Measurement-Based and Universal Blind Quantum Computation

Citation for published version:
Broadbent, A, Fitzsimons, J & Kashefi, E 2010, Measurement-Based and Universal Blind Quantum Computation. in A Aldini, M Bernardo, A Di Pierro & H Wiklicky (eds), Formal Methods for Quantitative Aspects of Programming Languages: 10th International School on Formal Methods for the Design of Computer, Communication and Software Systems, SFM 2010, Bertinoro, Italy, June 21-26, 2010, Advanced Lectures. Lecture Notes in Computer Science, vol. 6154, Springer Berlin Heidelberg, pp. 43-86. DOI: 10.1007/978-3-642-13678-8_2

Digital Object Identifier (DOI):
10.1007/978-3-642-13678-8_2

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Formal Methods for Quantitative Aspects of Programming Languages

Measurement-based and Universal Blind Quantum Computation

Anne Broadbent[1], Joseph Fitzsimons[1,2], Elham Kashefi [3] *

A Introduction

Traditionally, the main framework to explore quantum computation has been the circuit model [Deu89], based on unitary evolution. This is very useful for complexity analysis [BV97]. There are other models such as quantum Turing machines [Deu85] and quantum cellular automata [Wat95, vD96, DS96, SW04]. Although they are all proved to be equivalent from the point of view of expressive power, there is no agreement on what is the canonical model for exposing the key aspects of quantum computation.

Recently, distinctly different models have emerged, namely adiabatic and topological quantum computing. Both suggest different architectures, and fault tolerant schemes, and provide specific approaches to new applications and algorithms, and specific means to compare classical and quantum computation. Another family of models, collectively called measurement-based quantum computing (MBQC), has also received wide attention. MBQC is very different from the circuit model where measurement is done only at the end to extract classical output. In measurement-based quantum computing the main operation to manipulate information and control computation is measurement [GC99, RB01, RBB03, Nie03]. This is surprising because measurement creates indeterminacy, yet it is used to express deterministic computation defined by a unitary evolution.

More precisely, a computation consists of a phase in which a collection of qubits are set up in a standard entangled state. Then measurements are applied to individual qubits and the outcomes of the measurements may be used to determine further adaptive measurements. Finally – again depending on measurement outcomes – local adaptive unitary operators, called corrections, are applied to some qubits; this allows the elimination of the indeterminacy introduced by measurements. Conceptually MBQC highlights the role of entanglement and separates the quantum and classical aspects of computation; thus it clarifies, in particular, the interplay between classical control and the quantum evolution process.

The first structural feature of MBQC is a key factorisation property, namely that entanglement can be done first, and then local measurements, can be reduced to confluence properties of a simple algebraic rewriting system. This is captured by the Measurement Calculus [DKP07], one can think of it as an "assembly language" for MBQC with a notation for such classically correlated sequences of entanglements, measurements, and local corrections. Computations are organised in patterns, we use the word "pattern" rather than "program", because this corresponds to the commonly used terminology in the physics literature. Measurement Calculus consists of local equations over

[1] Institute for Quantum Computing, University of Waterloo, Waterloo, Ontario, Canada.

[2] Materials Department, University of Oxford, Oxford, United Kingdom.

[3] School of Informatics, University of Edinburgh, Edinburgh, Scotland, United Kingdom.

[*] Email: albroadb@iqc.ca, joe.fitzsimons@materials.ox.ac.uk, ekashefi@inf.ed.ac.uk

patterns that exploits some special algebraic properties of the entanglement, measurement and correction operators. More precisely, it uses the fact that 1-qubit measurements are closed under conjugation by Pauli operators and the entanglement command belongs to the normaliser of the Pauli group. This calculus is sound in that it preserves the interpretation of patterns. Most importantly, one can derive from it a simple algorithm (called *standardisation*) by which any general pattern can be put into a standard form where entanglement is done first, then measurements, then corrections.

The consequences of the existence of such a procedure are far-reaching. Since entangling comes first, one can prepare the entire entangled state needed during the computation right at the start: one never has to do "on the fly" entanglements. Furthermore, the rewriting of a pattern to standard form reveals parallelism in the pattern computation. In a general pattern, one is forced to compute sequentially and to strictly obey the command sequence, whereas, after standardisation, the dependency structure is relaxed, resulting in lower computational depth complexity [BK09]. The existence of a standard form for any pattern also has interesting corollaries beyond implementation and complexity matters, as it follows from it that patterns using no dependencies, or using only the restricted class of Pauli measurements, can only realise a unitary belonging to the Clifford group, and hence can be efficiently simulated by a classical computer [DKP07].

The second structural feature of MBQC is captured by the notation of *Flow* [DK06]. Although quantum measurements are inherently not deterministic, one can sometimes ensure the global determinism of the computation using suitable dependencies between measurements. Flow asserts that under a graph-theoretic condition on the entanglement underlying a given computation, it is possible to construct such dependencies. This is a significant progress in the direct understanding of the specifics of measurement-based information processing. Building on this criterion, and the well known stabiliser formalism, a full characterisation of determinism is obtained [BKMP07].

Having obtained the rigourous mathematical model underlying MBQC, we then present how this model suggests new techniques for designing quantum protocols. We present a protocol, called *Universal Blind Quantum Computation* (UBQC) which allows a client to have a server carry out a quantum computation for her such that the client's inputs, outputs and computation remain perfectly private, and where she does not require any quantum computational power or memory. The client only needs to be able to prepare single qubits randomly chosen from a finite set and send them to the server, who has the balance of the required quantum computational resources. UBQC protocol is the first universal scheme which detects a cheating server, as well as the first protocol which does not require any quantum computation whatsoever on the client's side. The novelty of UBQC protocol is in using the unique features of MBQC which allows one to clearly distinguish between the quantum and classical aspects of a quantum computation.

B Preliminaries

We give a brief summary of quantum mechanics and quantum computing. We develop some of the algebra, define some notations, and prove a couple of equations which are used in this chapter. The reader will find the expository book of Nielsen and Chuang [NC00] useful for quantum computation or the excellent book by Peres [Per95] for general background on quantum mechanics.

The vector spaces that arise in quantum mechanics are *Hilbert spaces* and are thus usually written $\mathfrak{H}$; that is they have an inner product usually written $\langle u, v\rangle$ where u and v are vectors. Following Dirac, it is customary to call elements of $\mathfrak{H}$ *kets* and write them in the form $|u\rangle$ or whatever symbol is appropriate inside the half-bracket. The dual vectors are called *bras* and are written $\langle v|$; the pairing thus can naturally be identified – conceptually and notationally – with the inner product.

A *hermitian operator* A is one such that $A = A^\dagger$ and a *unitary operator* U is one such that $U^{-1} = U^\dagger$. A *projection* P is a linear operator such that $P^2 = P$ and $P = P^\dagger$. A projection operator can be identified with a subspace, namely its range. The eigenvalues of a hermitian operator are always real. Suppose U is a unitary, and P a projection, then $UPU^\dagger$ is also a projection. The spectral theorem for hermitian operators states that if M is a hermitian operator, λ_i are its eigenvalues and P_i are projection operators onto the corresponding eigenspaces then one can write

$$M = \sum_i \lambda_i P_i.$$

If we have $|i\rangle$ as the normalised eigenvectors for the eigenvalues λ_i then we can write this in Dirac notation as:

$$M = \sum_i \lambda_i |i\rangle\langle i|.$$

Finally we need to combine Hilbert spaces. Given two Hilbert spaces $\mathfrak{H}$ with basis vectors $\{a_i | 1 \leq i \leq n\}$ and $\mathfrak{H}'$ with basis $\{b_j | 1 \leq j \leq m\}$ we define the *tensor product*, written $\mathfrak{H} \otimes \mathfrak{H}'$, as the vector space of dimension $n \cdot m$ with basis $a_i \otimes b_j$. We almost never write the symbol $\otimes$ between the vectors. In the Dirac notation this is always omitted and one writes, for example, $|uv\rangle$ instead of $|u\rangle \otimes |v\rangle$.

The important point is that there are vectors that cannot be written as the tensor product of vectors. This means that given a general element of $\mathfrak{H} \otimes \mathfrak{H}'$ one cannot produce elements of $\mathfrak{H}$ and $\mathfrak{H}'$; this is very different from the cartesian product of sets. This is the mathematical manifestation of entanglement.

A very important function on square matrices is the *trace*. The usual trace – i.e. the sum of the diagonal entries – is basis independent and is actually equal to the sum of the eigenvalues, counted with multiplicity. The trace of A is written $tr(A)$ and satisfies the cyclicity property $tr(AB) = tr(BA)$; applying this repeatedly one gets

$$tr(A_1 \ldots A_n) = tr(A_{\sigma(1)} \ldots A_{\sigma(n)})$$

where σ is a cyclic permutation. The explicit formula for the trace of $A : V \rightarrow V$ is $tr(A) = \sum_i \langle i|A|i\rangle$ where $|i\rangle$ is a basis for V. One often needs to compute a *partial* trace. Consider a linear map $L : V \otimes W \rightarrow V \otimes W$. Suppose that $|v_i\rangle$ is a basis for V and $|w_i\rangle$ is a basis for W then $|v_i w_j\rangle$ is a basis for $V \otimes W$. Now we can define the partial trace over V as

$$tr_V(A) : W \rightarrow W = \sum_i \langle v_i|A|v_i\rangle.$$

This corresponds to removing the V dependency; often we use the phrase "tracing out the V component."

We can now state the basic facts of quantum mechanics and will not discuss the experimental basis for this framework. The key aspects of quantum mechanics are:

- the states of a quantum system form a Hilbert space,

- when two quantum systems are combined, the state space of the composite system is obtained as the tensor product of the state spaces of the individual systems, and
- the evolution of a quantum system is given by a unitary operator, and
- the effect of a measurement is indeterminate.

The first says that one can form *superpositions* of the states. This is one of the most striking features of quantum mechanics. Thus states are not completely distinct as they are in classical systems. The inner product measures the extent to which states are distinct. The fact that systems are combined by tensor product says that there are states that of composite systems that cannot be decomposed into individual pieces. This is the phenomenon of entanglement or *non-locality*.

Measurement is what gives quantum mechanics its indeterminate character. The usual case, called projective measurements, is when the quantity being measured is described by a hermitian operator M. The possible outcomes are the *eigenvalues* of M. If M is an observable (hermitian operator) with eigenvalues λ_i and eigenvectors $|\phi_i\rangle$ and we have a generic state $|\psi\rangle = \sum_i c_i|\phi_i\rangle$ then the probabilities and expectation values of the measurement outcomes are given by:

- $Prob(\lambda_i||\psi\rangle) = |c_i|^2$
- $E[M||\psi\rangle] = \sum_i |c_i|^2\lambda_i = \sum_i c_i\bar{c_i}\langle\phi_i, M\phi_i\rangle = \langle\psi, M\psi\rangle$.

It is important to note that the effect of the measurement is that the projection operator P_i is applied when the result λ_i is observed. The operator M does not describe the effect of the measurement.

Quantum computation is carried out with *qubits* the quantum analogues of bits. Just as a bit has two possible values, a qubit is a two dimensional complex Hilbert space, in other words it is (isomorphic to) the two dimensional complex vector space $\mathbb{C}^2$. One works with a preferred basis, physically this corresponds to two distinguishable states, like "spin up" and "spin down". One writes $|0\rangle$, and $|1\rangle$ for its canonical basis, so that any vector ψ can be written as $\alpha|0\rangle + \beta|1\rangle$ with α, β in $\mathbb{C}$. Furthermore, $\mathbb{C}^2$ can be turned into a Hilbert space with the following inner product:

$$\langle\alpha|0\rangle + \beta|1\rangle, \alpha'|0\rangle + \beta'|1\rangle\rangle \quad := \quad \alpha^\star\alpha' + \beta^\star\beta'$$

where $\alpha^\star$ is the complex conjugate of α. One then obtains the norm of a vector as:

$$\|\psi\| \quad := \quad \langle\psi, \psi\rangle^{\frac{1}{2}} \quad = \quad (\alpha^\star\alpha + \beta^\star\beta)^{\frac{1}{2}}$$

Given V a finite set, one writes $\mathfrak{H}_V$ for the Hilbert space $\otimes_{u\in V}\mathbb{C}^2$; the notation means an n-fold tensor product of the $\mathbb{C}^2$ where n is the size of V. A vector in $\mathfrak{H}_V$ is said to be *decomposable* if it can be written $\otimes_{u\in V}\psi_u$ for some $\psi_u \in \mathbb{C}^2$. Such decomposable vectors will be written ϵ in the sequel. Decomposable vectors can be represented by a map from V to $\mathbb{C}^2$, and we will use both notations depending on which is more convenient. As we have noted before there are some vectors that are not decomposable.

As in the case of $\mathbb{C}^2$, there is a canonical basis for $\mathfrak{H}_V$, sometimes also called the *computational basis*, containing decomposable vectors ϵ such that for all $v \in V$, $\epsilon(v) = |0\rangle$ or $\epsilon(v) = |1\rangle$.

The inner product on $\mathfrak{H}_V$, according to the general definition given above, is defined on decomposable vectors as:

$$\langle\epsilon, \epsilon'\rangle \quad := \quad \prod_{v\in V}\langle\epsilon(v), \epsilon'(v)\rangle$$

Note that all vectors in the computational basis are orthogonal and of norm 1. The vectors of norm 1 are usually called *unit vectors*; we always assume that states are described by unit vectors as noted before.

Here are some common states that arise in quantum computation:

$$|0\rangle = |\uparrow\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = |\downarrow\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, |+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, |-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

It is easy to see that a linear operator is unitary if it preserves the inner product and hence the norm. Thus unitaries can be viewed as maps from quantum states to quantum states.

Some particularly useful unitaries are the *Pauli operators* given by the following matrices in the canonical basis of $\mathbb{C}^2$:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

We note that all these operators are involutive, self-adjoint, and therefore unitaries. All these matrices have determinant $= -1$. Some basic algebra of these matrices are given below. First they all square to the identity.

$$X^2 = Y^2 = Z^2 = I.$$

The Pauli operators do not commute and we have the following relations:

$$\begin{array}{llll} XY = iZ & YX = -iZ & [X,Y] = 2iZ & \{X,Y\} = 0 \\ ZX = iY & XZ = -iY & [Z,X] = 2iY & \{Z,X\} = 0 \\ YZ = iX & ZY = -iX & [Y,Z] = 2iX & \{Y,Z\} = 0 \end{array}$$

Definition 1 *Define the* Pauli group, $\mathbf{P}_n$*, as the group consisting of tensor products of I, X, Y, and Z on n qubits, with an overall phase of* ± 1 *or* $\pm i$*.*

A very important related group is called the Clifford group.

Definition 2 *The* Clifford group, $\mathbf{C}_n$*, is the group of unitary operators that leave the Pauli group invariant under conjugation, i.e. it is the normaliser of the Pauli group viewed as a subgroup of the unitary group.*

The Clifford group on n qubits can be generated by the Hadamard transform, the controlled-X ($CNOT$) or controlled-Z ($\wedge Z$), and the single-qubit phase rotation:

$$H = \tfrac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \wedge Z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \quad P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

The importance of the Clifford group for quantum computation is that a computation consisting of only Clifford operations on the computational basis followed by final Pauli measurements can be efficiently simulated by a classical computer, this is the Gottesman-Knill theorem [Got97, NC00].

