University of BRISTOL

Instantaneous Quantum Computation

Dan Shepherd
and Michael Bremner

arXiv:0809.0847

QICS

arXiv:0809.0847

Sampling strings

Temporally Unstructured Quantum Computation,

a.k.a. Instantaneous Quantum Computation.

- Hamiltonian

$$H = \theta \cdot (X_1 X_4 + X_1 X_2 X_5 + X_2 X_5)$$

- Probability Distribution

$$\mathbb{P}(\mathbf{X} = \mathbf{x}) \;=\; \left| \langle\, \mathbf{x} \,|\, \exp\left(i \cdot H\right) \,|\, \mathbf{0} \,\rangle \right|^2$$

- What sort of quantum computing paradigm might be suitable for sampling from this kind of probability distribution?

- To what sort of use might we put this kind of string-sampling computational power?

IQP oracle

<u>Definition:</u>

An IQP oracle is any device that samples the random variable **X** on input matrix P and action angle θ.

- 'Instantaneous' = No inherent temporal structure in the computations used to render a sample
- 'Quantum' = Seems to require quantum mechanics to render a sample from an IQP distribution
- 'Polynomially bounded' = We won't allow more than poly(n) terms in the Hamiltonian, to keep things 'realistic'

IQP oracle

Alternative representation :

$$H = \theta \cdot (X_1 X_4 + X_1 X_2 X_5 + X_2 X_5)$$

$$P = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

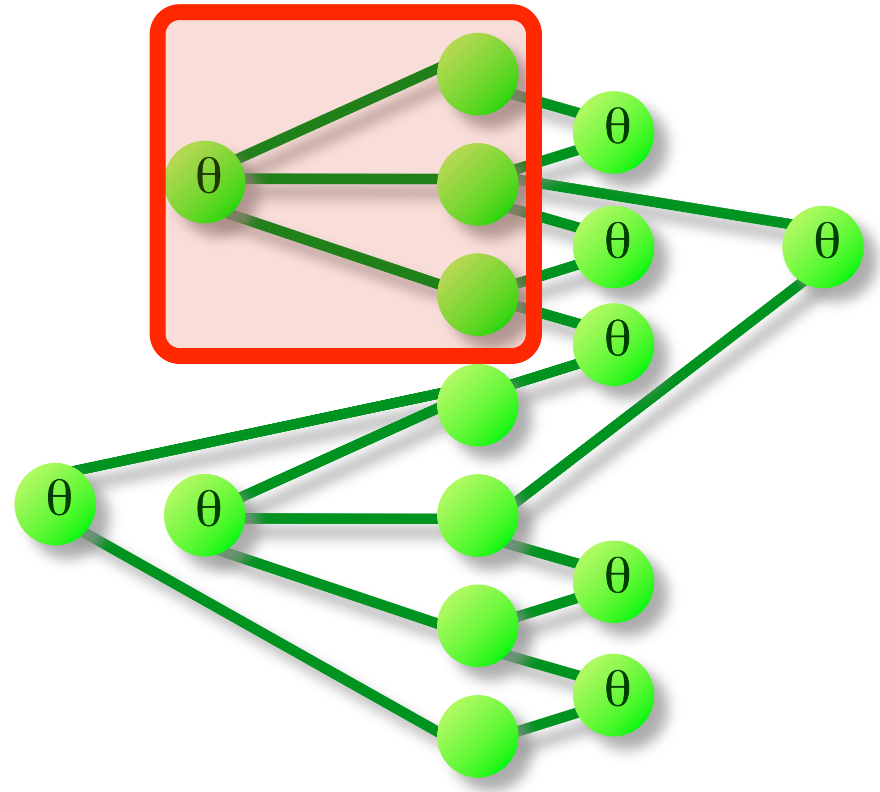
One could allow different θ for different rows, in general

