Universal Gates in Other Universes

Jonathan A. Poritz

Department of Mathematics and Physics
Colorado State University – Pueblo
2200 Bonforte Blvd.
Pueblo, CO 81001
USA
jonathan.poritz@gmail.com
http://www.poritz.net/jonathan

Abstract. I describe a new formalization for computation which is similar to traditional circuit models but which depends upon the choice of a family of [semi]groups – essentially, a choice of the structure group of the universe of the computation. Choosing the symmetric groups results in the reversible version of classical computation; the unitary groups give quantum computation. Other groups can result in models which are stronger or weaker than the traditional models, or are hybrids of classical and quantum computation.
One particular example, built out of the semigroup of doubly stochastic matrices, yields classical but probabilistic computation, helping explain why probabilistic computation can be so fast. Another example is a smaller and entirely $\mathbb{R}$eal version of the quantum one which uses a (real) rotation matrix in place of the (complex, unitary) Hadamard gate to create algorithms which are exponentially faster than classical ones.
I also articulate a conjecture which would help explain the different powers of these different types of computation, and point to many new avenues of investigation permitted by this model.

Keywords: reversible computation, quantum computation, structure group, universal families of gates, unitary groups, symmetric groups, circuit models of computation, probabilistic computation, exponential speed-up

1 Introduction

The foundations of quantum mechanics place unitary groups, and the Hilbert spaces on which they act, in a surprisingly central location. The complex linear algebraic structure, Hermitian form, and group of transformations that preserve this structure are certainly unexpected when compared to classical mechanics, where phase spaces are symplectic manifolds and the passage of time is a symplectomorphism.

* Appeared as G.W. Dueck and D.M. Miller (Eds.): RC 2013, LNCS7948, pp. 155-167 2013.

When quantum computation arrived on the scene, it posited wires with quantum states in a Hilbert space interacting in gates which were unitary transformations in the tensor product of the Hilbert spaces of the input wires. This was again a striking contrast with classical circuit models of computation with values from a finite alphabet on the wires, the alphabet being $\mathbb{Z}_2$ in the simplest case or some other finite alphabet when the circuit is viewed as embodying a Turing machine.

In this paper, I describe a new model of circuits with very general gates which allows a uniform description of several important issues in computation. In particular, the gates are to lie in a structure group – really, a system of groups – as specified below, with the wires carrying elements in a vector spaces on which representations of the structure groups act.

The first examples I consider are when the structure groups are permutation groups, which encompasses classical (reversible) computation. What is then very interesting is that I can use even a semigroup, and when the semigroup is the Birkhoff polytope of doubly stochastic matrices, the model implements classical probabilistic/non-deterministic computation. There are the first hints here, to my knowledge, of why probabilistic computation can be faster than deterministic.

After this, I proceed to the unitary structure group, which gives just standard quantum computation. However, the simple algebraic structure of the representation of the unitary group which allows destructive interference to create exponential speed-up over classical approaches can also be realized in a purely real situation which I then describe. In particular, using the special orthogonal structure group and a gate in $SO(2)$ which has a very similar form to the Hadamard gate of quantum computation, I provide an explicit example of such an exponential speed-up.

As Blum, Cucker, Shub and Smale [5] brought techniques of ring theory and algebraic geometry to a computation model which generalized classical computation, and Coecke and many co-workers [7] brought category theory and graphical calculi, the goal of the current work is to bring techniques of [Lie] group theory and (eventually) functional analysis to a different generalization.

1.1 Notation

Here are some basic notations and assumptions I use in this entire paper:

- $\mathbb{F}$ is a fixed field.
- Hilbert spaces are always assumed to be defined over $\mathbb{F}$. When $\mathbb{F} = \mathbb{R}$, we shall use the term *Hilbert space* to refer to a complete (real) inner product space; When $\mathbb{F} = \mathbb{C}$, inner products are always Hermitian.
- No matter the base field $\mathbb{F}$, we shall use the adjective *unitary* to mean a transformation which preserves the inner product. For example, where we say *unitary representation* below, if $\mathbb{F} = \mathbb{R}$ this is what is usually called an *orthogonal representation*.

