Issue 2, pp. 363–381, 1989.

[11] Joseph M. Renes, Robin Blume-Kohout, A. J. Scott, Carlton M. Caves, "Symmetric Informationally Complete Quantum Measurements," *J. Math. Phys.*, vol. 45, pp. 2171–2180, 2004. arXiv:quant-ph/0310075v1

[12] Gen Kimura, "The Bloch vector for N-level systems," *Physics Letters A*, vol. 314, Issues 5–6, pp. 339–349, 2003. `arXiv:quant-ph/0301152v2`

[13] Gen Kimura, Andrzej Kossakowski, "The Bloch-Vector Space for N-Level Systems: the Spherical-Coordinate Point of View," *Open Syst. Inf. Dyn.*, vol. 12, no. 3, pp. 207–229, 2005. `arXiv:quant-ph/0408014v2`

[14] Mark S. Byrd, Navin Khaneja, "Characterization of the Positivity of the Density Matrix in Terms of the Coherence Vector Representation," *Phys. Rev. A*, vol. 68, Issue 6, 062322, 2003. `arXiv:quant-ph/0302024v2`

[15] Hioe F.T., Eberly J.H., "N-Level Coherence Vector and Higher Conservation Laws in Quantum Optics and Quantum Mechanics," *Phys. Rev. Lett.*, vol. 47, pp. 838–841, 1981.

[16] Ingemar Bengtsson, Åsa Ericsson, "Mutually Unbiased Bases and the Complementarity Polytope," *Open Syst. Inf. Dyn.*, vol. 12, no. 2, pp. 107–120, 2005. `arXiv:quant-ph/0410120v1`

[17] Michael A. Nielsen, Isaac L. Chuang, "Quantum Computation and Quantum Information," Cambridge University Press, 2000.

[18] Asher Peres, "Quantum Theory: Concepts and Methods," Kluwer Academic Publishers, 2002.

[19] Roger A. Horn, Charles R. Johnson, "Matrix Analysis," Cambridge University Press, 1985.