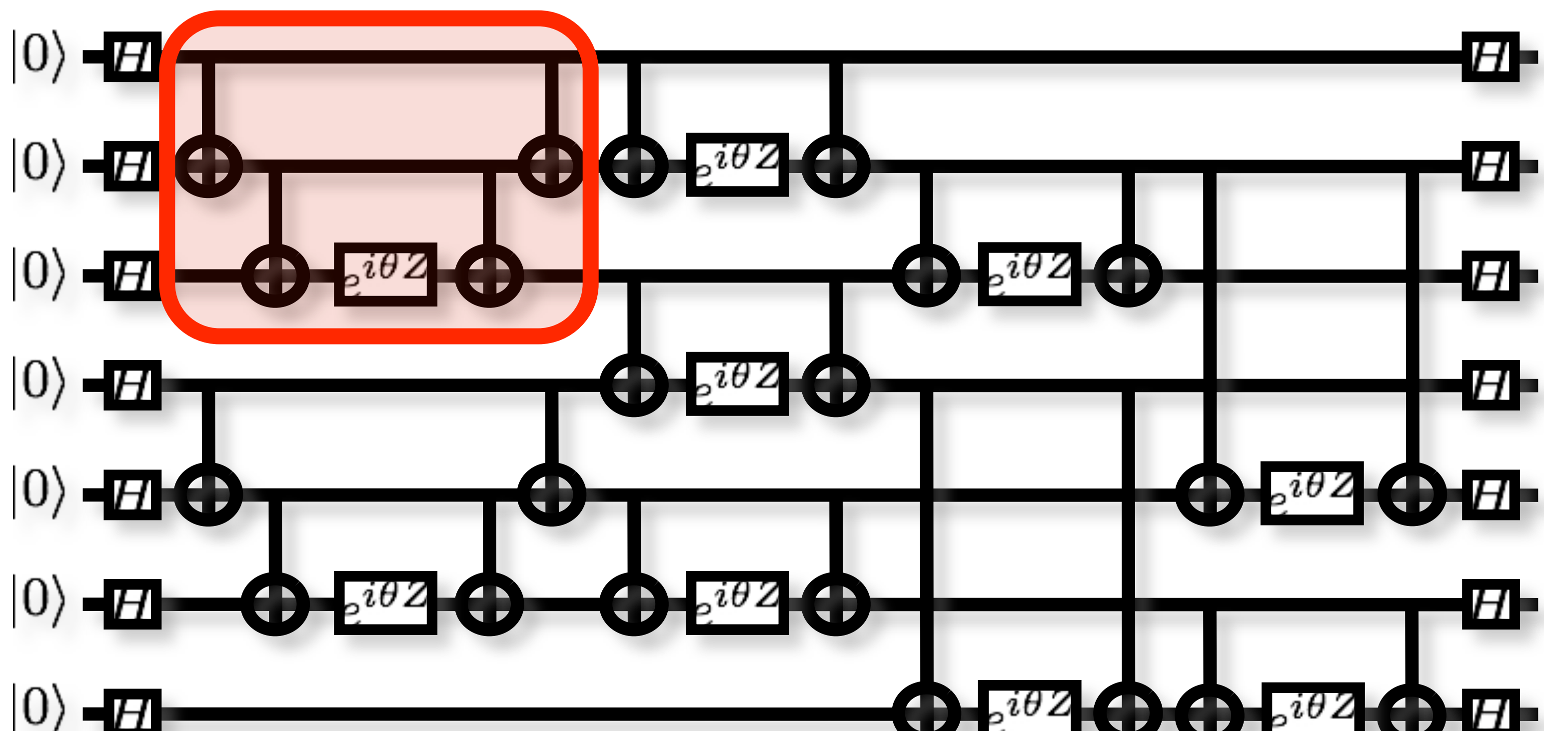
$$\mathbb{P}(\mathbf{X} = \mathbf{x}) := \left| \langle \mathbf{x} | \exp \left(\sum_{\mathbf{p}} i\theta_{\mathbf{p}} \bigotimes_{j: p_j = 1} X_j \right) | \mathbf{0}^n \rangle \right|^2$$

Implementing IQP

Different architectures are possible...

$$P = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$





Two-Party protocol

(all signalling is classical)

Alice : “Here’s a matrix P, and an angle θ; try using those.”

Bob :“Okely dokely. [...time passes...] Here’s a big pile of independently generated sample strings.”

Alice : “Thanks Bob, now give me a minute while I decide whether you really are a functioning IQP oracle...”

Cryptographic analogy

- Think of Alice's matrix P as a 'public key'. When she creates it, she should also create a corresponding 'private key'.
- At the final stage of the protocol, Alice outputs a single bit to say whether she believes Bob's claim to be IQP-capable. She can use her 'private key' when making this decision : it tells her what features of Bob's data to focus on verifying.
- Bob might not have any 'quantum ability', but still want to cheat. He cheats successfully if he can send *any* data that would cause Alice to accept with non-negligible probability. But because Bob doesn't know Alice's 'private key', there's no generically easy way for him to make fake data with the right kind of 'signal' in it.

Goals of this talk

We show formally :

- A construction for Alice to use in generating a ‘public’ matrix and a `private key’,
- A classical hypothesis test for Alice to implement when verifying Bob’s data,

We indicate heuristically :

- Why Bob probably can’t find the ‘private key’ efficiently,
- Why Bob probably can’t use classical techniques to make a dataset that would stand a good chance of convincing Alice.

Basic features of IQP

Choosing θ :

$$\mathbb{P}(\mathbf{X}=\mathbf{x}) := \left| \langle \mathbf{x} | \exp\left(\sum_{\mathbf{p}} i\theta_{\mathbf{p}} \bigotimes_{j:p_j=1} X_j \right) | \mathbf{0}^n \rangle \right|^2$$

- If θ is a multiple of π/4, then the Gottesman-Knill theorem applies.
- All our constructions use θ = π/8.
- For θ = π/8, there seems to be a classical technique that goes some way to approximating the IQP distribution, but not close enough to allow Bob to cheat.