- Linear transformations are always assumed to be bounded, when defined on an infinite dimensional Hilbert space.
- For a Hilbert space V, $\mathrm{Aut}(V)$ denotes the (bounded) linear self-maps – in particular, even if V has an inner product, $\mathrm{Aut}(V)$ does not contain only isometries – and likewise for maps between different Hilbert spaces.
- Σ will be a fixed finite alphabet, often used when we need to do symbolic computation on strings; it usually suffices to use $\Sigma = \{0, 1\}$.

2 The Computational Model

In this section, I present the new generalized computational model ... but first, some foundations. The goal is to replace classical logical gates acting on bits or the unitary group acting on qubits with a general [semi]group action on a vector space. Therefore, *wires* in circuits (or *cells* in a Turing machine tape) will have values in this vector space.

2.1 Generalizing gates and wires

The operations in our computations come from the following:

Definition 1. *Let V a vector space and for each $k \in \mathbb{N}$ let G_k be a group (resp., semigroup) and $\rho_k : G_k \to \mathrm{End}(V^{\otimes k})$ a homomorphism. We shall call the collection $\mathcal{G} = \{G_k, \rho_k \mid k \in \mathbb{N}\}$ a* **system of groups** *(resp.,* **semigroups***)* **acting on [the tensor powers of]** *V.*

Here are the most basic examples:

Example 1. Fix $d \in \mathbb{N}$ and let $V = \mathbb{F}^d$. Then there is the usual representation of the symmetric group S_d on V as permutation matrices (permuting the standard basis vectors), and of S_{d^k} on $V^{\otimes k}$ in the same way.

Using $G_k = S_{d^k}$ and this standard representation results in the **system $\mathcal{S}ym$ of symmetric groups acting on [the tensor powers of]** $V = \mathbb{F}^d$.

Example 2. Again fix $d \in \mathbb{N}$ and let $V = \mathbb{F}^d$. A very general system of groups acting on V is **the system $\mathcal{G}l$ of general linear groups acting on** V given by the full general linear group $GL(d^k, \mathbb{F})$ acting on $V^{\otimes k}$ under the usual representation.

There is an interesting example which mixes these first two, built out of the following.

Definition 2. *An $n \times n$ matrix over $\mathbb{R}$ is called* **doubly stochastic** *if all the column- and row-sums equal 1. The set of such matrices is denoted $\mathbf{B}_n$ and called the* **Birkhoff polytope**.

The Birkhoff-von Neumann Theorem states that the convex hull of the permutation matrices, viewed as sitting in $\mathbb{R}^{n^2}$, is exactly the Birkhoff polytope $\mathbf{B}_n$ (see, *e.g.,* the original [4] or a more modern [14]). A simple calculation shows that $\mathbf{B}_n$ is a semigroup but not a group: it contains the identity and all products, but not inverses.

Example 3. The **Birkhoff-von Neumann system of semigroups $\mathcal{B}v\mathcal{N}$ acting on** $V = \mathbb{R}^2$ is the system which has the semigroup $\mathbf{B}_{2^d}$ acting on $V^{\otimes d}$ by the standard representation.

Going instead in the direction of complexity, we have

Example 4. Fix $d \in \mathbb{N}$ and now let $V = \mathbb{C}^d$. The group of unitary matrices $U(d)$ acts on V and the group $U(d^k)$ acts on $V^{\otimes k}$ by the natural representation. This collection of groups and representations will be called the **system $\mathcal{U}$ of unitary groups acting on [the tensor powers of]** $V = \mathbb{C}^d$ or, if we restrict to matrices of determinant one, the **system $\mathcal{SU}$ of special unitary groups**.

In exactly the same way, if $\mathbb{F} = \mathbb{R}$, we get the **system $\mathcal{SO}$ of special orthogonal groups acting on [the tensor powers of]** $V = \mathbb{R}^d$.

Let us first fix a notation with tensor products and corresponding suggestive terminology which will be useful when we want to promote an endomorphism of a tensor power of a vector space to one of a higher tensor power of that space.

Definition 3. *Suppose V is a vector space over $\mathbb{F}$ and $j, k, l \in \mathbb{N}$ satisfy $j - 1 + l \leq k$. If $T \in \mathrm{End}(V^{\otimes l})$, we use T on the l factors of $V^{\otimes k}$ starting with the jth to define an endomorphism $T^{\wedge j} = \mathrm{Id}^{\otimes(j-1)} \otimes T \otimes \mathrm{Id}^{\otimes(k-j+1-l)}$ of $V^{\otimes k}$ which we shall call the T-**gate starting at j operating on** $V^{\otimes k}$.*

[Here the associativity of the tensor product is crucial.]