In order to capture partial information about quantum systems one uses *density matrices.* Before we describe density matrices we review some linear algebra in the bra-ket notation. Given a ket $|\psi\rangle$ the notation $|\psi\rangle\langle\psi|$ denotes the projection operator onto the one dimensional subspace spanned by $|\psi\rangle$. If $|\psi_i\rangle$ is an orthonormal basis for $\mathfrak{H}$ the identity matrix is written $\sum_i |\psi_i\rangle\langle\psi_i|$. If Q is a linear operator with eigenvalues q_i and eigenvectors $|q_i\rangle$, which form an orthonormal basis for $\mathfrak{H}$, we can represent Q as $\sum_i q_i|q_i\rangle\langle q_i|$. A state $|\psi\rangle$ in $\mathfrak{H}$ is called a *pure* state. If a and b are distinct eigenvalues of some observable A with corresponding eigenvectors $|a\rangle$ and $|b\rangle$ it is perfectly possible to prepare a state of the form $\frac{1}{\sqrt{2}}(|a\rangle + |b\rangle)$. A measurement of A on such a state will yield either a or b each with probability $\frac{1}{2}$. However, it is also possible that a mixture is prepared. That is to say instead of a quantum superposition a classical stochastic mixture is prepared. In order to describe these we will use density matrices. For a system in a pure state $|\psi\rangle$, the density matrix is just the projection operator $|\psi\rangle\langle\psi|$.

What if the state is not known completely? Suppose that we only know that a system is one of several possible states $|\psi_1\rangle, \ldots, |\psi_k\rangle$ with probabilities $p_1, \ldots, p_k$ respectively. We define the density matrix for such a state to be

$$\rho = \sum_{i=1}^{k} p_i|\psi_i\rangle\langle\psi_i|.$$

The same formulas for the probability of observing a value q_i , i.e. $Tr(P_i\rho)$ and for the expectation value of Q, i.e. $Tr(Q\rho)$ apply. One can check directly that a density matrix has the following two properties.

Proposition 3 *An operator ρ on $\mathfrak{H}$ is a* density matrix *if and only if*

- *ρ has trace 1 and*
- *ρ is a positive operator, which means that it has only positive eigenvalues or, equivalently, that for any $x \in \mathfrak{H}$ we have $\langle x, \rho x\rangle \geq 0$.*

Furthermore, if ρ is a density operator, $Tr(\rho^2) \leq 1$ with equality if and only if ρ is a pure state (i.e. a projection operator).

The axioms of quantum mechanics are easily stated in the language of density matrices. For example, if evolution from time t_1 to time t_2 is described by the unitary transformation U and ρ is the density matrix for time t_1, then the evolved density matrix ρ' for time t_2 is given by the formula $\rho' = U\rho U^\dagger$. Similarly, one can describe measurements represented by projective operators in terms of density matrices [NC00, Pre98]. Thus if a projector P acts on a state $|\psi\rangle$ then the result is $P|\psi\rangle$; the resulting transformation of density matrices is $|\psi\rangle\langle\mapsto |P|\psi\rangle\langle P|$. For a general density matrix ρ we have $\rho \mapsto P\rho P$, note that since P is self-adjoint we do not have to write $P^\dagger$.

What are the legitimate "physical" transformations on density matrices? The legitimate transformations obviously take density matrices to density matrices. They have to be *positive maps* considered as maps between the appropriate ordered vector spaces. The appropriate ordered vector spaces are the vector spaces of linear operators on $\mathfrak{H}$ the Hilbert space of pure states. Unfortunately the tensor product of two positive maps is not positive in general. The remedy is to require the appropriate condition by fiat.

Definition 4 *A* completely positive map *K is a positive map such that for every identity map $I_n : \mathbb{C}^n \to \mathbb{C}^n$ the tensor product $K \otimes I_n$ is positive.*

It is not hard to show that the tensor of completely positive maps is always a completely positive map. The important result in this regard is the Kraus representation theorem [Cho75].

Theorem 1 (Kraus) *The general form for a completely positive map $\mathcal{E} : \mathcal{B}(\mathfrak{H}_1) \to \mathcal{B}(\mathfrak{H}_2)$ is*

$$\mathcal{E}(\rho) = \sum_m A_m \rho A_m^\dagger$$

where the $A_m : \mathfrak{H}_1 \to \mathfrak{H}_2$.

Here $\mathcal{B}(\mathfrak{H})$ is the Banach space of bounded linear operators on $\mathfrak{H}$. If, in addition, we require that the trace of $\mathcal{E}(\rho) \leq 1$ then the A_m will satisfy

$$\sum_m A_m^\dagger A_m \leq I.$$

The following term is common in the quantum computation literature.

Definition 5 *A* superoperator *T is a linear map from $\mathfrak{B}_V$ to $\mathfrak{B}_U$ that is completely positive and trace preserving.*

C MBQC - Syntax

We first develop a notation for 1-qubit measurement-based computations. The basic commands one can use in a pattern are:

- 1-qubit auxiliary preparation N_i
- 2-qubit entanglement operators E_{ij}
- 1-qubit measurements M_i^α
- and 1-qubit Pauli operators corrections X_i and Z_i

The indices i, j represent the qubits on which each of these operations apply, and α is a parameter in $[0, 2\pi]$. Expressions involving angles are always evaluated modulo 2π. These types of command will be referred to as N, E, M and C. Sequences of such commands, together with two distinguished – possibly overlapping – sets of qubits corresponding to inputs and outputs, will be called *measurement patterns*, or simply patterns. These patterns can be combined by composition and tensor product.

Importantly, corrections and measurements are allowed to depend on previous measurement outcomes. It is known that patterns without these classical dependencies can only realise unitaries that are in the Clifford group [DKP07]. Thus, dependencies are crucial if one wants to define a universal computing model; that is to say, a model where all unitaries over $\otimes^n \mathbb{C}^2$ can be realised. It is also crucial to develop a notation that will handle these dependencies. This is what we do now.

Preparation N_i prepares qubit i in state $|+\rangle_i$. The entanglement commands are defined as $E_{ij} := \wedge Z_{ij}$ (controlled-Z), while the correction commands are the Pauli operators X_i and Z_i.

Measurement M_i^α is defined by orthogonal projections on

$$\begin{array}{rcl} |+_\alpha\rangle & := & \frac{1}{\sqrt{2}}(|0\rangle + e^{i\alpha}|1\rangle) \\ |-_\alpha\rangle & := & \frac{1}{\sqrt{2}}(|0\rangle - e^{i\alpha}|1\rangle) \end{array}$$

followed by a trace-out operator. The parameter $\alpha \in [0, 2\pi]$ is called the *angle* of the measurement. For $\alpha = 0$, $\alpha = \frac{\pi}{2}$, one obtains the X and Y Pauli measurements. Operationally, measurements will be understood as destructive measurements, consuming their qubit. The *outcome* of a measurement done at qubit i will be denoted by $s_i \in \mathbb{Z}_2$. Since one only deals here with patterns where qubits are measured at most once (see condition (D1) below), this is unambiguous. We take the specific convention that $s_i = 0$ if under the corresponding measurement the state collapses to $|{+_\alpha}\rangle$, and $s_i = 1$ if to $|{-_\alpha}\rangle$.

Outcomes can be summed together resulting in expressions of the form $s = \sum_{i \in I} s_i$ which we call *signals*, and where the summation is understood as being done in $\mathbb{Z}_2$. We define the *domain* of a signal as the set of qubits on which it depends.

As we have said before, both corrections and measurements may depend on signals. Dependent corrections will be written X_i^s and Z_i^s and dependent measurements will be written ${}^t[M_i^\alpha]^s$, where $s, t \in \mathbb{Z}_2$ and $\alpha \in [0, 2\pi]$. The meaning of dependencies for corrections is straightforward: $X_i^0 = Z_i^0 = I$, no correction is applied, while $X_i^1 = X_i$ and $Z_i^1 = Z_i$. In the case of dependent measurements, the measurement angle will depend on s, t and α as follows:

$$
{}^t[M_i^\alpha]^s \quad := \quad M_i^{(-1)^s \alpha + t\pi} \tag{1}
$$

so that, depending on the parities of s and t, one may have to modify the α to one of $-\alpha$, $\alpha + \pi$ and $-\alpha + \pi$. These modifications correspond to conjugations of measurements under X and Z:

$$
\begin{aligned}
X_i M_i^\alpha X_i &= M_i^{-\alpha} &(2)\\
Z_i M_i^\alpha Z_i &= M_i^{\alpha+\pi} &(3)
\end{aligned}
$$

accordingly, we will refer to them as the X and Z-actions. Note that these two actions commute, since $-\alpha + \pi = -\alpha - \pi$ up to 2π, and hence the order in which one applies them does not matter.

As we will see later, relations (2) and (3) are key to the propagation of dependent corrections, and to obtaining patterns in the standard entanglement, measurement and correction form. Since the measurements considered here are destructive, the above equations actually simplify to

$$
\begin{aligned}
M_i^\alpha X_i &= M_i^{-\alpha} &(4)\\
M_i^\alpha Z_i &= M_i^{\alpha-\pi} &(5)
\end{aligned}
$$

Another point worth noticing is that the domain of the signals of a dependent command, be it a measurement or a correction, represents the set of measurements which one has to do before one can determine the actual value of the command.

We have completed our catalog of basic commands, including dependent ones, and we turn now to the definition of measurement patterns. For convenient reference, the language syntax is summarised in Figure 1. We proceed now with the formal definition of a measurement pattern.

Definition 6 *Patterns consists of three finite sets V, I, O, together with two injective maps $\iota : I \to V$ and $o : O \to V$ and a finite sequence of commands $A_n \ldots A_1$, read from right to left, applying to qubits in V in that order, i.e. A_1 first and A_n last, such that:*

(D0) *no command depends on an outcome not yet measured;*

(D1) *no command acts on a qubit already measured;*

$$
\begin{array}{rcll}
S & := & 0,\, 1,\, s_i,\, S+S & \text{Signals}\\
A & := & N_i & \text{Preparations}\\
 & & E_{ij} & \text{Entanglements}\\
 & & {}^t[M_i^\alpha]^s & \text{Measurements}\\
 & & X_i^s,\, Z_i^s & \text{Corrections}
\end{array}
$$

Figure 1: 1-qubit based measurement language syntax

(D2) *no command acts on a qubit not yet prepared, unless it is an input qubit;*

(D3) *a qubit i is measured if and only if i is not an output.*

The set V is called the pattern *computation space*, and we write $\mathfrak{H}_V$ for the associated quantum state space $\otimes_{i\in V}\mathbb{C}^2$. To ease notation, we will omit the maps ι and o, and write simply I, O instead of $\iota(I)$ and $o(O)$. Note, however, that these maps are useful to define classical manipulations of the quantum states, such as permutations of the qubits. The sets I, O are called respectively the pattern *inputs* and *outputs*, and we write $\mathfrak{H}_I$, and $\mathfrak{H}_O$ for the associated quantum state spaces. The sequence $A_n \dots A_1$ is called the pattern *command sequence*, while the triple (V, I, O) is called the pattern *type*.

To run a pattern, one prepares the input qubits in some input state $\psi \in \mathfrak{H}_I$, while the non-input qubits are all set to the $|+\rangle$ state, then the commands are executed in sequence, and finally the result of the pattern computation is read back from outputs as some $\phi \in \mathfrak{H}_O$. Clearly, for this procedure to succeed, we had to impose the (D0), (D1), (D2) and (D3) conditions. Indeed if (D0) fails, then at some point of the computation, one will want to execute a command which depends on outcomes that are not known yet. Likewise, if (D1) fails, one will try to apply a command on a qubit that has been consumed by a measurement (recall that we use destructive measurements). Similarly, if (D2) fails, one will try to apply a command on a non-existent qubit. Condition (D3) is there to make sure that the final state belongs to the output space $\mathfrak{H}_O$, *i.e.*, that all non-output qubits, and only non-output qubits, will have been consumed by a measurement when the computation ends.

We write (D) for the conjunction of our definiteness conditions (D0), (D1), (D2) and (D3). Whether a given pattern satisfies (D) or not is statically verifiable on the pattern command sequence. We could have imposed a simple type system to enforce these constraints but, in the interests of notational simplicity, we chose not to do so.

Here is a concrete example:

$$\mathcal{H} \;:=\; (\{1,2\},\{1\},\{2\},X_2^{s_1}M_1^0E_{12}N_2)$$

with computation space $\{1,2\}$, inputs $\{1\}$, and outputs $\{2\}$. To run $\mathcal{H}$, one first prepares the first qubit in some input state ψ, and the second qubit in state $|+\rangle$, then these are entangled to obtain $\wedge Z_{12}(\psi_1 \otimes |+\rangle_2)$. Once this is done, the first qubit is measured in the $|+\rangle$, $|-\rangle$ basis. Finally an X correction is applied on the output qubit, if the measurement outcome was $s_1 = 1$. We will do this calculation in detail later, and prove that this pattern implements the Hadamard operator H.

In general, a given pattern may use auxiliary qubits that are neither input nor output qubits. Usually one tries to use as few such qubits as possible, since these contribute to the *space complexity* of the computation.

A last thing to note is that one does not require inputs and outputs to be disjoint subsets of V. This, seemingly innocuous, additional flexibility is actually quite useful to give parsimonious implementations of unitaries [DKP05].

Next we described how one can combine patterns in order to obtain bigger ones. The first way to combine patterns is by composing them. Two patterns $\mathcal{P}_1$ and $\mathcal{P}_2$ may be composed if $V_1 \cap V_2 = O_1 = I_2$. Provided that $\mathcal{P}_1$ has as many outputs as $\mathcal{P}_2$ has inputs, by renaming the pattern qubits, one can always make them composable.

Definition 7 *The composite pattern $\mathcal{P}_2\mathcal{P}_1$ is defined as:*
— $V := V_1 \cup V_2$*,* $I = I_1$*,* $O = O_2$*,*
— *commands are concatenated.*

The other way of combining patterns is to tensor them. Two patterns $\mathcal{P}_1$ and $\mathcal{P}_2$ may be tensored if $V_1 \cap V_2 = \varnothing$. Again one can always meet this condition by renaming qubits in such a way that these sets are made disjoint.

Definition 8 *The tensor pattern $\mathcal{P}_1 \otimes \mathcal{P}_2$ is defined as:*
— $V = V_1 \cup V_2$*,* $I = I_1 \cup I_2$*, and* $O = O_1 \cup O_2$*,*
— *commands are concatenated.*

In contrast to the composition case, all the unions involved here are disjoint. Therefore commands from distinct patterns freely commute, since they apply to disjoint qubits, and when we say that commands have to be concatenated, this is only for definiteness. It is routine to verify that the definiteness conditions (D) are preserved under composition and tensor product.

Before turning to this matter, we need a clean definition of what it means for a pattern to implement or to realise a unitary operator, together with a proof that the way one can combine patterns is reflected in their interpretations.

D MBQC - Semantics

In this section we give a formal operational semantics for the pattern language as a probabilistic labeled transition system. We define deterministic patterns and thereafter concentrate on them. We show that deterministic patterns compose. We give a denotational semantics of deterministic patterns; from the construction it will be clear that these two semantics are equivalent.