Basic features of IQP

Consider the previous example...

$$H = \theta \cdot (X_1 X_4 + X_1 X_2 X_5 + X_2 X_5)$$

$$P = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Note that the third qubit is not touched by the Hamiltonian...

So any output sample must be orthogonal to this direction :

$$\mathbf{s} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\mathbb{P}(\mathbf{X} \cdot \mathbf{s} = 0) \ = \ 1$$

Equivalently, this *bias probability in direction s* is unaffected by any row of P that is orthogonal to s. (In this example, they all are.)

Linear codes

The matrix P can be thought of as a generator of a linear code. If some rows are deleted, one obtains a 'punctured code', which has codewords of a shorter length. A punctured code may have a smaller rank.

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

k

Rank

Columns form a spanning set for the code

<u>Defn</u> (linear binary code):

C is a code of length *k* is a subspace of the vectorspace $\mathbb{F}_2^k$.

The elements of *C* are called codewords, and the Hamming weight *wt(c)∈[0...k]* of some *c∈C* is the number of 1's it has. The rank of *C* is its rank as a vector space.

Probability bias

Theorem:

- For any direction **s**, the bias expression $\mathbb{P}(\boldsymbol{X}.\boldsymbol{s} = 0)$ for the IQP random variable ***X*** depends only on the action θ (assumed constant) and the weight enumerator polynomial of the binary code $C_{\mathbf{s}}$ obtained by deleting rows of *P* that are orthogonal to **s**.
- The following formula expresses this bias :

$$\mathbb{P}(\mathbf{X} \cdot \mathbf{s} = 0) = \mathbb{E}_{\mathbf{c} \sim \mathcal{C}_{\mathbf{s}}} \left[\cos^2 \left(\theta (n_{\mathbf{s}} - 2.wt(\mathbf{c})) \right) \right]$$

- (n_s = number of rows left, after deletion = length of $C_{\mathbf{s}}$)

Alice's construction

Here is a generator matrix of a length 7 *quadratic residue code* :

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

q=7

$(q+1)/2=4$

$$P_{\mathbf{s}} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Adding all ones vector does not change the code

$$\mathbf{s} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Obfuscation

We now want to define a *P* which has C_s as a hidden code.

- If we add *q* points (or rows) which are random except having a 0 in the leftmost column then every new row will be orthogonal to **s**.

$$P_{\mathbf{s}} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Recall that reordering rows has no effect on the program.

$$P = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

$$\mathbf{s} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Obfuscation

We now want to define a *P* which has C_s as a hidden code.

- If we add *q* points (or rows) which are random except having a 0 in the leftmost column then every new row will be orthogonal to **s**.

$$P = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Recall that reordering rows has no effect on the program.

$$\mathbf{s} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$P = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Hiding matroids

Column echelon reduction removes extraneous information.

- It changes the direction **s** to (01110).
- This direction should be kept **secret**.
- For quadratic residue codes, we deduce
  - $\mathbb{P}(\mathbf{X.s}=0) = \cos^2(\pi/8) = 85.4\%$
- Alice will test this bias to decide.
- Imagine a much bigger version of this!

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix}$$

Conjecture:

We conjecture that it is NP-hard to find a hidden submatroid of this form when only given matroid P. (Cf. Subgraph Isomorphism problem.)

Alice's verification

Recall that Alice's goal all along has been to verify whether Bob's device can do something quantum.

- All that she is testing for is whether or not samples that are *orthogonal* to **s** occur with the correct probability.
- Note that she is not calculating the probability of specific strings occuring! She is simply taking the inner product between her **s** and everything that Bob sends her.
- If Alice is to believe that her test is truly verifying that something quantum is going on she has to believe
  - that IQP is not classically simulable
  - that **s** is really truly hidden

Faking it classically

Here is a recipe for the best general classical strategy **Y** that we could find for approximating the IQP distribution **X** :

- Pick two directions, **d**, **e**, at random
- Add together all the rows of P orthogonal to neither **d** nor **e**
- Return the sum, **y**

<u>Theorem:</u>

- For the case θ = π/8, for all directions **s**,

$$\mathbb{P}(\mathbf{Y} \cdot \mathrm{s}) = (1 + 2^{-rank(P_{\mathrm{s}}^{T} \cdot P_{\mathrm{s}})})/2$$

- For our quadratic residue codes, for the secret **s**,
  - $\mathbb{P}$(**Y.s**=0) = 3/4 = 75%

Decision languages?