Once we have a system of [semi]groups acting on the tensor powers of a vector space, we can let a set of its elements act on n-tuples of "wires" in an appropriate sense.

Definition 4. *Given a system $\mathcal{G} = \{G_k, \rho_k\}$ of groups acting on the vector space V and $n, m \in \mathbb{N}$, a* **circuit $\mathcal{C}$ with structure group $\mathcal{G}$ on n wires using m gates** *is a sequence $((g_1, j_1, d_1), \ldots, (g_m, j_m, d_m))$ of triples of the form (g, j, d) where $g \in G_d$ and $j, d \in \mathbb{N}$ satisfy $j - 1 + d \leq n$.*

Such a circuit induces a map, for which we shall also use the same symbol,

$$\mathcal{C} : V^{\otimes n} \to V^{\otimes n} : v \mapsto \rho_{d_m}(g_m)^{\wedge j_m} \cdot \cdots \cdot \rho_{d_1}(g_1)^{\wedge j_1}(v)$$

called the **computation with $\mathcal{C}$ on vectors** *(or* **raw computation with $\mathcal{C}$***); this element $\rho_{d_m}(g_m)^{\wedge j_m} \cdot \cdots \cdot \rho_{d_1}(g_1)^{\wedge j_1} \in \mathrm{End}(V^{\otimes k})$ will be called* **the program implemented by the circuit $\mathcal{C}$**.

Note 1. These circuits use wires in a slightly different way from some other authors (see, *e.g.,* [6]), where sometimes gates are permitted to operate on any input wires, in any order, selected from all the available wires. The difference is that in the approach of this paper, some permutation gates would have to be used first to bring these wires together, and $\mathcal{G}$ would have to be large enough to contain these permutations as well.

We need some additional information before we can use the above circuits to do symbolic computation. In particular, we need a way to get information out of a circuit.

Definition 5. *Given a Hilbert space V and a finite alphabet Σ, an* **observable decomposition of V with values in Σ** *is a collection of mutually orthogonal, closed subspaces $\mathcal{O} = \{V_\sigma \mid \sigma \in \Sigma\}$ indexed by Σ which is complete in the sense that its sum is all of V, so $V = \bigoplus_{\sigma\in\Sigma} V_\sigma$. This decomposition allows us to define a probabilistic function (again overloading the notation) $\mathcal{O} : S^1(V) \to \Sigma$ by saying that $\mathcal{O}(v) = \sigma$ with probability $\|P_\sigma(v)\|^2$, where $S^1(V)$ is the unit sphere in V and $P_\sigma : V \twoheadrightarrow V_\sigma$ is the orthogonal projection.*

Such an $\mathcal{O}$ induces also a decomposition of $V^{\otimes n}$ and hence a similar probabilistic function $\mathcal{O} : S^1(V^{\otimes n}) \to \Sigma^n$.

Definition 6. *Fix a circuit $\mathcal{C}$ with structure group $\mathcal{G}$ on n wires having values in the Hilbert space V. Given a function $\epsilon : \Sigma \to S^1(V)$ (called the* **encoding function***) and an observable decomposition $\mathcal{O}$ of V, we define a probabilistic function*

$$\mathcal{C} : \Sigma^n \to \Sigma^n : (\sigma_1, \ldots, \sigma_n) \mapsto \mathcal{O}(\mathcal{C}(\epsilon(\sigma_1) \otimes \cdots \otimes \epsilon(\sigma_n)))$$

called the **symbolic** *(or* **cooked***)* **computation with $\mathcal{C}$.**

Example 5. For any base field $\mathbb{F}$, let $d \in \mathbb{N}$ and $V = \mathbb{F}^d$. We will use the physicists' notation for the standard orthonormal basis of V, being $\{|0\rangle, \ldots, |d-1\rangle\}$. Then the **standard d-ary encoding** is the encoding function on the alphabet $\Sigma = \mathbb{Z}_d$ defined by $\epsilon_d : a \mapsto |a\rangle$. In this situation, vectors in V are often called **qudits**. When $d = 2$, the encoding is termed **binary** and the vectors **qubits**.