Besides quantum states, which are non-zero vectors in some Hilbert space $\mathfrak{H}_V$, one needs a classical state recording the outcomes of the successive measurements one does in a pattern. If we let V stand for the finite set of qubits that are still active (i.e. not yet measured) and W stands for the set of qubits that have been measured (i.e. they are now just classical bits recording the measurement outcomes), it is natural to define the computation state space as:

$$\mathcal{S} \quad := \quad \Sigma_{V,W} \mathfrak{H}_V \times \mathbb{Z}_2^W .$$

In other words the computation states form a V, W-indexed family of pairs q, Γ, where q is a quantum state from $\mathfrak{H}_V$ and Γ is a map from some W to the outcome space $\mathbb{Z}_2$. We call this classical component Γ an *outcome map*, and denote by $\varnothing$ the empty outcome map in $\mathbb{Z}_2^{\varnothing}$. We will treat these states as pairs unless it becomes important to show how V and W are altered during a computation, as happens during a measurement.

Operational semantics

We need some preliminary notation. For any signal s and classical state $\Gamma \in \mathbb{Z}_2^W$, such that the domain of s is included in W, we take s_Γ to be the value of s given by the outcome map Γ. That is to say, if $s = \sum_I s_i$, then $s_\Gamma := \sum_I \Gamma(i)$ where the sum is taken in $\mathbb{Z}_2$. Also if $\Gamma \in \mathbb{Z}_2^W$, and $x \in \mathbb{Z}_2$, we define:

$$\Gamma[x/i](i) = x,\ \Gamma[x/i](j) = \Gamma(j) \text{ for } j \neq i$$

which is a map in $\mathbb{Z}_2^{W \cup \{i\}}$.

We may now view each of our commands as acting on the state space $\mathcal{S}$; we have suppressed V and W in the first 4 commands:

$$\begin{array}{lcl}
q, \Gamma & \xrightarrow{N_i} & q \otimes |+\rangle_i, \Gamma \\
q, \Gamma & \xrightarrow{E_{ij}} & \wedge Z_{ij} q, \Gamma \\
q, \Gamma & \xrightarrow{X_i^s} & X_i^{s_\Gamma} q, \Gamma \\
q, \Gamma & \xrightarrow{Z_i^s} & Z_i^{s_\Gamma} q, \Gamma \\
V \cup \{i\}, W, q, \Gamma & \xrightarrow{{}_t[M_i^\alpha]^s} & V, W \cup \{i\}, \langle +_{\alpha_\Gamma}|_i q, \Gamma[0/i] \\
V \cup \{i\}, W, q, \Gamma & \xrightarrow{{}_t[M_i^\alpha]^s} & V, W \cup \{i\}, \langle -_{\alpha_\Gamma}|_i q, \Gamma[1/i]
\end{array}$$

where $\alpha_\Gamma = (-1)^{s_\Gamma} \alpha + t_\Gamma \pi$ following equation (1). Note how the measurement moves an index from V to W; a qubit once measured cannot be neasured again. Suppose $q \in \mathfrak{H}_V$, for the above relations to be defined, one needs the indices i, j on which the various command apply to be in V. One also needs Γ to contain the domains of s and t, so that s_Γ and t_Γ are well-defined. This will always be the case during the run of a pattern because of condition (D).

All commands except measurements are deterministic and only modify the quantum part of the state. The measurement actions on $\mathcal{S}$ are not deterministic, so that these are actually binary relations on $\mathcal{S}$, and modify both the quantum and classical parts of the state. The usual convention has it that when one does a measurement the resulting state is *renormalised* and the probabilities are associated with the transition. We do not adhere to this convention here, instead we leave the states unnormalised. The reason for this choice of convention is that this way, the probability of reaching a given state can be read off its norm, and the overall treatment is simpler. As we will show later, all the patterns implementing unitary operators will have the same probability for all the branches and hence we will not need to carry these probabilities explicitly.

Denotational semantics

Let $\mathcal{P}$ be a pattern with computation space V, inputs I, outputs O and command sequence $A_n \ldots A_1$. To execute a pattern, one starts with some input state q in $\mathfrak{H}_I$, together with the empty outcome map $\varnothing$. The input state q is then tensored with as many $|+\rangle$s as there are non-inputs in V (the N commands), so as to obtain a state in the full space $\mathfrak{H}_V$. Then E, M and C commands in $\mathcal{P}$ are applied in sequence from right to left. We can summarise the situation as follows:

$$\begin{tikzcd}
\mathfrak{H}_I \arrow[rr, dotted] \arrow[d] & & \mathfrak{H}_O \\
\mathfrak{H}_I \times \mathbb{Z}_2^{\varnothing} \arrow[r, "prep"] & \mathfrak{H}_V \times \mathbb{Z}_2^{\varnothing} \arrow[r, "A_1 \ldots A_n"] & \mathfrak{H}_O \times \mathbb{Z}_2^{V \smallsetminus O} \arrow[u]
\end{tikzcd}$$

If m is the number of measurements, which is also the number of non outputs, then the run may follow 2^m different branches. Each branch is associated with a unique binary string $\mathbf{s}$ of length m, representing the classical outcomes of the measurements along that branch, and a unique *branch map* $A_\mathbf{s}$ representing the linear transformation from $\mathfrak{H}_I$ to $\mathfrak{H}_O$ along that branch. This map is obtained from the operational semantics via the sequence (q_i, Γ_i) with $1 \leq i \leq n+1$, such that:

$$\begin{array}{l} q_1, \Gamma_1 = q \otimes |+ \ldots +\rangle, \varnothing \\ q_{n+1} = q' \neq 0 \\ \text{and for all } i \leq n : q_i, \Gamma_i \xrightarrow{A_i} q_{i+1}, \Gamma_{i+1}. \end{array}$$

Definition 9 *A pattern $\mathcal{P}$ realises a map on density matrices ρ given by $\rho \mapsto \sum_\mathbf{s} A_\mathbf{s}^\dagger(\rho) A_\mathbf{s}$. We write $[\![\mathcal{P}]\!]$ for the map realised by $\mathcal{P}$.*

Proposition 10 *Each pattern realises a completely positive trace preserving map.*

Proof. Later on we will show that every pattern can be put in a semantically equivalent form where all the preparations and entanglements appear first, followed by a sequence of measurements and finally local Pauli corrections. Hence branch maps decompose as $A_\mathbf{s} = C_\mathbf{s} \Pi_\mathbf{s} U$, where $C_\mathbf{s}$ is a unitary map over $\mathfrak{H}_O$ collecting all corrections on outputs, $\Pi_\mathbf{s}$ is a projection from $\mathfrak{H}_V$ to $\mathfrak{H}_O$ representing the particular measurements performed along the branch, and U is a unitary embedding from $\mathfrak{H}_I$ to $\mathfrak{H}_V$ collecting the branch preparations, and entanglements. Note that U is the same on all branches. Therefore,

$$\begin{array}{rcl} \sum_\mathbf{s} A_\mathbf{s}^\dagger A_\mathbf{s} & = & \sum_\mathbf{s} U^\dagger \Pi_\mathbf{s}^\dagger C_\mathbf{s}^\dagger C_\mathbf{s} \Pi_\mathbf{s} U \\ & = & \sum_\mathbf{s} U^\dagger \Pi_\mathbf{s}^\dagger \Pi_\mathbf{s} U \\ & = & U^\dagger (\sum_\mathbf{s} \Pi_\mathbf{s}) U \\ & = & U^\dagger U = I \end{array}$$

where we have used the fact that $C_\mathbf{s}$ is unitary, $\Pi_\mathbf{s}$ is a projection and U is independent of the branches and is also unitary. Therefore the map $T(\rho) := \sum_\mathbf{s} A_\mathbf{s}(\rho) A_\mathbf{s}^\dagger$ is a trace-preserving completely-positive map (cptp-map), explicitly given as a Kraus decomposition. □

Hence the denotational semantics of a pattern is a cptp-map. In our denotational semantics we view the pattern as defining a map from the input qubits to the output qubits. We do not explicitly represent the result of measuring the final qubits; these may be of interest in some cases. Techniques for dealing with classical output explicitly are given by Selinger [Sel04] and Unruh [Unr05]. With our definitions in place, we will show that the denotational semantics is compositional.

Theorem 2 *For two patterns $\mathcal{P}_1$ and $\mathcal{P}_2$ we have $[\![\mathcal{P}_1\mathcal{P}_2]\!] = [\![\mathcal{P}_2]\!][\![\mathcal{P}_1]\!]$ and $[\![\mathcal{P}_1 \otimes \mathcal{P}_2]\!] = [\![\mathcal{P}_2]\!] \otimes [\![\mathcal{P}_1]\!]$.*

Proof. Recall that two patterns $\mathcal{P}_1$, $\mathcal{P}_2$ may be combined by composition provided $\mathcal{P}_1$ has as many outputs as $\mathcal{P}_2$ has inputs. Suppose this is the case, and suppose further that $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively realise some cptp-maps T_1 and T_2. We need to show that the composite pattern $\mathcal{P}_2\mathcal{P}_1$ realises T_2T_1.

Indeed, the two diagrams representing branches in $\mathcal{P}_1$ and $\mathcal{P}_2$:

$$\begin{tikzcd} \mathfrak{H}_{I_1} \arrow[rr, dotted] \arrow[d] & & \mathfrak{H}_{O_1} \\ \mathfrak{H}_{I_1} \times \mathbb{Z}_2^{\varnothing} \arrow[r, "p_1"] & \mathfrak{H}_{V_1} \times \mathbb{Z}_2^{\varnothing} \arrow[r] & \mathfrak{H}_{O_1} \times \mathbb{Z}_2^{V_1 \smallsetminus O_1} \arrow[u] \end{tikzcd} \qquad \begin{tikzcd} \mathfrak{H}_{I_2} \arrow[rr, dotted] \arrow[d] & & \mathfrak{H}_{O_2} \\ \mathfrak{H}_{I_2} \times \mathbb{Z}_2^{\varnothing} \arrow[r, "p_2"] & \mathfrak{H}_{V_2} \times \mathbb{Z}_2^{\varnothing} \arrow[r] & \mathfrak{H}_{O_2} \times \mathbb{Z}_2^{V_2 \smallsetminus O_2} \arrow[u] \end{tikzcd}$$

can be pasted together, since $O_1 = I_2$, and $\mathfrak{H}_{O_1} = \mathfrak{H}_{I_2}$. But then, it is enough to notice 1) that preparation steps p_2 in $\mathcal{P}_2$ commute with all actions in $\mathcal{P}_1$ since they apply on disjoint sets of qubits, and 2) that no action taken in $\mathcal{P}_2$ depends on the measurements outcomes in $\mathcal{P}_1$. It follows that the pasted diagram describes the same branches as does the one associated to the composite $\mathcal{P}_2\mathcal{P}_1$.

A similar argument applies to the case of a tensor combination, and one has that $\mathcal{P}_2 \otimes \mathcal{P}_1$ realises $T_2 \otimes T_1$. □

E MBQC - Universality

In this section we first introduce a simple parameterised family $J(\alpha)$ that generates all unitaries over $\mathbb{C}^2$. By adding the unitary operator controlled-Z ($\wedge Z$) defined over $\mathbb{C}^2 \otimes \mathbb{C}^2$, one then obtains a set of generators for all unitary maps over $\otimes^n \mathbb{C}^2$. Both $J(\alpha)$ and $\wedge Z$, have simple realisations in MBQC, using only two qubits. As a consequence, one obtains an implementation of the controlled-U ($\wedge U$) family of unitaries, using only 14 qubits. Combining these as building blocks, any general unitary can be obtained by using relatively few auxiliary qubits [DKP05]. Furthermore, this building blocks have an interesting property, namely that their underlying entanglement graphs have no odd-length cycles, and such states have been shown to be robust against decoherence [DAB03].

Consider the following one-parameter family $J(\alpha)$:

$$J(\alpha) \quad := \quad \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & e^{i\alpha} \\ 1 & -e^{i\alpha} \end{pmatrix},$$

we can see already that the Pauli spin matrices, phase and Hadamard operators can be described using only $J(\alpha)$:

$$\begin{array}{ll} X = J(\pi)J(0) & P(\alpha) = J(0)J(\alpha) \\ Z = J(0)J(\pi) & H = J(0) \end{array}$$

We will also use the following equations:

$$\begin{array}{lcl} J(0)^2 & = & I \\ J(\alpha)J(0)J(\beta) & = & J(\alpha+\beta) \\ J(\alpha)J(\pi)J(\beta) & = & e^{i\alpha} Z\, J(\beta-\alpha) \end{array}$$

The second and third equations are referred to as the *additivity* and *subtractivity* relations. Additivity gives another useful pair of equations:

$$XJ(\alpha) = J(\alpha+\pi) = J(\alpha)Z \tag{6}$$

Any unitary operator U on $\mathbb{C}^2$ can be written:

$$U = e^{i\alpha} J(0)J(\beta)J(\gamma)J(\delta)$$

for some α, β, γ and δ in $\mathbb{R}$. We will refer to this as a *J-decomposition* of U. To prove this note that all three Pauli rotations are expressible in terms of $J(\alpha)$:

$$R_x(\alpha) \quad = \quad e^{-i\frac{\alpha}{2}} J(\alpha)J(0) \tag{7}$$

$$R_y(\alpha) \quad = \quad e^{-i\frac{\alpha}{2}} J(0)J(\frac{\pi}{2})J(\alpha)J(-\frac{\pi}{2}) \tag{8}$$

$$R_z(\alpha) \quad = \quad e^{-i\frac{\alpha}{2}} J(0)J(\alpha) \tag{9}$$

From the Z–X decomposition, we know that every 1-qubit unitary operator U can be written as:

$$U \;=\; e^{i\alpha} R_z(\beta) R_x(\gamma) R_z(\delta)$$

and using equations (9) and (7) we get:

$$U \;=\; e^{i\alpha} e^{-i\frac{\beta+\gamma+\delta}{2}} J(0)J(\beta)J(\gamma)J(\delta)$$

We conclude in particular, that $J(\alpha)$ generates all 1-qubit unitary operators.

Next, we turn to the decomposition of $\wedge U$ in terms of $J(\alpha)$ and $\wedge Z$. Subscripts to operators indicate the qubit to which they apply, and we sometimes abbreviate $J_i(\alpha)$ as J_i^{α}.

Suppose U has J-decomposition $e^{i\alpha}J(0)J(\beta)J(\gamma)J(\delta)$, then $\wedge U$ can also be decomposed as follows:

$$\wedge U_{12} \;=\; J_1^0 J_1^{\alpha'} J_2^0 J_2^{\beta+\pi} J_2^{-\frac{\gamma}{2}} J_2^{-\frac{\pi}{2}} J_2^0 \wedge Z_{12} J_2^{\frac{\pi}{2}} J_2^{\frac{\gamma}{2}} J_2^{\frac{-\pi-\delta-\beta}{2}} J_2^0 \wedge Z_{12} J_2^{\frac{-\beta+\delta-\pi}{2}}$$

with $\alpha' = \alpha + \frac{\beta+\gamma+\delta}{2}$.