Two-party protocols based on probabilistic sampling are all very well, but can IQP extend complexity classes?

- **BPP^{IQP} = BPP ?**

This might well be true, even though IQP be hard to simulate classically. One possible reason is that very strong probability biases *are* easy to simulate.

<u>Theorem:</u>

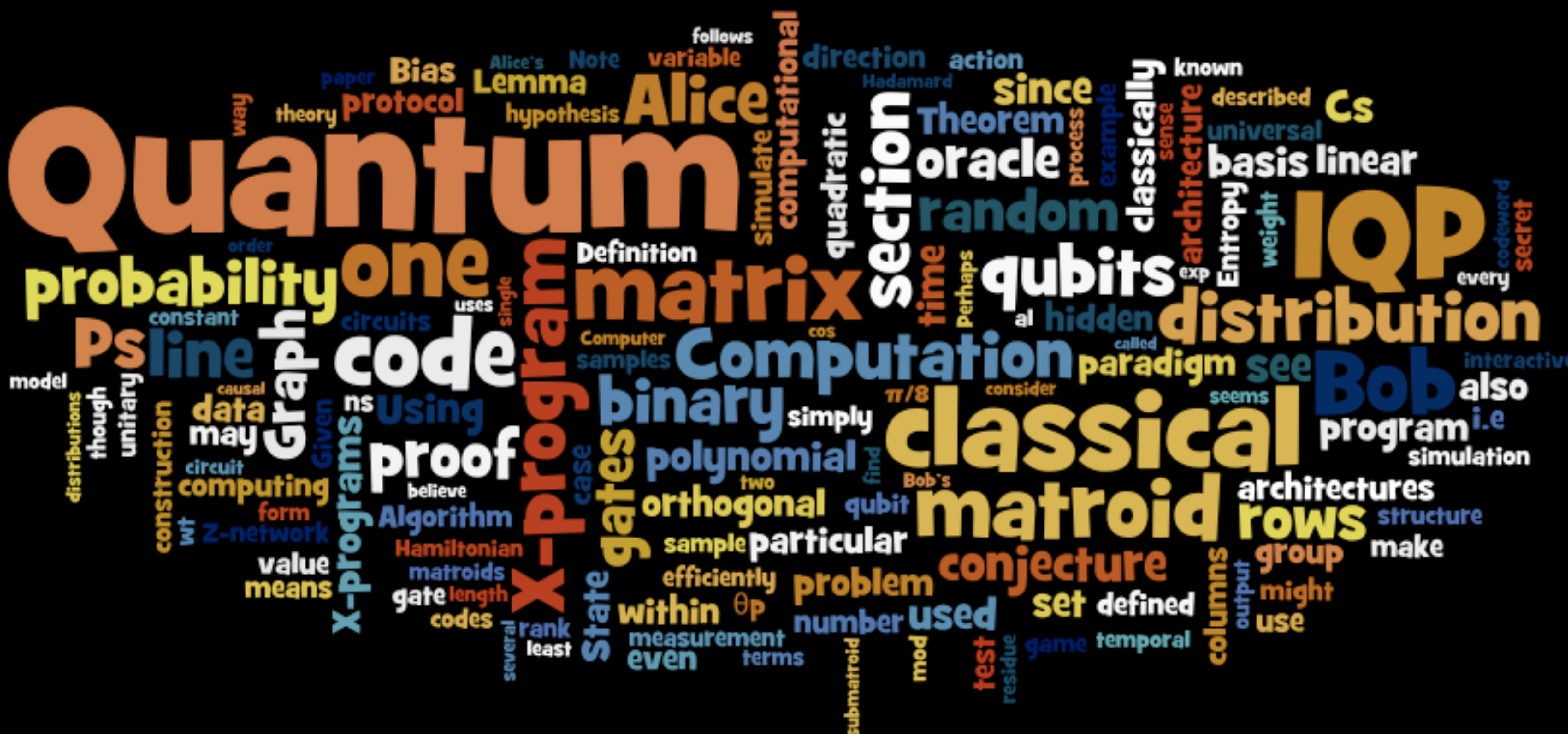
- For the case $\theta = \pi/8$, for all directions **s**,

$$\mathbb{P}(\mathbf{X} \cdot \mathrm{s}) = 1 \;\Rightarrow\; \mathbb{P}(\mathbf{Y} \cdot \mathrm{s}) = 1$$

Future directions

- Clean up as many of the conjectures in the paper as we can.
  - We'd especially like to prove the complexity of our procedure for hiding **s**.
  - Also we'd like some more arguments about the difficulty of classically faking Alice's test.
  - Prove that there are no new BPP decision languages when IQP is allowed as an oracle.
- Explore the idea of weighted matroids, where θ varies.
- Work on some implementations of IQP processors.

Check out http://quantumchallenges.wordpress.com



arXiv:0809.0847