Corresponding to the standard basis, there is an observable decomposition of V with values in Σ defined by the subspaces $\{\mathbb{F} \cdot |0\rangle, \ldots, \mathbb{F} \cdot |d-1\rangle\}$ which is called **measurement in the d-ary computational basis**, written $\mathcal{O}_{|d\rangle}$.

2.2 Notions of universality

There are a number of ways in which the word "universal" could be used in this computational model. One way is external to the computational processes we are using here, in that it asks if our cooked computations can yield some large known universe of computations, such as perhaps computing all Boolean functions. This will be investigated below in §3.2.

Another sense for universality is bases on the practical consideration that if we are to construct computational circuits in the real world, we want to have a limited number of specific gates which we can figure out how to construct, but which in combination will be able to do a large universe of useful work. Since in practice it is hard to anticipate what circuits we will eventually figure out do useful computation, we usually seek a small number of gates which in combination will create *any* gate from the structure group in question, operating on any number of wires.

There are in fact two versions of this notion of sufficiently powerful sets of gates which are useful.

Definition 7. *Let V be a vector space on which a system $\mathcal{G} = \{G_k, \rho_k\}$ of groups acts. A collection of sets of gates in specific dimensions $\mathcal{H} = \{H_k \mid k \in K\}$, where $K \subseteq \mathbb{N}$ is some index set and $H_k \subseteq G_k$ $\forall k \in K$, generates a set of circuits built just as in Def. 4 except that only group elements from one of the H_k are allowed, only operating on k wires, where $k \in K$.*

We say the collection $\mathcal{H}$ is **exactly universal for** *$\mathcal{G}$ if the program implemented by any circuit with structure group $\mathcal{G}$ is also implemented by a circuit generated as above with gates only from $\mathcal{H}$.*

$\mathcal{H}$ is instead said to be **approximately universal for** *$\mathcal{G}$ if the programs of circuits with structure group $\mathcal{G}$ on n wires can be approximated to any desired accuracy in the operator norm on $\mathrm{End}(V^{\otimes n})$ by the programs of circuits generated as above with gates only from $\mathcal{H}$.*

3 Relation to Classical Circuits

3.1 Deterministic classical

The first thing to notice about our computational model is that it includes all of classical, deterministic computation.

For this, use the standard binary encoding ϵ_2, the system $\mathcal{S}ym$ of symmetric groups acting on $\mathbb{F}^2$ ($\mathbb{F}$ being either $\mathbb{R}$ or $\mathbb{C}$) and finish with measurement $\mathcal{O}_{|2\rangle}$ in the binary computational basis. As has been well known for some time (see for example [3], [17], or [11] for early work, or many explanations of quantum computing, such as [2] or [15], for more modern descriptions) all functions computed by a traditional Boolean circuit can be realized as computations of reversible circuits with ancilla (extra wires).

Reversible gates are permutations of their input spaces, which amounts to the standard basis vectors in $(\mathbb{F}^2)^{\otimes d}$ for a d-bit gate, and such gates are all available in the full symmetric system $\mathcal{S}ym$. Further, observation in the computational basis simply decodes this association of the objects being permuted and the standard basis elements of $(\mathbb{F}^2)^{\otimes d}$.

3.2 Universality in deterministic classical

The classical results mentioned above give a kind of universality for the circuit based on permutations, *external* in the sense of §2.2: with ancilla and techniques such as Bennett's famous "trick", any of these external objects – Boolean functions – can be computed by our $\mathcal{S}ym$-structured circuits.

Note that it was traditional even before the connection with reversible computation to say that the NAND gate is universal in the context of Boolean circuits. This simply meant that the other basic gates of Boolean circuits could be built up out of NAND. Then within the realm of reversible computation often uses the Toffoli gate as its basic universal and reversible gate, in the same sense (now with ancilla).

Taking into account the crossing of wires implicit in traditional circuit models (as mentioned above in Note 1), we can rephrase this result as

Theorem 1. *The two gates*

$$SWAP = \begin{pmatrix} 1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1 \end{pmatrix} \qquad \text{and} \qquad TOF = \begin{pmatrix} 1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&1&0 \end{pmatrix}$$

generate a set of set of circuits acting on the tensor powers of $V = \mathbb{R}^2$ large enough to compute all Boolean functions, using ancilla and the Bennett trick.