To prove the above decomposition, we first define auxiliary unitary operators:

$$\begin{aligned} A &= J(0)J(\beta+\pi)J(-\tfrac{\gamma}{2})J(-\tfrac{\pi}{2}) \\ B &= J(0)J(\tfrac{\pi}{2})J(\tfrac{\gamma}{2})J(\tfrac{-\pi-\delta-\beta}{2}) \\ C &= J(0)J(\tfrac{-\beta+\delta-\pi}{2}) \end{aligned}$$

Then, using the additivity relation we obtain $ABC = I$. On the other hand, using both the subtractivity relation and equations (6), we get:

$$\begin{aligned} AXBXC &= J(0)J(\beta+\pi)J(-\tfrac{\gamma}{2})J(-\tfrac{\pi}{2})J(\pi)J(\tfrac{\pi}{2})J(\tfrac{\gamma}{2})J(\tfrac{-\pi-\delta-\beta}{2})J(\pi)J(\tfrac{-\beta+\delta-\pi}{2}) \\ &= e^{-i\frac{\delta+\beta+\gamma}{2}} J(0)J(\beta)J(\gamma)J(\delta) \end{aligned}$$

Therefore one also has $e^{i\frac{2\alpha+\beta+\gamma+\delta}{2}} AXBXC = U$.

Combining our two equations in A, B, C, we obtain $\wedge U_{12} = P_1(\alpha')A_2 \wedge X_{12} B_2 \wedge X_{12} C_2$ with $\alpha' = \alpha + \frac{\beta+\gamma+\delta}{2}$; a decomposition which we can rewrite using our generating set:

$$\begin{aligned} P(\alpha)_1 &= J_1^0 J_1^{\alpha} \\ \wedge X_{12} &= H_2 \wedge Z_{12} H_2 \;=\; J_2^0 \wedge Z_{12} J_2^0 \end{aligned}$$

to obtain the above decomposition of $\wedge U$.

Having all unitaries U over $\mathbb{C}^2$ and all unitaries of the form $\wedge U$ over $\mathbb{C}^2 \otimes \mathbb{C}^2$ we can conclude that:

Theorem 3 (Universality) *The set $\{J(\alpha), \wedge Z\}$ generates all unitaries.*

The following unitaries $H = J(0)$, $P(\frac{\pi}{4}) = J(0)J(\frac{\pi}{4})$, and $\wedge X = J(0) \wedge Z J(0)$, are known to be *approximately universal*, in the sense that any unitary can be approximated within any precision by combining these [NC00]. Therefore the set $J(0)$, $J(\frac{\pi}{4})$ and $\wedge Z$ is also approximately universal.

It is easy to verify that the following patterns implement our generators

$$\begin{aligned} \mathcal{J}(\alpha) &:= X_2^{s_1} M_1^{-\alpha} E_{12} \\ \wedge\mathcal{Z} &:= E_{12} \end{aligned}$$

where in the first pattern 1 is the only input and 2 is the only output, while in the second both 1 and 2 are inputs and outputs (note that we are allowing patterns to have overlapping inputs and outputs). Combining these two patterns, by composition and tensoring, will therefore generate patterns realising all unitaries over $\otimes^n \mathbb{C}^2$. These patterns are indeed among the simplest possible. Remarkably, there is only one single dependency overall, which occurs in the correction phase of $\mathcal{J}(\alpha)$. No set of patterns without any measurement could be a generating set, since those can only implement unitaries in the Clifford group.

F Measurement Calculus

We turn now to the structural result on MBQC asserting that the key factorisation property, namely that entanglement can be done first, and then local measurements, can be reduced to confluence properties of a simple algebraic rewriting system [DKP07].

The expressions appearing as commands are all linear operators on Hilbert space. At first glance, the appropriate equality between commands is equality as operators. For the deterministic commands, the equality that we consider is indeed equality as operators. This equality implies equality in the denotational semantics. However, for measurement commands one needs a stricter definition for equality in order to be able to apply them as rewriting rules. Essentially we have to take into the account the effect of different branches that might result from the measurement process. The precise definition is below.

Definition 11 *Given two patterns $\mathcal{P}$ and $\mathcal{P}'$ we define $\mathcal{P} = \mathcal{P}'$ if and only if for any branch s, we have $A_s^{\mathcal{P}} = A_s^{\mathcal{P}'}$, where $A_s^{\mathcal{P}}$ and $A_s^{\mathcal{P}'}$ are the branch map A_s defined in Section D.*

The first set of equations gives the means to propagate local Pauli corrections through the entangling operator E_{ij}.

$$\begin{aligned} E_{ij}X_i^s &= X_i^s Z_j^s E_{ij} && (10)\\ E_{ij}X_j^s &= X_j^s Z_i^s E_{ij} && (11)\\ E_{ij}Z_i^s &= Z_i^s E_{ij} && (12)\\ E_{ij}Z_j^s &= Z_j^s E_{ij} && (13)\end{aligned}$$

These equations are easy to verify and are natural since E_{ij} belongs to the Clifford group, and therefore maps under conjugation the Pauli group to itself. Note that, despite the symmetry of the E_{ij} operator *qua* operator, we have to consider all the cases, since the rewrite system defined below does not allow one to rewrite E_{ij} to E_{ji}. If we did allow this the rewrite process could loop forever.

A second set of equations allows one to push corrections through measurements acting on the same qubit. Again there are two cases:

$$\begin{aligned} {}^t[M_i^\alpha]^s X_i^r &= {}^t[M_i^\alpha]^{s+r} && (14)\\ {}^t[M_i^\alpha]^s Z_i^r &= {}^{t+r}[M_i^\alpha]^s && (15)\end{aligned}$$

These equations follow easily from equations (4) and (5). They express the fact that the measurements M_i^α are closed under conjugation by the Pauli group, very much like equations (10),(11),(12) and (13) express the fact that the Pauli group is closed under conjugation by the entanglements E_{ij}.

Define the following convenient abbreviations:

$$[M_i^\alpha]^s := {}^0[M_i^\alpha]^s \quad {}^t[M_i^\alpha] := {}^t[M_i^\alpha]^0 \quad M_i^\alpha := {}^0[M_i^\alpha]^0 \quad M_i^x := M_i^0 \quad M_i^y := M_i^{\frac{\pi}{2}}$$

Particular cases of the equations above are:

$$\begin{array}{rcl} M_i^x X_i^s & = & M_i^x \\ M_i^y X_i^s & = & [M_i^y]^s \quad = \quad {}^s[M_i^y] \quad = \quad M_i^y Z_i^s \end{array}$$

The first equation, follows from the fact that $-0 = 0$, so the X action on M_i^x is trivial; the second equation, is because $-\frac{\pi}{2}$ is equal $\frac{\pi}{2} + \pi$ modulo 2π, and therefore the X and Z actions coincide on M_i^y. So we obtain the following:

$$\begin{array}{rcl} {}^t[M_i^x]^s & = & {}^t[M_i^x] \\ {}^t[M_i^y]^s & = & {}^{s+t}[M_i^y] \end{array} \qquad \begin{array}{r} (16) \\ (17) \end{array}$$

which we will use later to prove that patterns with measurements of the form M^x and M^y may only realise unitaries in the Clifford group.

We now define a set of rewrite rules, obtained by orienting the equations above. Recall that patterns are executed from right to left:

$$\begin{array}{lcll} E_{ij} X_i^s & \Rightarrow & X_i^s Z_j^s E_{ij} & EX \\ E_{ij} X_j^s & \Rightarrow & X_j^s Z_i^s E_{ij} & EX \\ E_{ij} Z_i^s & \Rightarrow & Z_i^s E_{ij} & EZ \\ E_{ij} Z_j^s & \Rightarrow & Z_j^s E_{ij} & EZ \\ {}^t[M_i^\alpha]^s X_i^r & \Rightarrow & {}^t[M_i^\alpha]^{s+r} & MX \\ {}^t[M_i^\alpha]^s Z_i^r & \Rightarrow & {}^{r+t}[M_i^\alpha]^s & MZ \end{array}$$

to which we need to add the *free commutation rules*, obtained when commands operate on disjoint sets of qubits:

$$\begin{array}{lcll} E_{ij} A_{\vec{k}} & \Rightarrow & A_{\vec{k}} E_{ij} & \text{where } A \text{ is not an entanglement} \\ A_{\vec{k}} X_i^s & \Rightarrow & X_i^s A_{\vec{k}} & \text{where } A \text{ is not a correction} \\ A_{\vec{k}} Z_i^s & \Rightarrow & Z_i^s A_{\vec{k}} & \text{where } A \text{ is not a correction} \end{array}$$

where $\vec{k}$ represent the qubits acted upon by command A, and are supposed to be distinct from i and j. Clearly these rules could be reversed since they hold as equations but we are orienting them this way in order to obtain termination. Condition (D) is easily seen to be preserved under rewriting.

Under rewriting, the computation space, inputs and outputs remain the same, and so do the entanglement commands. Measurements might be modified, but there is still the same number of them, and they still act on the same qubits. The only induced modifications concern local corrections and dependencies. If there was no dependency at the start, none will be created in the rewriting process.

In order to obtain rewrite rules, it was essential that the entangling command ($\wedge Z$) belongs to the normaliser of the Pauli group. The point is that the Pauli operators are the correction operators and they can be dependent, thus we can commute the entangling commands to the beginning without inheriting any dependency. Therefore the entanglement resource can indeed be prepared at the outset of the computation.

Write $\mathcal{P} \Rightarrow \mathcal{P}'$, respectively $\mathcal{P} \Rightarrow^\star \mathcal{P}'$, if both patterns have the same type, and one obtains the command sequence of $\mathcal{P}'$ from the command sequence of $\mathcal{P}$ by applying one, respectively any number, of the rewrite rules of the previous section. We say that $\mathcal{P}$ is *standard* if for no $\mathcal{P}'$, $\mathcal{P} \Rightarrow \mathcal{P}'$ and the procedure of writing a pattern to standard form is called standardisation. We use the word "standardisation" instead of the more usual "normalisation" in order not to cause terminological confusion with the physicists' notion of normalisation.

One of the most important results about the rewrite system is that it has the desirable properties of determinacy (confluence) and termination (standardisation). In other words, we will show that for all $\mathcal{P}$, there exists a unique standard $\mathcal{P}'$, such that $\mathcal{P} \Rightarrow^\star \mathcal{P}'$. It is, of course, crucial that the standardisation process leaves the semantics of patterns invariant. This is the subject of the next simple, but important, proposition,

Proposition 12 *Whenever $\mathcal{P} \Rightarrow^\star \mathcal{P}'$, $[\![\mathcal{P}]\!] = [\![\mathcal{P}']\!]$.*

Proof. It is enough to prove it when $\mathcal{P} \Rightarrow \mathcal{P}'$. The first group of rewrites has been proved to be sound in the preceding subsections, while the free commutation rules are obviously sound. □

We now begin the main proof of this section. First, we prove termination.

Theorem 4 (Termination) *All rewriting sequences beginning with a pattern $\mathcal{P}$ terminate after finitely many steps. For our rewrite system, this implies that for all $\mathcal{P}$ there exist finitely many $\mathcal{P}'$ such that $\mathcal{P} \Rightarrow^\star \mathcal{P}'$ where the $\mathcal{P}'$ are standard.*

Proof. Suppose $\mathcal{P}$ has command sequence $A_n \dots A_1$; so the number of commands is n. Let $e \leq n$ be the number of E commands in $\mathcal{P}$. As we have noted earlier, this number is invariant under $\Rightarrow$. Moreover E commands in $\mathcal{P}$ can be ordered by increasing depth, read from right to left, and this order, written $<_E$, is also invariant, since EE commutations are forbidden explicitly in the free commutation rules.

Define the following depth function d on E and C commands in $\mathcal{P}$:

$$d(A_i) = \begin{cases} i & \text{if } A_i = E_{jk} \\ n - i & \text{if } A_i = C_j \end{cases}$$

Define further the following sequence of length e, $d_E(\mathcal{P})(i)$ is the depth of the E-command of rank i according to $<_E$. By construction this sequence is strictly increasing. Finally, we define the measure $m(\mathcal{P}) := (d_E(\mathcal{P}), d_C(\mathcal{P}))$ with:

$$d_C(\mathcal{P}) \quad = \quad \textstyle\sum_{C \in \mathcal{P}} d(C)$$

We claim the measure we just defined decreases lexicographically under rewriting, in other words $\mathcal{P} \Rightarrow \mathcal{P}'$ implies $m(\mathcal{P}) > m(\mathcal{P}')$, where $<$ is the lexicographic ordering on $\mathbb{N}^{e+1}$.

To clarify these definitions, consider the following example. Suppose $\mathcal{P}$'s command sequence is of the form $EXZE$, then $e = 2$, $d_E(\mathcal{P}) = (1, 4)$, and $m(\mathcal{P}) = (1, 4, 3)$. For the command sequence EEX we get that $e = 2$, $d_E(\mathcal{P}) = (2, 3)$ and $m(\mathcal{P}) = (2, 3, 2)$. Now, if one considers the rewrite $EEX \Rightarrow EXZE$, the measure of the left hand side is $(2, 3, 2)$, while the measure of the right hand side, as said, is $(1, 4, 3)$, and indeed $(2, 3, 2) > (1, 4, 3)$. Intuitively the reason is clear: the Cs are being pushed to the left, thus decreasing the depths of Es, and concomitantly, the value of d_E.

Let us now consider all cases starting with an EC rewrite. Suppose the E command under rewrite has depth d and rank i in the order $<_E$. Then all Es of smaller rank have same depth in the right hand side, while E has now depth $d-1$ and still rank i. So the right hand side has a strictly smaller measure. Note that when $C = X$, because of the creation of a Z (see the example above), the last element of $m(\mathcal{P})$ may increase, and for the same reason all elements of index $j > i$ in $d_E(\mathcal{P})$ may increase. This is why we are working with a lexicographical ordering.

Suppose now one does an MC rewrite, then $d_C(\mathcal{P})$ strictly decreases, since one correction is absorbed, while all E commands have equal or smaller depths. Again the measure strictly decreases.

Next, suppose one does an EA rewrite, and the E command under rewrite has depth d and rank i. Then it has depth $d-1$ in the right hand side, and all other E commands have invariant depths, since we forbade the case when A is itself an E. It follows that the measure strictly decreases.

Finally, upon an AC rewrite, all E commands have invariant depth, except possibly one which has smaller depth in the case $A = E$, and $d_C(\mathcal{P})$ decreases strictly because we forbade the case where $A = C$. Again the claim follows.

So all rewrites decrease our ordinal measure, and therefore all sequences of rewrites are finite, and since the system is finitely branching (there are no more than n possible single step rewrites on a given sequence of length n), we get the statement of the theorem. □

The next theorem establishes the important determinacy property and furthermore shows that the standard patterns have a certain canonical form which we call the NEMC form. The precise definition is:

Definition 13 *A pattern has a* NEMC *form if its commands occur in the order of Ns first, then Es , then Ms, and finally Cs.*

We will usually just say "EMC" form since we can assume that all the auxiliary qubits are prepared in the $|+\rangle$ state and we usually just elide these N commands.