Notice that this notion of universality is not the one described in §2.2 for small sets generating all of an ambient structure group. There are many obvious families which work in this particular case, however, particularly easily because the groups are finite. A favorite of mathematicians, for example, would be to take some traditional set of generators for each S_{2^k}, such as all the transpositions, or a large cycle and a single appropriate transposition. Which generating set is most convenient depends upon the physical realization one wants to attempt: nearest-neighbor interactions may be the easiest to realize, plus some single larger operation (such is in linear ion traps). Any proposed generating set can then be tested to see if it generates, in the sense of §2.2 the structure group $\mathcal{S}ym$ (or some subset necessary for a particular family of computations).

3.3 Nondeterministic classical

There is a tension between nondeterminism and reversibility which I have not seen spelled out explicitly elsewhere, but which is fairly clear using the computational approach of this paper. The issue is that a circuit with access to nondeterminism – which is usually imagined as a gate with no inputs but an output that is uniformly distributed on $\{0, 1\}$ – really cannot be reversed. That random bit came from nowhere, and even if we imagine the gate as having an input wire in addition to the output, we have no way of knowing what the input was simply by knowing the output.

In our model, this amounts to a wire on which the gate that produces a random bit is realized as the matrix (remember we are using $V = \mathbb{R}^2$ or $\mathbb{C}^2$)

$$RAND_{1/2} = \begin{pmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{pmatrix} \quad .$$

Notice that this $RAND_{1/2}$ is an equally weighted convex combination of the identity $\mathrm{Id}_2 = \left(\begin{smallmatrix} 1&0\\0&1 \end{smallmatrix}\right)$ and the gate $NOT = \left(\begin{smallmatrix} 0&1\\1&0 \end{smallmatrix}\right)$ and implements a tradition fair coin.

Biased coins can also be incorporated into this model. For $\alpha \in [0, 1]$, let

$$RAND_\alpha = \begin{pmatrix} \alpha & 1-\alpha \\ 1-\alpha & \alpha \end{pmatrix} = \alpha\,\mathrm{Id}_2 + (1-\alpha)\,NOT \quad .$$

If for some physical reason in our computational hardware, we only have access to a biased $RAND_\alpha$ but want the fair $RAND_{1/2}$, there are well known ways to

make a fair coin out of a biased one. These methods perhaps began with the *von Neumann extractor* [18] but have been improved over the years in several ways, see [12] and [16], for example. All of these methods require several flips of the biased coin (hopefully independently) and some (Boolean) computation, which may therefore require additional ancilla.

On the other hand, once we have a circuit which implements $RAND_{1/2}$, if we wish in some computation to use a $RAND_\alpha$ for some other $\alpha \in [0,1]$, we can proceed as follows. First, choose some $e \in \mathbb{N}$, from which we shall allow an error bound of size 2^{-e}. Run e copies of the gate $RAND_{1/2}$ in parallel and apply (the reversible version of) a Boolean circuit which compares the resulting bits to the first e binary digits of α to the right of the binary point and outputs a 1 if the random number is less than that portion of α, or 0 otherwise. This output bit, once ancilla are discarded, amounts to a $RAND_\alpha$.

Suppose we wish to add the single random-bit gate $RAND_\alpha$ on the jth wire at the beginning of a computation performed by a circuit over the structure group $\mathcal{S}ym$ with program $P = \rho_{d_m}(g_m)^{\wedge j_m} \cdot \cdots \cdot \rho_{d_1}(g_1)^{\wedge j_1}$ (in the sense of Def. 4). Since tensoring with other matrices commutes with taking linear combinations, the resulting program will be the linear combination

$$\alpha \, \mathrm{Id}_2^{\wedge j} \cdot P + (1-\alpha)\, NOT^{\wedge j} \cdot P \; = \alpha \, P + (1-\alpha)\, NOT^{\wedge j} \cdot P$$

Adding further nondeterministic gates to the circuit – additional randomness in the form of $RAND_\alpha$ gates – results in circuits with raw computations coming from convex combinations of permutation matrices with additional NOT's at arbitrary locations. The coefficients in these convex combinations resulting from using several fair coins or any number of the $RAND_\alpha$'s produced by the simple scheme described above will always be dyadic rationals.