Theorem 5 (Confluence) *For all $\mathcal{P}$, there exists a unique standard $\mathcal{P}'$, such that $\mathcal{P} \Rightarrow^\star \mathcal{P}'$, and $\mathcal{P}'$ is in EMC form.*

Proof. Since the rewriting system is terminating, confluence follows from local confluence by Newman's lemma, see, for example, [Bar84]. This means that whenever two rewrite rules can be applied to a term t yielding t_1 and t_2, one can rewrite both t_1 and t_2 to a common third term t_3, possibly in many steps. Then the uniqueness of the standard form is an immediate consequence.

In order to prove the local confluence we look for critical pairs, that is occurrences of three successive commands where two rules can be applied simultaneously. One finds that there are only five types of critical pairs, of these the three involve the N command, these are of the form: NMC, NEC and NEM; and the remaining two are: $E_{ij}M_kC_k$ with i, j and k all distinct, $E_{ij}M_kC_l$ with k and l distinct. In all cases local confluence is easily verified.

Suppose now $\mathcal{P}'$ does not satisfy the EMC form conditions. Then, either there is a pattern EA with A not of type E, or there is a pattern AC with A not of type C. In the former case, E and A must operate on overlapping qubits, else one may apply a free commutation rule, and A may not be a C since in this case one may apply an EC rewrite. The only remaining case is when A is of type M, overlapping E's qubits, but this is what condition (D1) forbids, and since (D1) is preserved under rewriting, this contradicts the assumption. The latter case is even simpler. □

We have shown that under rewriting any pattern can be put in EMC form, which is what we wanted. We actually proved more, namely that the standard form obtained is unique. However, one has to be a bit careful about the significance of this additional piece of information. Note first that uniqueness is obtained because we dropped the CC and EE free commutations, thus having a rigid notion of command sequence. One cannot put them back as rewrite rules, since they obviously ruin termination and uniqueness of standard forms.

A reasonable thing to do, would be to take this set of equations as generating an equivalence relation on command sequences, call it $\equiv$, and hope to strengthen the results obtained so far, by proving that all reachable standard forms are equivalent.

But this is too naive a strategy, since $E_{12}X_1X_2 \equiv E_{12}X_2X_1$, and:

$$\begin{array}{rcl} E_{12}X_1^sX_2^t & \Rightarrow^\star & X_1^sZ_2^sX_2^tZ_1^tE_{12} \\ & \equiv & X_1^sZ_1^tZ_2^sX_2^tE_{12} \end{array}$$

obtaining an expression which is not symmetric in 1 and 2. To conclude, one has to extend $\equiv$ to include the additional equivalence $X_1^sZ_1^t \equiv Z_1^tX_1^s$, which fortunately is sound since these two operators are equal up to a global phase. Thus, these are all equivalent in our semantics of patterns. We summarise this discussion as follows.

Definition 14 *We define an equivalence relation $\equiv$ on patterns by taking all the rewrite rules as equations and adding the equation $X_1^sZ_1^t \equiv Z_1^tX_1^s$ and generating the smallest equivalence relation.*

With this definition we can state the following proposition.

Proposition 15 *All patterns that are equivalent by $\equiv$ are equal in the denotational semantics.*

This $\equiv$ relation preserves both the type (the (V, I, O) triple) and the underlying entanglement graph. So clearly semantic equality does not entail equality up to $\equiv$. In fact, by composing teleportation patterns one obtains infinitely many patterns for the identity which are all different up to $\equiv$. One may wonder whether two patterns with same semantics, type and underlying entanglement graph are necessarily equal up to $\equiv$. This is not true either. One has $J(\alpha)J(0)J(\beta) = J(\alpha+\beta) = J(\beta)J(0)J(\alpha)$ (where $J(\alpha)$ is defined in Section E), and this readily gives a counter-example. We can now formally describe a simple standardisation algorithm.

Algorithm. Input*: A pattern $\mathcal{P}$ on $|V| = N$ qubits with command sequence $A_M \cdots A_1$.*
Output*: An equivalent pattern $\mathcal{P}'$ in NEMC form.*

1. *Commute all the preparation commands (new qubits) to the right side.*

2. *Commute all the correction commands to the left side using the EC and MC rewriting rules.*

3. *Commute all the entanglement commands to the right side after the preparation commands.*

Note that since each qubit can be entangled with at most $N-1$ other qubits, and can be measured or corrected only once, we have $O(N^2)$ entanglement commands and $O(N)$ measurement commands. According to the definiteness condition, no command acts on a qubit not yet prepared, hence the first step of the above algorithm is based on trivial commuting rules; the same is true for the last step as no entanglement command can act on a qubit that has been measured. Both steps can be done in $O(N^2)$ time. The real complexity of the algorithm comes from the second step and the EX commuting rule. In the worst case scenario, commuting an X correction to the left might create $O(N^2)$ other Z corrections, each of which has to be commuted to the left themselves.

Thus one can have at most $O(N^3)$ new corrections, each of which has to be commuted past $O(N^2)$ measurement or entanglement commands. Therefore the second step, and hence the algorithm, has a worst case complexity of $O(N^5)$ time.

We conclude this subsection by emphasising the importance of the EMC form. Since the entanglement can always be done first, we can always derive the entanglement resource needed for the whole computation right at the beginning. After that only local operations will be performed. This will separate the analysis of entanglement resource requirements from the classical control. Furthermore, this makes it possible to extract the maximal parallelism for the execution of the pattern since the necessary dependencies are explicitly expressed, see [BK09] for further discussion. The EMC form also provides us with tools to prove general theorems about patterns, such as the fact that they always compute cptp-maps and the expressiveness theorems [DKP07]. Finally, we present later the first MBQC protocol designed using the EMC form which allows one to clearly distinguish between the quantum and classical aspects of a quantum computation.

G Determinism

An important aspect of MBQC is the way the inherent randomness of the measurement outcomes can be accounted for, so that the overall computation remains deterministic. This is accomplished by conditioning the basis of certain measurements upon the outcome of others, introducing a measurement order. We present first various notions of determinism. A pattern is said to be *deterministic* if it realises a cptp-map that sends pure states to pure states. This is equivalent to saying that for a deterministic pattern branch maps are proportional, that is to say, for all $q \in \mathfrak{H}_I$ and all $\mathbf{s_1}$, $\mathbf{s_2} \in \mathbb{Z}_2^n$, $A_{\mathbf{s_1}}(q)$ and $A_{\mathbf{s_2}}(q)$ differ only up to a scalar. The class of deterministic patterns include projections. A more restricted class contains all the unitary and unitary embedding operators: a pattern is said to be *strongly deterministic* when branch maps are equal (up to a global phase), *i.e.* for all $\mathbf{s}_1$, $\mathbf{s}_2 \in \mathbb{Z}_2^n$, $A_{\mathbf{s}_1} = e^{i\phi_{s_1,s_2}} A_{\mathbf{s}_2}$. These are the patterns implementing quantum algorithms and hence understanding their structural properties is of particular interest.

Proposition 16 *If a pattern is strongly deterministic, then it realises a unitary embedding.*

Proof. Define T to be the map realised by the pattern. We have $T = \sum_{\mathbf{s}} A_{\mathbf{s}}^\dagger A_{\mathbf{s}}$. Since the pattern in strongly deterministic all the branch maps are the same. Define A to be $2^{n/2} A_{\mathbf{s}}$, then A must be a unitary embedding, because $A^\dagger A = I$. □

An important sub-class of deterministic patterns are robust under the changes of the angles: a pattern is said to be *uniformly deterministic* if it is deterministic for all values of its measurement angles. In another words a uniformly deterministic pattern defines a class of quantum operators that can be performed given the same initial entanglement resources. On the other hand it is known that if we fix the angle of measurements to be Pauli the obtained operators is in Clifford group [DKP07]. That means uniform determinism allow us to associate to a family of quantum operators a canonical pattern implementing a Clifford operator, a potential valuable abstract reduction for the study of quantum operators.

Finally a pattern is said to be *stepwise deterministic* if it is deterministic after performing each single measurement together with all the corrections depending on the result of that measurement. In another words a pattern is stepwise deterministic if after each single measurements there exists a set of local corrections depending only on the result of this measurements to be performed on some or all of the non-measured qubits that will make the two branches equal (up to a global phase).

A variety of methods for constructing measurement patterns have been already proposed [RBB03, HEB04, CLN05a] that guarantee determinism by construction. We introduce a direct condition on graph states which guarantees a strong form of deterministic behaviour for a class of MBQC patterns defined over them [DK06]. Remarkably, this condition bears only on the geometric structure of the entangled graph states.

Let us define an *open graph state* (G, I, O) to be a state associated with an undirected graph G together with two subsets of nodes I and O, called inputs and outputs. We write V for the set of nodes in G, I^c, and O^c for the complements of I and O in V, $N_G(i)$ for the set of neighbours of i in G, $i \sim j$ for $(i,j) \in G$, and $E_G := \prod_{i \sim j} E_{ij}$ for the global entanglement operator associated to G. In what follows, $x \sim y$ denotes that x is adjacent to y in G, N_{I^c} denotes the sequence of preparation commands $\prod_{i \in I^c} N_i$.

Definition 17 *A* flow $(f, \preceq)$ *for an open graph state* (G, I, O) *consists of a map* $f : O^c \to I^c$ *and a partial order* $\preceq$ *over* V *such that for all* $x \in O^c$*:*

(i) $x \sim f(x)$;

(ii) $x \preceq f(x)$;

(iii) for all $y \sim f(x)$, $x \preceq y$.

As one can see, a flow consists of two structures: a function f over vertices and a matching partial order over vertices. In order to obtain a deterministic pattern for an open graph state with flow, dependent corrections will be defined based on function f. The order of the execution of the commands is given by the partial order induced by the flow. The matching properties between the function f and the partial order $\preceq$ will make the obtained pattern runnable. Figure 2 shows an open graph state together with a flow, where f is represented by arcs from O^c (measured qubits, black vertices) to I^c (prepared qubits, non boxed vertices). The associated partial order is given by the labelled sets of vertices. The coarsest order $\preceq$ for which $(f, \preceq)$ is a flow is called the *dependency order* induced by f and its depth (4 in Figure 2) is called *flow depth.*

The existence of a causal flow is a sufficient condition for determinism. First we need the following simple lemma which describes an essential property of graph state.

Lemma 18 *For any open graph* (G, I, O) *and any* $i \in I^c$,

$$E_G N_{I^c} = X_i Z_{N_G(i)} E_G N_{I^c}$$

Proof. The proof is based on equations 10, 12 of the Measurement Calculus, and the additional equation $X_i N_i = N_i$, which follows from the fact that N_i produces a qubit in the $|+\rangle$ state which is a fix point of X.

$$\begin{aligned}
E_G N_{I^c} &= E_G X_i N_{I^c} \\
&= \Big(\prod_{(k,l)\in G, k\neq i, l\neq i} E_{k,l}\Big) \Big(\prod_{j\in N_G(i)} E_{i,j}\Big) X_i N_{I^c} \\
&= \Big(\prod_{(k,l)\in G, k\neq i, l\neq i} E_{k,l}\Big) \Big(X_i \prod_{j\in N_G(i)} Z_j\Big) \Big(\prod_{j\in N_G(i)} E_{i,j}\Big) N_{I^c} \\
&= \Big(X_i \prod_{j\in N_G(i)} Z_j\Big) E_G N_{I^c} \\
&= X_i Z_{N_G(i)} E_G N_{I^c}
\end{aligned}$$

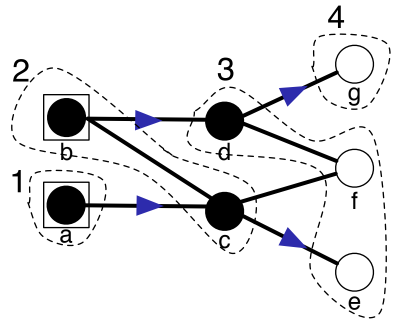

Figure 2: An open graph state with flow. The boxed vertices are the input qubits and the white vertices are the output qubits. All the non-output qubits, black vertices, are measured during the run of the pattern. The flow function is represented as arcs and the partial order on the vertices is given by the 4 partition sets.

□

The operator $K_i := X_i(\prod_{j\in N_G(i)} Z_j)$ is called *graph stabiliser* [HEB04] at qubit i and the above lemma proves $K_i E_G N_{I^c} = E_G N_{I^c}$. Note that this equation is slightly more general than the common graph stabiliser [HEB04] as it can be applied to open graph states where input qubits are prepared in arbitrary states.

Theorem 6 *Suppose the open graph state (G, I, O) has flow f, then the pattern:*

$$\mathcal{P}_{f,G,\vec{\alpha}} \quad := \quad \prod_{i\in O^c}^{\preceq} \Big(X_{f(i)}^{s_i} \prod_{\substack{k\sim f(i)\\ k\neq i}} Z_k^{s_i} M_i^{\alpha_i} \Big) E_G$$

where the product follows the dependency order $\preceq$ of f, is uniformly and strongly deterministic, and realises the unitary embedding:

$$U_{G,I,O,\vec{\alpha}} \quad := \quad 2^{|O^c|/2} \Big(\prod_{i\in O^c} \langle +_{\alpha_i}|_i \Big) E_G$$

Proof. The proof is based on *anachronical* patterns, i.e. patterns which do not satisfy the D0 condition (see section C) saying that no command depends on an outcome not yet measured. Indeed, in the anachronical pattern $M_i^\alpha Z_i^{s_i}$, the command $Z_i^{s_i}$ depends on the outcome s_i whereas the qubit i is not yet measured. However, by relaxing the D0 condition, we have the following equation:

$$\langle +_\alpha|_i = M_i^\alpha Z_i^{s_i}$$

Indeed, if $s_i = 0$ the measurement realises the projection $\langle +_\alpha|_i$, and if $s_i = 1$ the measurement realises the projection $\langle -_\alpha|_i = \langle +_\alpha|_i Z_i$. Thus, any correction-free pattern $\prod_{i\in O^c} M_i^{\alpha_i} E_G N_{I^c}$ can be turned into an anachronical strongly deterministic pattern $\prod_{i\in O^c} M_i^{\alpha_i} Z_i^{s_i} E_G N_{I^c}$ which realises U_G. The rest of the proof consists in transforming this anachronical pattern into a pattern which satisfies the D0 condition:

$$\begin{array}{rcl}\prod_{i\in O^c} M_i^{\alpha_i} Z_{s_i}^i E_G N_{I^c} & = & \prod_{i\in O^c} M_i^{\alpha_i} Z_i^{s_i} \left(X_{f(i)}^{s_i} \prod_{j \in N_G(f(i))} Z_j^{s_i} \right) E_G N_{I^c} \\ & = & \prod_{i\in O^c}^{\prec} \left(X_{f(i)}^{s_i} \prod_{j \in N_G(f(i)) \smallsetminus \{i\}} Z_j^{s_i} \right) M_i^{\alpha_i} E_G N_{I^c}\end{array}$$

Lemma 18 and condition 3 of the causal flow are used in the previous equation for eliminating the command $Z_{s_i}^i$, whereas conditions 1 and 2 ensure that the pattern satisfies the D0 condition.□

The intuition of the proof is that entanglement between two qubits i and j converts an anachronical Z correction at i, given in the term $M_i^\alpha Z_i^{s_i}$, into a pair of a 'future' X correction on qubit j. The existence of the flow is only a sufficient condition for determinism which assign to every single measured qubit a unique correcting vertices $f(i)$. A natural generalisation is to consider a set of vertices as a *correcting set* which leads to a full characterisation of determinism [BKMP07].