Nevertheless, it makes sense (by continuity, if for no other reason) to allow all convex combinations of permutations in non-deterministic circuits. As mentioned before, the Birkhoff-von Neumann Theorem then tells us that this amounts to using gates in the semigroup $\mathbf{B}_n$. In summary:

Theorem 2. *The symbolic computations performed by circuits with structure semigroup $\mathcal{B}v\mathcal{N}$ [using ancilla] are exactly the computations performed by classical probabilistic Boolean circuits.*

This theorem points to a kind of answer to the old question of why classical probabilistic computation produces algorithms which seem faster than deterministic: probabilistic computations are basic on the structure [semi]group $\mathcal{B}v\mathcal{N}$ which maybe have an algebraic defect (no inverses) but is a very large superset of the permutation structure group $\mathcal{S}ym$ which underlies deterministic computation. Furthermore, the doubly stochastic matrices in $\mathbf{B}_n$ have a quality of creating a mixture, or probabilistic superposition, of many computational paths simultaneously. That this only leads to some improvement and not the the exponential speed-up of quantum computation seems to be due to the fact that we cannot cancel the unwanted computational paths in the same way as the famous quantum computational algorithms. In §6 below, a conjecture is formulated which makes this crucial distinction precise.

4 Relation to Quantum Circuits

If we want to generalize classical computation in a way that allows both probabilistic and reversible computation, we can change our structure group from the symmetric $\mathcal{S}$ to something which includes mixtures like the gate providing a random bit and yet remains a group. The way this happened in the history of computer science was by going to the unitary structure group $\mathcal{U}$ and full-fledged quantum computation. With this group, we have access to the Hadamard gate

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

which is a nice generalization of the random-bit gate $RAND_{1/2}$ above, with only a well placed negative entry to make it invertible (and different over-all scale).

The computation model in this paper is clearly designed primarily to be an extension of that standard version of quantum computation, so using the structure group $\mathcal{U}$ simply implements that standard. The one thing I do here which is not always done in the basic quantum set-up is that I do all measurement at the end. However, the *principle of deferred measurement* (to use the name coined, I believe, by [15]) says that we may always do this, at least if we are not interested in the raw computation of a circuit, its effect on qubits, but want only to work with the symbolic computation. Thus

Theorem 3. *The symbolic computations produced by circuits in the sense of this paper with structure group $\mathcal{U}$, binary encoding, and measurement $\mathcal{O}_{|2\rangle}$ in the computational basis are exactly the symbolic computations produced by the standard model of quantum computation.*

4.1 Universal quantum gates

The machinery set up in this paper allows a different approach to proving universality of sets of quantum gates: one must show how to build up exactly or approximately the elements of $U(2^d)$ out of the chosen set of gates by multiplication of tensor products of the smaller gates from the set. This can be applied to the specific choices of universal families from [1], [10], or [6], for example. I defer those calculations to another work, in order to pursue further generalizations of the quantum model here.

I note here only a couple of facts which are known about universal families of quantum gates: it is known that almost any 2-bit gate, together will the whole set of 1-qubit gates, is an exactly universal family for quantum computation; also several finite subsets of the 1-qubit gates, again together with a 2-qubit gate, are known to be approximately universal. See references mentioned immediately above, as well as [13] and many other references in [15].

5 Universes with Other Structure Groups

The promise of the computational model presented in this paper is that it generalizes both classical and quantum computation, and leaves open the possibilities to explore computation performed by circuits using other structure groups. Since the unitary group carries the basic physics of the quantum universe in which we live, and the permutation group carries the basic physics of idealized classical computation (à la billiard ball computers or ideal Turing machines in a Newtonian universe), one way to think of this new possibility is as examining computation in other universes, with their own structure groups.

5.1 An un-real but $\mathbb{R}$eal universe

Here is just one example. The first quantum algorithm to show exponential speed-up was the algorithm to solve Deutsch's Problem [8], which uses the linear structure of the tensor product and the particular action of the Hadamard transformation. But the Hadamard matrix shown above looks very much like another matrix from elementary linear algebra: the real, special orthogonal 2×2 matrix

$$R_{\pi/4} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$$

which implements rotation by $\pi/4$ counterclockwise in $\mathbb{R}^2$. So let us use the structure group $\mathcal{SO}$ with the vector space $V = \mathbb{R}^2$ to build an entirely $\mathbb{R}$eal circuit that solves this problem in much the same was as the structure group $\mathcal{U}$ solves it in our universe governed by complex quantum mechanics.

Following the model exposition in [15], we suppose that we have a function $f : \{0,1\} \to \{0,1\}$ which may or may not be constant. We imagine have an oracle U_f in our $\mathbb{R}$eal world which can compute the $\mathcal{SO}$ version of f, as $U_f : |x\rangle|y\rangle \mapsto |x\rangle|f(x) \oplus y\rangle$.

We start with the initial state $|01\rangle = |0\rangle \otimes |1\rangle$ and apply $R_{\pi/4} \otimes R_{\pi/4}$, yielding

$$\left(R_{\pi/4}\right)^{\otimes 2} (|01\rangle) = \frac{1}{2} \left(|0\rangle + |1\rangle\right) \left(-|0\rangle + |1\rangle\right)$$

which we submit to the oracle U_f, yielding

$$\frac{1}{2} \left[|0\rangle \left(-|f(0) \oplus 0\rangle + |f(0) \oplus 1\rangle\right) + |1\rangle \left(-|f(1) \oplus 0\rangle + |f(1) \oplus 1\rangle\right)\right]$$

which simplifies a bit as

$$(-1)^{f(0)} \frac{1}{2} \left(|0\rangle + (-1)^{f(0) \oplus f(1)} |1\rangle\right) \left(-|0\rangle + |1\rangle\right) \ .$$

Finally, we apply $R_{\pi/4}^{\wedge 1} = R_{\pi/4} \otimes \mathrm{Id}_2$ ["$R_{\pi/4}$ on the first qubit"] to get

$$|f(0) \oplus f(1) \oplus 1\rangle \frac{1}{\sqrt{2}} \left(-|0\rangle + |1\rangle\right)$$

so the first bit of the measurement $\mathcal{O}_{|2\rangle}$ tells us the if $f(0) \oplus f(1)$ is 0 or 1, hence if f is constant or not.

The above is merely a toy model, but its use of the $\mathbb{R}$eal analogue $R_{\pi/4}$ of the quantum Hadamard demonstrates that the full Deutsch-Jozsa algorithm [9], based as it is on repeated use of the Hadamard gate and its tensor powers, will go straight through in the $\mathbb{R}$eal world of $\mathcal{SO}$ circuits.

Pulling back a little from the science-fictional conceit above that physical implementation of this approach would involve a visit to another universe, we can imagine now solving these kinds of problems – with exponential speed-up over classical computation – by finding a system with $SO(2)$ symmetry (a round ring, or rotationally (in $\mathbb{R}^2$) symmetric object or field, or ...?) of which multiple copies can be coupled, in the manner of the tensor products underlying our circuit model. Such a system would implement the $\mathbb{R}$eal computation described above, so would permit all of the usual exponential speed-up from quantum computation, regardless of whether the hardware used quantum mechanics at all – the only necessary ingredient being the algebraic structure described above.

6 Future Directions, Including a Conjecture and a Proposal

The computational model of this paper allows a search for other structure groups which may

1. be associated with physically realizable systems, and
2. be used to perform calculations at significant speedup over classical algorithms

There are two directions which I think are particularly promising for this approach. One is to vary the structure group (which variation so far has not be much the subject of investigation) in order to determine which groups and representations admit quantum speedup. On that matter, it seems to me that useful information is extracted, out of the exponentially large tensor product spaces in which states of the many wires of our circuits lie, only when there can be destructive interference.

Conjecture 1. I conjecture that structure groups $\mathcal{G} = \{G_k, \rho_k\}$ can only exhibit exponential speedup over classical algorithms if there are matrix coefficients of with different signs.

The converse probably needs some additional hypotheses, however.

In this context, it is worth thinking about the hierarchy of computational power we have seen in this paper, and how delicate it is:

- Straight permutations give circuits over $\mathcal{S}ym$ which yield classical deterministic computation.
- Other finite groups presumably do not give any speed-up – all finite groups arise as subgroups of permutation groups.