Having obtained the rigourous mathematical model underlying MBQC, we can now present how this model suggests new techniques for designing quantum protocols.

H Universal Blind Quantum Computing

When the technology to build quantum computers becomes available, it is likely that it will only be accessible to a handful of centres around the world. Much like today's rental system of supercomputers, users will probably be granted access to the computers in a limited way. How will a user interface with such a quantum computer? Here, we consider the scenario where a user is unwilling to reveal the computation that the remote computer is to perform, but still wishes to exploit this quantum resource. The solution is *Universal Blind Quantum Computing* (UBQC) protocol [BFK09] that allows a client Alice (who does not have any quantum computational resources or quantum memory) to interact with a server Bob (who has a quantum computer) in order for Alice to obtain the outcome of her target computation such that privacy is preserved. This means that Bob learns nothing about Alice's inputs, outputs, or desired computation. The privacy is perfect, does not rely on any computational assumptions, and holds no matter what actions a cheating Bob undertakes. Alice only needs to be able to prepare single qubits randomly chosen from a finite set and send them to the server, who has the balance of the required quantum computational resources. After this initial preparation, Alice and Bob use two-way classical communication which enables Alice to drive the computation by giving single-qubit measurement instructions to Bob, depending on previous measurement outcomes. Note that if Alice wanted to compute the solution to a classical problem in NP, she could efficiently verify the outcome. An interfering Bob is not so obviously detected in other cases. UBQC uses an authentication technique which performs this detection.

The UBQC protocol is constructed using the unique feature of MBQC that separates the classical and quantum parts of a computation, leading to a generic scheme for blind computation of any circuit without requiring any quantum memory for Alice. This is fundamentally different from previously known classical or quantum schemes. UBQC can be viewed as a distributed version of an MBQC computation (where Alice prepares the individual qubits, Bob does the entanglement and measurements, and Alice computes the classical feedforward mechanism), on top of which randomness is added in order to obscure the computation from Bob's point of view. This is the first time that a new functionality has been achieved thanks to MBQC (other theoretical advances due to MBQC appear in [RHG06, MS08]). From a conceptual point of view, this shows that MBQC has tremendous potential for the development of new protocols, and maybe even of algorithms.

UBQC can be used for any quantum circuit and also works for quantum inputs or outputs. We now give some applications.

- *Factoring.* Factoring is a prime application of UBQC: by implementing Shor's factoring algorithm [Sho97] as a blind quantum computation, Alice can use Bob to help her factor a product of large primes which is associated with an RSA public key [RSA78]. Thanks to the properties of UBQC, Bob will not only be unable to determine Alice's input, but will be completely oblivious to the fact that he is helping her factor.

- BQP*-complete problem.* UBQC could be used to help Alice solve a BQP-complete problem, for instance approximating the Jones polynomial [AJL06]. There is no known classical method to efficiently verify the solution; this motivates the need for authentication of Bob's computation, even in the case that the output is classical.

- *Processing quantum information.* Alice may wish to use Bob as a remote device to manipulate quantum information. Consider the case where Alice is participating in a quantum protocol such as a quantum interactive proof. She can use UBQC to prepare a quantum state, to perform a measurement on a quantum system, or to process quantum inputs into quantum outputs.

- *Quantum prover interactive proofs.* UBQC can be used to accomplish an interactive proof for any language in BQP, with a quantum prover and a nearly-classical verifier, where the verifier requires the power to generate random qubits chosen from a fixed set. Moreover, UBQC can be adapted to provide a two-prover interactive proof for any problem in BQP with a purely classical verifier. The modification requires that the provers share entanglement but otherwise be unable to communicate. Guided by the verifier, the first prover measures his part of the entanglement in order to create a shared resource between the verifier and the second prover. The remainder of the interaction involves the verifier and the second prover who essentially run the main protocol.

In the classical world, Feigenbaum introduced the notion of *computing with encrypted data* [Fei86], according to which a function f is *encryptable* if Alice can easily transform an instance x into instance x', obtain $f(x')$ from Bob and efficiently compute $f(x)$ from $f(x')$, in such a way that Bob cannot infer x from x'. Following this, Abadi, Feigenbaum and Kilian [AFK89] gave an impossibility result: no NP-hard function can be computed with encrypted data (even probabilistically and with polynomial interaction), unless the polynomial hierarchy collapses at the third level.

Ignoring the blindness requirement of UBQC yields an interactive proof with a BQP prover and a nearly-classical verifier. This scenario was first proposed in the work of [ABE08], using very different techniques based on authentication schemes. Their protocol can be also used for blind quantum computation. However, their scheme requires that Alice have quantum computational resources and memory to act on a constant-sized register. A related classical protocol for the scenario involving a P prover and a nearly-linear time verifier was given in [GKR08].

Returning to the cryptographic scenario, still in the model where the function is classical and public, Arrighi and Salvail [AS06] gave an approach using quantum resources. The idea of their protocol is that Alice gives Bob multiple quantum inputs, most of which are *decoys*. Bob applies the target function on all inputs, and then Alice verifies his behaviour on the decoys. There are two important points to make here. First, the protocol only works for a restricted set of classical functions called *random verifiable*: it must be possible for Alice to efficiently generate random input-output pairs. Second, the protocol does not prevent Bob from learning Alice's private input; it provides only *cheat sensitivity.*

The case of a blind *quantum* computation was first considered by Childs [Chi05] based on the idea of encrypting input qubits with a quantum one-time pad [AMTW00, BR03]. At each step, Alice sends the encrypted qubits to Bob, who applies a known quantum gate (some gates requiring further interaction with Alice). Bob returns the quantum state, which Alice decrypts using her key. Cycling through a fixed set of universal gates ensures that Bob learns nothing about the circuit. The protocol requires fault-tolerant quantum memory and the ability to apply local Pauli operators at each step, and does not provide any method for the detection of malicious errors.

The UBQC protocol [BFK09] is the first protocol for universal blind quantum computation where Alice has no quantum memory that works for *any* quantum circuit and assumes Alice has a classical computer, augmented with the power to prepare single qubits randomly chosen in

$$\{1/\sqrt{2}\left(|0\rangle + e^{i\theta}|1\rangle\right) \mid \theta = 0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}\,.$$

The required quantum and classical communication between Alice and Bob is linear in the size of Alice's desired quantum circuit. Interestingly, it is sufficient for our purposes to restrict Alice's classical computation to modulo 8 arithmetic! Similar observations in a non-cryptographic context have been made in [AB09]. Except for an unavoidable leakage of the size of Alice's data [AFK89], Alice's privacy is perfect. We provide an authentication technique to detect an interfering Bob with overwhelming probability; this is optimal since there is always an exponentially small probability that Bob can guess a path that will make Alice accept.

All previous protocols for blind quantum computation require technology for Alice that is today unavailable: Arrighi and Salvail's protocol requires multi-qubit preparations and measurements, Childs' protocol requires fault-tolerant quantum memory and the ability to apply local Pauli operators at each step, while Aharonov, Ben-Or and Eban's protocol requires a constant-sized quantum computer with memory. In sharp contrast to this, from Alice's point of view, UBQC can be implemented with physical systems that are already available and well-developed. The required apparatus can be achieved by making only minor modifications to equipment used in the BB84 key exchange protocol [BB84].

I The Brickwork States

The family of graph states called *cluster states* [RB01] is universal for MBQC, however, the method that allows arbitrary computation on the cluster state consists in first tailoring the cluster state to the specific computation by performing some computational basis measurements. If one was to use this principle or any arbitrary graph sates for blind quantum computing, Alice would have to reveal information about the structure of the underlying graph. Instead UBQC uses a new family of states called the *brickwork states* (Figure 3) which are universal for $X-Y$ plane measurements and thus do not require the initial computational basis measurements. Other universal graph states for that do not require initial computational basis measurements have appeared in [CLN05b].

Definition 19 *A brickwork state $\mathcal{G}_{n\times m}$, where $m \equiv 5 \pmod 8$, is an entangled state of $n \times m$ qubits constructed as follows (see also Figure 3):*

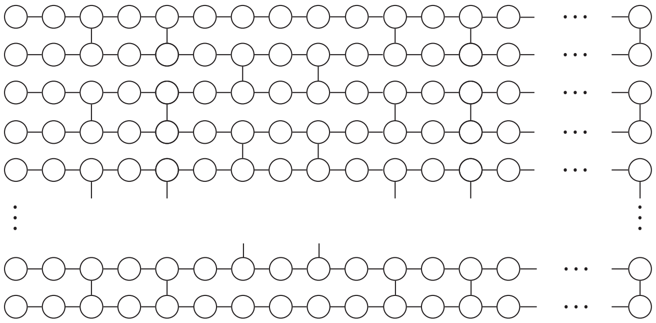

Figure 3: The *brickwork state, $\mathcal{G}_{n\times m}$*. Qubits $|\psi_{x,y}\rangle$ ($x = 1,\dots,n, y = 1,\dots,m$) are arranged according to layer x and row y, corresponding to the vertices in the above graph, and are originally in the $|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ state. Controlled-Z gates are then performed between qubits which are joined by an edge.

1. *Prepare all qubits in state $|+\rangle$ and assign to each qubit an index (i,j), i being a column ($i \in [n]$) and j being a row ($j \in [m]$).*
2. *For each row, apply the operator* CTRL-Z *on qubits (i,j) and $(i,j+1)$ where $1 \leq j \leq m-1$.*
3. *For each column $j \equiv 3 \pmod 8$ and each odd row i, apply the operator* CTRL-Z *on qubits (i,j) and $(i+1,j)$ and also on qubits $(i,j+2)$ and $(i+1,j+2)$.*
4. *For each column $j \equiv 7 \pmod 8$ and each even row i, apply the operator* CTRL-Z *on qubits (i,j) and $(i+1,j)$ and also on qubits $(i,j+2)$ and $(i+1,j+2)$.*

Theorem 7 (Universality) *The brickwork state $\mathcal{G}_{n\times m}$ is universal for quantum computation. Furthermore, we only require single-qubit measurements under the angles $\{0, \pm\pi/4, \pm\pi/2\}$, and measurements can be done layer-by-layer.*

Proof. It is well-known that the set $U = \{\text{CTRL-}X, H, \frac{\pi}{8}\}$ is a universal set of gates; we will show how the brickwork state can be used to compute any gate in U. Recall the *rotation* transformations: $X(\theta) = e^{\frac{i\theta X}{2}}$ and $Z(\theta) = e^{\frac{i\theta Z}{2}}$.

Consider the measurement pattern and underlying graph state given in Figure 4. The implicit required corrections are implemented according to the flow condition [DK06] which guarantees determinism, and allows measurements to be performed layer-by-layer. The action of the measurement of the first three qubits on each wire is clearly given by the rotations in the right-hand part of Figure 4 [BB06]. The circuit identity follows since CTRL-Z commutes with $Z(\alpha)$ and is self-inverse.

By assigning specific values to the angles, we get the Hadamard gate (Figure 5), the $\pi/8$ gate (Figure 6) and the identity (Figure 7). By symmetry, we can get H or $\pi/8$ acting on logical qubit 2 instead of logical qubit 1.

In Figure 8, we give a pattern and show using circuit identities that it implements a CTRL-X. The verification of the circuit identities is straightforward. Again by symmetry, we can reverse the control and target qubits. Note that as long as we have CTRL-Xs between any pair of neighbours, this is sufficient to implement CTRL-X between further away qubits.

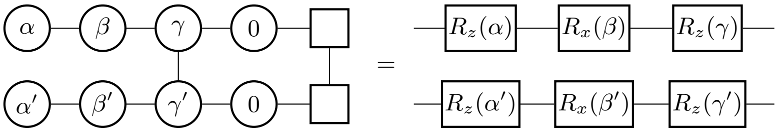

Figure 4: Pattern with arbitrary rotations. Squares indicate output qubits.

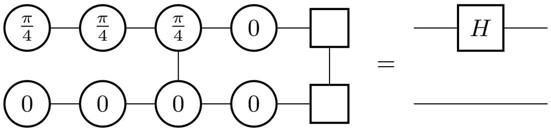

Figure 5: Implementation of a Hadamard gate.

We now show how we can tile the patterns as given in Figures 4 through 8 (the underlying graph states are the same) to implement any circuit using U as a universal set of gates. In Figure 9, we show how a 4-qubit circuit with three gates, U_1, U_2 and U_3 (each gate acting on a maximum of two adjacent qubits) can be implemented on the brickwork state $G_{9,4}$. We have completed the top and bottom logical wires with a pattern that implements the identity. Generalising this technique, we get the family of brickwork states as given in Figure 3 and Definition 19. □

Here we only consider approximate universality. This allows us to restrict the angles of preparation and measurement to a finite set and hence simplify the description of the protocol. However one can easily extend UBQC to achieve exact universality as well, provided Alice can communicate real numbers to Bob.

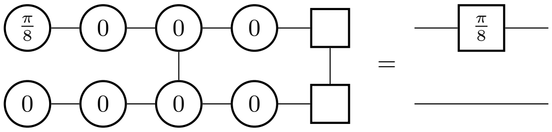

Figure 6: Implementation of a $\pi/8$ gate.

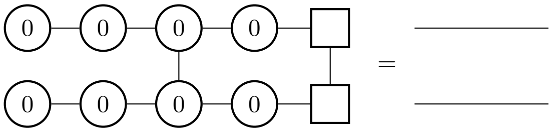

Figure 7: Implementation of the identity.

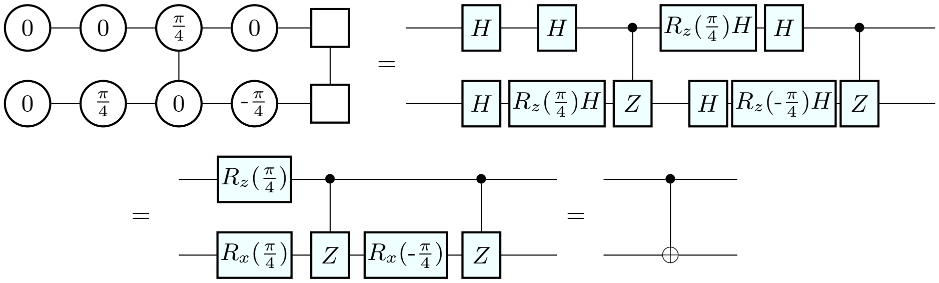

Figure 8: Implementation of a CTRL-X.

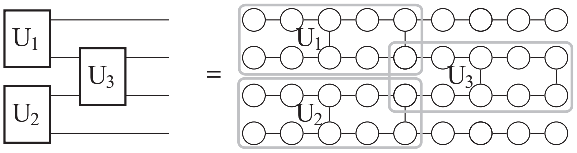

Figure 9: Tiling for a 4-qubit circuit with three gates.