- Taking convex combinations of permutations yields $\mathcal{B}v\mathcal{N}$ circuits, and probabilistic classical computation – which is intriguingly faster than deterministic. This structure semigroup is no longer finite (it is uncountable), but it is compact and non-Abelian, and its standard representation has only positive matrix coefficients.
- We have seen above an example (the $\mathbb{R}$eal world of circuits over $\mathcal{SO}$) with exponential speed-up over classical computation. The structure group here is an actual group: compact and Abelian, and has a representation on a real vector space, but also with negative matrix coefficients.
- The (complex) unitary structure group of quantum mechanics also has its famous algorithms with exponential speed-up over classical ones. $U(n)$ is compact and non-Abelian, with non-positive matrix coefficients in its standard representation. Yet **finite** [universal] **families of gates** can implement these fast algorithms.

That finite structure groups are classical (slow) and finite families of unitary gates are quantum (fast) is quite surprising, as is the example of quantum-like algorithms in a two-dimensional $\mathbb{R}$eal set-up.

Finally, having opened up the possibility of circuits with different structure groups, we can seek out much larger groups where there may be radically different behavior. I therefore propose that some of these much large groups, but perhaps groups which are symmetries of real, known, physical systems, be investigated. Of particular interest would be

1. non-compact, non-Abelian Lie groups – these usually have their interesting representations on infinite dimensional vector spaces, but such function spaces and group actions do occur in quantum mechanics; and
2. really large groups, such as ones which are not even locally compact (as sometimes arise in quantum field theory, among other areas), for example the group of isometries of a separable Hilbert space, or the group of diffeomorphisms of some manifold.

Acknowledgements. I would like to thank Normal Herzberg for a very important comment at a crucial point in this work.

References

1. Barenco, A.: A universal two-bit gate for quantum computation. Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences 449(1937), 679–683 (1995)
2. Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D.P., Margolus, N., Shor, P., Sleator, T., Smolin, J.A., Weinfurter, H.: Elementary gates for quantum computation. Physical Review A 52(5), 3457 (1995)
3. Bennett, C.H.: Logical reversibility of computation. IBM journal of Research and Development 17(6), 525–532 (1973)
4. Birkhoff, G.: Tres observaciones sobre el algebra lineal. Univ. Nac. Tucumán Rev. Ser. A 5, 147–151 (1946)

5. Blum, L., Cucker, F., Shub, M., Smale, S.: Complexity and real computation. Springer Verlag (1998)
6. Brylinski, J.L., Brylinski, R.: Universal quantum gates. Mathematics of Quantum Computation pp. 101–116 (2002)
7. Coecke, B.: New structures for physics, vol. 813. Springer (2010)
8. Deutsch, D.: Quantum theory, the church-turing principle and the universal quantum computer. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences 400(1818), 97–117 (1985)
9. Deutsch, D., Jozsa, R., Deutsch, D., Jozsa, R.: Rapid solution of problems by quantum computation. Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences 439(1907), 553–558 (1992)
10. DiVincenzo, D.P.: Two-bit gates are universal for quantum computation. Physical Review A 51(2), 1015 (1995)
11. Fredkin, E., Toffoli, T.: Conservative logic. International Journal of Theoretical Physics 21(3), 219–253 (1982)
12. Juels, A., Jakobsson, M., Shriver, E., Hillyer, B.K.: How to turn loaded dice into fair coins. Information Theory, IEEE Transactions on 46(3), 911–921 (2000)
13. Lloyd, S.: Almost any quantum logic gate is universal. Physical Review Letters 75(2), 346–349 (1995)
14. Minc, H.: Nonnegative matrices. Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, New York (1988)
15. Nielsen, M.A., Chuang, I.L.: Quantum computation and quantum information. Cambridge university press (2010)
16. Shaltiel, R.: Recent developments in explicit constructions of extractors. Current Trends in Theoretical Computer Science: Algorithms and complexity 1, 189 (2004)
17. Toffoli, T.: Reversible computing. Automata, Languages and Programming pp. 632–644 (1980)
18. Von Neumann, J.: Various techniques used in connection with random digits. Applied Math Series 12(36-38), 1 (1951)