J The UBQC Protocol

Suppose Alice has in mind a unitary operator U that is implemented with a pattern on a brickwork state $\mathcal{G}_{n\times m}$ (Figure 3) with measurements given as multiples of $\pi/4$. This pattern could have been designed either directly in MBQC or from a circuit construction. Each qubit $|\psi_{x,y}\rangle \in \mathcal{G}_{n\times m}$ is indexed by a *column* $x \in \{1,\ldots,n\}$ and a *row* $y \in \{1,\ldots,m\}$. Thus each qubit is assigned: a measurement angle $\phi_{x,y}$, a set of X-dependencies $D_{x,y} \subseteq [x-1]\times[m]$, and a set of Z-dependencies $D'_{x,y} \subseteq [x-1]\times[m]$. Here, we assume that the dependency sets $X_{x,y}$ and $Z_{x,y}$ are obtained via the flow construction [DK06]. During the execution of the pattern, the actual measurement angle $\phi'_{x,y}$ is a modification of $\phi_{x,y}$ that depends on previous measurement outcomes in the following way: let $s^X_{x,y} = \oplus_{i\in D_{x,y}} s_i$ be the parity of all measurement outcomes for qubits in $X_{x,y}$ and similarly, $s^Z_{x,y} = \oplus_{i\in D'_{x,y}} s_i$ be the parity of all measurement outcomes for qubits in $Z_{x,y}$. Then $\phi'_{x,y} = (-1)^{s^X_{x,y}}\phi_{x,y} + s^Z_{x,y}\pi$. **Protocol 1** implements a blind quantum computation for U. Note that we assume that Alice's input to the computation is built into U. In other words, Alice wishes to compute $U|0\rangle$, her input is classical and the first layers of U may depend on it.

Protocol 1 Universal Blind Quantum Computation

1. Alice's preparation

For each column $x = 1,\ldots,n$

For each row $y = 1,\ldots,m$

1.1 Alice prepares $|\psi_{x,y}\rangle \in_R \{|+_{\theta_{x,y}}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\theta_{x,y}}|1\rangle) \mid \theta_{x,y} = 0, \pi/4, \ldots, 7\pi/4\}$ and sends the qubits to Bob.

2. Bob's preparation

2.1 Bob creates an entangled state from all received qubits, according to their indices, by applying CTRL-Z gates between the qubits in order to create a brickwork state $\mathcal{G}_{n\times m}$ (see Definition 19).

3. Interaction and measurement

For each column $x = 1,\ldots,n$

For each row $y = 1,\ldots,m$

3.1 Alice computes $\phi'_{x,y}$ where $s^X_{0,y} = s^Z_{0,y} = 0$.

3.2 Alice chooses $r_{x,y} \in_R \{0,1\}$ and computes $\delta_{x,y} = \phi'_{x,y} + \theta_{x,y} + \pi r_{x,y}$.

3.3 Alice transmits $\delta_{x,y}$ to Bob. Bob measures in the basis $\{|+_{\delta_{x,y}}\rangle, |-_{\delta_{x,y}}\rangle\}$.

3.4 Bob transmits the result $s_{x,y} \in \{0,1\}$ to Alice.

3.5 If $r_{x,y} = 1$ above, Alice flips $s_{x,y}$; otherwise she does nothing.

The universality of **Protocol 1** follows from the universality of brickwork state for measurement-based quantum computing. *Correctness* refers to the fact that the outcome of the protocol is the same as the outcome if Alice had run the pattern herself. The fact that Protocol 1 correctly computes $U|0\rangle$ follows from the commutativity of Alice's rotations and Bob's measurements in the rotated bases. This is formalised below.

Theorem 8 (Correctness) *Assume Alice and Bob follow the steps of **Protocol 1**. Then the outcome is correct.*

Proof. Firstly, since CTRL-Z commutes with Z-rotations, steps 1 and 2 do not change the underlying graph state; only the phase of each qubit is locally changed, and it is as if Bob had done the Z-rotation after the CTRL-Z. Secondly, since a measurement in the $|+_\phi\rangle, |-_\phi\rangle$ basis on a state $|\psi\rangle$ is the same as a measurement in the $|+_{\phi+\theta}\rangle, |-_{\phi+\theta}\rangle$ basis on $Z(\theta)|\psi\rangle$, and since $\delta = \phi' + \theta + \pi r$, if $r = 0$, Bob's measurement has the same effect as Alice's target measurement; if $r = 1$, all Alice needs to do is flip the outcome. □

We now define and prove the security of the protocol. Intuitively, we wish to prove that whatever Bob chooses to do (including arbitrary deviations from the protocol), his knowledge on Alice's quantum computation does not increase. Note, however that Bob does learn the dimensions of the brickwork state, giving an upper bound on the size of Alice's computation. This is unavoidable: a simple adaptation of the proof of Theorem 2 from [AFK89], confirms this. We incorporate this notion of leakage in our definition of *blindness*. A *quantum delegated computation* protocol is a protocol by which Alice interacts quantumly with Bob in order to obtain the result of a computation, $U(x)$, where $X = (\tilde{U}, x)$ is Alice's input with $\tilde{U}$ being a description of U.

Definition 20 *Let P be a quantum delegated computation on input X and let $L(X)$ be any function of the input. We say that a quantum delegated computation protocol is* blind while leaking at most L(X) *if, on Alice's input X, for any fixed $Y = L(X)$, the following two hold when given Y:*

1. *The distribution of the classical information obtained by Bob in P is independent of X.*
2. *Given the distribution of classical information described in 1, the state of the quantum system obtained by Bob in P is fixed and independent of X.*

Definition 20 captures the intuitive notion that Bob's view of the protocol should not depend on X (when given Y); since his view consists of classical and quantum information, this means that the distribution of the classical information should not depend on X (given Y) and that for any fixed choice of the classical information, the state of the quantum system should be uniquely determined and not depend on X (given Y). We are now ready to state and prove our main theorem. Recall that in **Protocol 1**, (n, m) is the dimension of the brickwork state.

Theorem 9 (Blindness) ***Protocol 1** is blind while leaking at most (n, m).*

Proof. Let (n, m) (the dimension of the brickwork state) be given. Note that the universality of the brickwork state guarantees that Bob's creating of the graph state does not reveal anything on the underlying computation (except n and m).

Alice's input consists of

$$\phi = (\phi_{x,y} \mid x \in [n], y \in [m])$$

with the actual measurement angles

$$\phi' = (\phi'_{x,y} \mid x \in [n], y \in [m])$$

being a modification of ϕ that depends on previous measurement outcomes. Let the classical information that Bob gets during the protocol be

$$\delta = (\delta_{x,y} \mid x \in [n], y \in [m])$$

and let A be the quantum system initially sent from Alice to Bob.

To show independence of Bob's classical information, let $\theta'_{x,y} = \theta_{x,y} + \pi r_{x,y}$ (for a uniformly random chosen $\theta_{x,y}$) and $\theta' = (\theta'_{x,y} \mid x \in [n], y \in [m])$. We have $\delta = \phi' + \theta'$, with θ' being uniformly random (and independent of ϕ and/or ϕ'), which implies the independence of δ and ϕ.

As for Bob's quantum information, first fix an arbitrary choice of δ. Because $r_{x,y}$ is uniformly random, for each qubit of A, one of the following two has occurred:

1. $r_{x,y} = 0$ so $\delta_{x,y} = \phi'_{x,y} + \theta'_{x,y}$ and $|\psi_{x,y}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i(\delta_{x,y} - \phi'_{x,y})}|1\rangle$.

2. $r_{x,y} = 1$ so $\delta_{x,y} = \phi'_{x,y} + \theta'_{x,y} + \pi$ and $|\psi_{x,y}\rangle = \frac{1}{\sqrt{2}}(|0\rangle - e^{i(\delta_{x,y} - \phi'_{x,y})}|1\rangle$.

Since δ is fixed, θ' depends on ϕ' (and thus on ϕ), but since $r_{x,y}$ is independent of everything else, without knowledge of $r_{x,y}$ (i.e. taking the partial trace of the system over Alice's secret), A consists of copies of the two-dimensional completely mixed state, which is fixed and independent of ϕ. □

There are two malicious scenarios that are covered by Definition 20 and that we explicitly mention here. Suppose Bob has some prior knowledge, given as some *a priori* distribution on Alice's input X. Since Definition 20 applies to any distribution of X, we can simply apply it to the conditional distribution representing the distribution of X given Bob's a priori knowledge; we conclude that Bob does not learn any information on X beyond what he already knows, as well as what is leaked. The second scenario concerns a Bob whose goal it is to find Alice's output. Definition 20 forbids this: learning information on the output would imply learning information on Alice's input.

Note that the protocol does not allow Alice to reveal to Bob whether or not she accepts the result of the computation as this bit of information could be exploited by Bob to learn some information about the actual computation. In this scenario, **Protocol 4** can be used instead.

K Quantum Inputs and Outputs

We can slightly modify **Protocol 1** to deal with both quantum inputs and outputs. In the former case, no extra channel resources are required, while the latter case requires a quantum channel from Bob to Alice in order for him to return the output qubits. Alice will also need to be able to apply X and Z Pauli operators in order to undo the quantum one-time pad. Note that these protocols can be combined to obtain a protocol for quantum inputs and outputs.

Consider the scenario where Alice's input is the form of m physical qubits and she has no efficient classical description of the inputs to be able to incorporate it into **Protocol 1**. In this case, she needs to be able to apply local Pauli-X and Pauli-Z operators to implement a full one-time pad over the input qubits. The first layer of measurements are adapted to undo the Pauli-X operation if necessary. By the quantum one-time pad, Theorem 8 and Theorem 9, this modified protocol, given in **Protocol 2** is still correct and private.

Here we assume that Alice already has in her hands the quantum inputs: unless she receives the inputs one-by-one, she requires for this initial step some quantum memory. She also needs to be able to apply the single-qubit gates as described above. Note that this is only asking slightly more than Alice choosing between four single-qubit gates, which would be the minimum required in *any* blind quantum computation protocol with quantum inputs.

Suppose Alice now requires a quantum output, for example in the case of blind quantum state preparation. In this scenario, instead of measuring the last layer of qubits, Bob returns it to Alice,

Protocol 2 Universal Blind Quantum Computation with Quantum Inputs

1. Alice's input preparation

For the input column ($x = 0, y = 1, \ldots, m$) corresponding to Alice's input

1.1 Alice applies $Z_{0,y}(\theta_{0,y})$ for $\theta_{0,y} \in_R \{0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$.

1.2 Alice chooses $i_{0,y} \in_R \{0,1\}$ and applies $X_{0,y}^{i_{0,y}}$. She sends the qubits to Bob.

2. Alice's auxiliary preparation

For each column $x = 1, \ldots, n$

For each row $y = 1, \ldots, m$

2.1 Alice prepares $|\psi_{x,y}\rangle \in_R \{|+_{\theta_{x,y}}\rangle \mid \theta_{x,y} = 0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$ and sends the qubits to Bob.

3. Bob's preparation

3.1 Bob creates an entangled state from all received qubits, according to their indices, by applying CTRL-Z gates between the qubits in order to create a brickwork state $\mathcal{G}_{(n+1)\times m}$.

4. Interaction and measurement

For each column $x = 0, \ldots, n$

For each row $y = 1, \ldots, m$

4.1 Alice computes $\phi'_{x,y}$ with the special case $\phi'_{0,y} = (-1)^{i_{0,y}}\phi_{0,y}$.

4.2 Alice chooses $r_{x,y} \in_R \{0,1\}$ and computes $\delta_{x,y} = \phi'_{x,y} + \theta_{x,y} + \pi r_{x,y}$.

4.3 Alice transmits $\delta_{x,y}$ to Bob.

4.4 Bob measures in the basis $\{|+_{\delta_{x,y}}\rangle, |-_{\delta_{x,y}}\rangle\}$.

4.5 Bob transmits the result $s_{x,y} \in \{0,1\}$ to Alice.

4.6 If $r_{x,y} = 1$ above, Alice flips $s_{x,y}$; otherwise she does nothing.

who performs the final layer of Pauli corrections. The following theorem shows a privacy property on the quantum states that Bob manipulates.

Theorem 10 *At every step of **Protocol 1**, Bob's quantum state is one-time padded.*

Proof. During the execution of the protocol the value of s^X and s^Z are unknown to Bob since they have been one-time padded using the random key r at each layer. Due to the flow construction [DK06], each qubit (starting at the third column) receives independent Pauli operators, which act as the full quantum one-time pad over Bob's state. Since our initial state is $|+\rangle$, and since the first layer performs a hidden Z-rotation, it follows that the qubits in the second layer are also completely encrypted during the computation. □This result together with Theorems 8 and 9 proves the correctness and privacy of **Protocol 3** that deals with quantum outputs.

L Authentication and Fault-Tolerance

We now focus on Alice's ability to detect if Bob is not cooperating. There are two possible ways in which Bob can be uncooperative: he can refuse to perform the computation (this is immediately apparent to Alice), or he can actively interfere with the computation, while pretending to follow the protocol. It is this latter case that we focus on detecting. The authentication technique enables Alice to detect an interfering Bob with overwhelming probability (strictly speaking, either Bob's interference is corrected and he is not detected, or his interference is detected with overwhelming probability). Note that this is the best that we can hope for since nothing prevents Bob from refusing to perform the computation. Bob could also be lucky and guess a path that Alice will accept. This happens with exponentially small probability, hence our technique is optimal.

In the case that Alice's computation has a classical output and that she does not require fault-tolerance, a simple protocol for blind quantum computing with authentication exists: execute **Protocol 1**, on a modification of Alice's target circuit: she adds N randomly placed trap wires that are randomly in state $|0\rangle$ or $|1\rangle$ (N is the number of qubits in the computation). If Bob interferes, either his interference has no effect on the classical output, or he will get caught with probability at least $\frac{1}{2}$ (he gets caught if Alice finds that the output of at least one trap wire is incorrect). The protocol is repeated s times (the traps are randomly re-positioned each time); if Bob is not caught cheating, Alice accepts if all outputs are identical; otherwise she rejects. The probability of an incorrect output being accepted is at most 2^{-s}.

Protocol 4 is more general than this scheme since it works for quantum outputs and is fault-tolerant. If the above scheme is used for quantum inputs, they must be given to Alice as multiple copies. Similarly (but more realistically), if **Protocol 4** is to be used on quantum inputs, these must already be given to Alice in an encoded form as in **step 2** of **Protocol 4** (because Alice has no quantum computational power). In the case of a quantum output, it will be given to Alice in a known encoded form, which she can pass on to a third party for verification.

The theory of quantum error correction provides a natural mechanism for detecting unintended changes to a computation, whereas the theory of fault-tolerant computation provides a way to process information even using error-prone gates. Unfortunately, error correction, even when combined with fault-tolerant gate constructions is insufficient to detect malicious tampering if the error correction code is known. As evidenced by the quantum authentication protocol [BCG$^+$02], error correction encodings can, however, be adapted for this purpose.

Protocol 3 Universal Blind Quantum Computation with Quantum Outputs

1. **Alice's auxiliary preparation**
   For each column $x = 1, \ldots, n-1$
    For each row $y = 1, \ldots, m$

   1.1 Alice prepares $|\psi_{x,y}\rangle \in_R \{|+_{\theta_{x,y}}\rangle \mid \theta_{x,y} = 0, \pi/4, 2\pi/4, \ldots, 7\pi/4\}$ and sends the qubits to Bob.

2. **Alice's output preparation**

   2.1 Alice prepares the last column of qubits $|\psi_{n,y}\rangle = |+\rangle$ $(y = 1, \ldots, m)$ and sends the qubits to Bob.

3. **Bob's preparation**

   3.1 Bob creates an entangled state from all received qubits, according to their indices, by applying CTRL-Z gates between the qubits in order to create a brickwork state $\mathcal{G}_{n \times m}$.

4. **Interaction and measurement**
   For each column $x = 1, \ldots, n-1$
    For each row $y = 1, \ldots, m$

   4.1 Alice computes $\phi'_{x,y}$ where $s^X_{0,y} = s^Z_{0,y} = 0$ for the first column.

   4.2 Alice chooses $r_{x,y} \in_R \{0, 1\}$ and computes $\delta_{x,y} = \phi'_{x,y} + \theta_{x,y} + \pi r_{x,y}$.

   4.3 Alice transmits $\delta_{x,y}$ to Bob.

   4.4 Bob measures in the basis $\{|+_{\delta_{x,y}}\rangle, |-_{\delta_{x,y}}\rangle\}$.

   4.5 Bob transmits the result $s_{x,y} \in \{0, 1\}$ to Alice.

   4.6 If $r_{x,y} = 1$ above, Alice flips $s_{x,y}$; otherwise she does nothing.

5. **Output Correction**

   5.1 Bob sends to Alice all qubits in the last layer.

   5.2 Alice performs the final Pauli corrections $Z^{s^Z_{n,y}} X^{s^X_{n,y}}$.

Protocol 4 Blind Quantum Computing with Authentication (classical input and output)

1. Alice chooses $\mathbb{C}$, where $\mathbb{C}$ is some $n_{\mathbb{C}}$-qubit error-correcting code with distance $d_{\mathbb{C}}$. The security parameter is $d_{\mathbb{C}}$.

2. In the circuit model, starting from circuit for U, Alice converts target circuit to fault-tolerant circuit:

   2.1 Use error-correcting code $\mathbb{C}$. The encoding appears in the initial layers of the circuit.

   2.2 Perform all gates and measurements fault-tolerantly.

   2.3 Some computational basis measurements are required for the fault-tolerant implementation (for verification of ancillae and non-transversal gates). Each measurement is accomplished by making and measuring a *pseudo-copy* of the target qubit: a CTRL-X is performed from the target to an ancilla qubit initially set to $|0\rangle$, which is then measured in the Z-basis.

   2.4 Ancilla qubit wires are evenly spaced through the circuit.

   2.5 The ancillae are re-used. All ancillae are measured at the same time, at regular intervals, after each fault-tolerant gate (some outputs may be meaningless).

3. Within each encoded qubit, permute all wires, keeping these permutations secret from Bob.

4. Within each encoded qubit, add $3n_T$ randomly interspersed *trap* wires, each trap being a random eigenstate of X, Y or Z (n_T of each). For security, we must have $n_T \propto n_{\mathbb{C}}$; for convenience, we choose $n_T = n_{\mathbb{C}}$. The trap qubit wire (at this point) does not interact with the rest of the circuit. The wire is initially $|0\rangle$, and then single-qubit gates are used to create the trap state. These single-qubit gates appear in the initial layers of the circuit.

5. Trap qubits are verified using the same ancillae as above: they are rotated into the computational basis, measured using the pseudo-copy technique above, and then returned to their initial basis.

6. Any fault-tolerant measurement is randomly interspersed with verification of $3n_T$ random trap wires. For this, identity gates are added as required.

7. For classical output, the trap wires are rotated as a last step, so that the following measurement in the computational basis is used for a final verification.

8. Convert the whole circuit above to a measurement-based computation on the brickwork state, with the addition of regular Z-basis measurements corresponding to the measurements on ancillae qubits above. Swap and identity gates are added as required, and trap qubits are left untouched.

9. Perform the blind quantum computation:

   9.1 Execute **Protocol 1**, to which we add that Alice periodically instructs Bob to measure in Z-basis as indicated above.

   9.2 Alice uses the results of the trap qubit measurements to estimate the error rate; if it is below the threshold (see discussion in the main text), she accepts, otherwise she rejects.

UBQC proceeds along the following lines. Alice chooses an $n_{\mathbb{C}}$-qubit error correction code $\mathbb{C}$ with distance $d_{\mathbb{C}}$. (The values of $n_{\mathbb{C}}$ and $d_{\mathbb{C}}$ are taken as security parameters.) If the original computation involves N logical qubits, the authenticated version involves $N(n_{\mathbb{C}} + 3n_T)$ (with $n_T = n_{\mathbb{C}}$), logical qubits: throughout the computation, each logical qubit is encoded with $\mathbb{C}$, while the remaining $3Nn_T$ qubits are used as traps to detect an interfering Bob. The trap qubits are prepared as a first step of the computation in eigenstates of the Pauli operators X, Y and Z, with an equal number of qubits in each state.

The protocol also involves fault-tolerant gates, for some of which it is necessary to have Bob periodically measure qubits [ZCC07]. In order to accomplish this, the blind computation protocol is extended by allowing Alice to instruct Bob to measure specific qubits within the brickwork state in the computational basis at regular intervals. These qubits are chosen at regular spacial intervals so that no information about the structure of the computation is revealed. It should be noted that in **Protocol 4**, we allow Alice to reveal to Bob whether or not she accepts the final result.

UBQC can also be used in the scenario of non-malicious faults: because it already uses a fault-tolerant construction, the measurement of trap qubits in **Protocol 4** allows for the estimation of the error rate (whether caused by the environment or by an adversary); if this error rate is below a certain threshold (this threshold is chosen below the fault-tolerance threshold to take into account sampling errors), Alice accepts the computation. As long as this is below the fault-tolerance threshold, an adversary would still have to guess which qubits are part of the code, and which are traps, so Theorem 13 also holds in the fault-tolerant version. The only difference is that the adversary can set off a few traps without being detected, but he must still be able to correctly guess which qubits are in the encoded qubit and which are traps. Increasing the security parameters will make up for the fact that Bob can set off a few traps without making the protocol abort. This yields a linear trade-off between the error rate and the security parameter. Note that the brickwork state (Figure 3) can be extended to multiple dimensions, which may be useful for obtaining better fault-tolerance thresholds [Got00]. While the quantum Singleton bound [KL00] allows error correction codes for which $d_{\mathbb{C}} \propto n_{\mathbb{C}}$, it may be more convenient to use the Toric Code [Kit97] for which $d_{\mathbb{C}} \propto \sqrt{n_{\mathbb{C}}}$, as this represents a rather simple encoding while retaining a high ratio of $d_{\mathbb{C}}$ to $n_{\mathbb{C}}$. For the special case of deterministic classical output, a classical repetition code is sufficient and preferable as such an encoding maximises $n_{\mathbb{C}}$.

Theorem 11 (Fault Tolerance) ***Protocol 4*** *is fault-tolerant.*

Proof. By construction, the circuit created in step 2.1 is fault-tolerant. Furthermore, the permutation of the circuit wires and insertion of trap qubits (steps 2.2 and 2.3) preserves the fault tolerance. This is due to the fact that qubits are permuted only within blocks of constant size. The fault-tolerant circuit given in step 2.1 can be written as a sequence of local gates and CTRL-X gates between neighbours. Clearly permutation does not affect the fidelity of local operations. As qubits which are neighbours in the initial fault-tolerant circuit become separated by less than twice the number of qubits in a single block, the maximum number of nearest-neighbour CTRL-X gates required to implement CTRL-X from the original circuit is in $O(n_{\mathbb{C}} + 3n_T)$ (the size of a block). (If required, the multi-dimensional analogue of the two-dimensional brickwork state can be used in order to substantially reduce this distance.) As this upper bound is constant for a given implementation, a lower bound for the fault-tolerance threshold can be obtained simply be scaling the threshold such that the error rate for this worst-case CTRL-X is never more than the threshold for the original circuit. Thus, while the threshold is reduced, it remains non-zero.

Step 8 converts the fault-tolerant circuit to a measurement pattern; it is known that this transformation retains the fault-tolerance property [ND05, AL06]. Finally, in step 9, distributing the fault-tolerant measurement pattern between Alice and Bob does not disturb the fault tolerance since the communication between them is only classical. □

Theorem 12 (Blindness) ***Protocol 4*** *is blind while leaking at most* (n, m).

Proof. **Protocol 4** differs from **Protocol 1** in the following two ways: Alice instructs Bob to perform regular Z-basis measurements and she reveals whether or not she accepts or rejects the computation. It is known that Z measurements change the underlying graph state into a new graph state [HEB04]. The Z measurements in the protocol are inserted at regular intervals and their numbers are also independent of the underlying circuit computation. Therefore their action transforms the generic brickwork state into another generic resource still independent of Alice's input and the blindness property is obtained via the same proof of Theorem 9. Finally, from Alice's decision to accept or reject, only information relating to the trap qubits is revealed to Bob, since Alice rejects if and only if the estimated error rate is too high. The trap qubits are uncorrelated with the underlying computation (in the circuit picture, they do not interact with the rest of the circuit) and hence they reveal no information about Alice's input. □

In the following theorem, for simplicity, we consider the scenario with zero error rate; a proof for the full fault-tolerant version is similar.

Theorem 13 (Authentication) *For the zero-error case of* ***Protocol 4****, if Bob interferes with an authenticated computation, then either he is detected except with exponentially small probability (in the security parameter), or his actions fail to alter the computation.*

Proof. If Bob interferes with the computation, then in order for his actions to affect the outcome of the computation without being detected, he must perform a non-trivial operation (*i.e.* an operation other than the identity) on the subspace in which the logical qubits are encoded. Due to the fault-tolerant construction of Alice's computation (Theorem 11), Bob's operation must have weight at least $d_{\mathbb{C}}$. Due to discretisation of errors, we can treat Bob's action as introducing a Pauli error with some probability p.

If a Pauli error acts non-trivially on a trap qubit then the probability of this going undetected is 1/3. Pauli operators which remain within the code space must act on at least $d_{\mathbb{C}}$ qubits. As Bob has no knowledge about the roles of qubits (Theorem 12), the probability of him acting on any qubit is equal. As the probability of acting on a trap is $3n_T/(n_{\mathbb{C}} + 3n_T)$, for each qubit upon which he acts non-trivially, the probability of Bob being detected is $2n_T/(n_{\mathbb{C}}+3n_T)$. Thus the probability of an M-qubit Pauli operator going undetected is below $(1 - 2n_T/(n_{\mathbb{C}} + 3n_T))^M$. Since $n_T = n_{\mathbb{C}}$, the minimum probability of Bob affecting the computation and going undetected is $\epsilon = 2^{-d_{\mathbb{C}}}$. □

M Entangled Servers

As stated before, one can view UBQC as an interactive proof system where Alice acts as the verifier and Bob as the prover. An important open problem is to find an interactive proof for any problem in BQP with a BQP prover, but with a purely classical verifier. **Protocol 4** makes progress towards finding a solution by providing an interactive proof for any language in BQP, with a quantum prover and a BPP verifier that also has the power to generate random qubits chosen from a fixed set and send them to the prover. This perspective was first proposed by Aharonov, Ben-Or and Eban [ABE08], however their scheme demands a more powerful verifier.

Protocol 5 is a solution to another closely related problem, namely the case of a purely classical verifier interacting with two non-communicating entangled provers. The idea is to adapt **Protocol 1** so that one prover (that we now call a server) is used to prepare the random qubits that would have been generated by Alice in the original protocol, while the other server is used for universal blind quantum computation. Using the authenticated protocol (**Protocol 4**) between Alice and the second server, Alice will detect any cheating servers as clearly, any cheating by Server 1 is equivalent to a deviation from the protocol by Server 2, which is detected in step 2 of the protocol, (the proof is directly obtained from Theorem 13). On the other hand, since Server 2 has access to only half of each entangled state, from his point of view, his sub-system remains in a completely mixed state independently of Server 1's actions and the blindness of the protocol is obtained directly from Theorem 12.

Protocol 5 Universal Blind Quantum Computation with Entangled Servers

Initially, Servers 1 and 2 share $|\Phi^+_{x,y}\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ $(x = 1, \dots, n, y = 1, \dots, m)$.

1. **Alice's preparation with Server 1**
   For each column $x = 1, \dots, n$
   For each row $y = 1, \dots, m$

   1.1 Alice chooses
   $$\tilde{\theta}_{x,y} \in_R \{0, \pi/4, 2\pi/4, \dots, 7\pi/4\}$$
   and sends it to Server 1, who measures his part of $|\Phi^+_{x,y}\rangle$ in $|\pm_{\tilde{\theta}_{x,y}}\rangle$.

   1.2 Server 1 sends $m_{x,y}$, the outcome of his measurement, to Alice.

2. **Alice's computation with Server 2**

   2.1 Alice runs the authenticated blind quantum computing protocol (**Protocol 4**) with Server 2, taking $\theta_{x,y} = \tilde{\theta}_{x,y} + m_{x,y}\pi$.

Acknowledgments

We would like to thank our collaborators and co-authors in the series of the papers that this chapter is based on: Vincent Danos and Prakash Panangaden.

References

[AB09] J. Anders and D. E. Browne. Computational power of correlations. *Physical Review Letters*, 102:050502 [4 pages], 2009.

[ABE08] D. Aharonov, M. Ben-Or, and E. Eban. Interactive proofs for quantum computations. Available as arXiv:0810.5375 [quant-ph], 2008.

[AFK89] M. Abadi, J. Feigenbaum, and J. Kilian. On hiding information from an oracle. *Journal of Computer and System Sciences*, 39:21–50, 1989.

[AJL06] D. Aharonov, V. Jones, and Z. Landau. A polynomial quantum algorithm for approximating the Jones polynomial. In *Proceedings of the 38th annual ACM symposium on Theory of computing (STOC 2006)*, pages 427–436, 2006.

[AL06] P. Aliferis and D. W. Leung. Simple proof of fault tolerance in the graph-state model. *Phys. Rev. A*, 73, 2006.

[AMTW00] A. Ambainis, M. Mosca, A. Tapp, and R. de Wolf. Private quantum channels. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS 2000)*, pages 547–553, 2000.

[AS06] P. Arrighi and L. Salvail. Blind quantum computation. *International Journal of Quantum Information*, 4:883–898, 2006.

[Bar84] H. P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics.* Studies in Logic. North-Holland, 1984.

[BB84] G. Brassard and C. H. Bennett. Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing*, 1984.

[BB06] D. E. Browne and H. J. Briegel. One-way quantum computation. In *Lectures on Quantum Information*, pages 359–380. Wiley-VCH, Berlin, 2006.

[BCG$^+$02] H. Barnum, C. Crépeau, D. Gottesman, A. Smith, and A. Tapp. Authentication of quantum messages. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2002)*, page 449, 2002.

[BFK09] A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal blind quantum computation. In *Proceedings of the 50th Annual Symposium on Foundations of Computer Science*, 2009.

[BK09] A. Broadbent and E. Kashefi. On parallelizing quantum circuits. *Theoretical Computer Science*, 2009.

[BKMP07] D. Browne, E. Kashefi, M. Mhalla, and S. Perdrix. Generalized flow and determinism in measurement-based quantum computation. *New Journal of Physics*, 9, 2007.

[BR03] P. O. Boykin and V. Roychowdhury. Optimal encryption of quantum bits. *Physical Review A*, 67:042317, 2003.

[BV97] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal of Computing*, 5(26), 1997.

[Chi05] A. M. Childs. Secure assisted quantum computation. *Quantum Information and Computation*, 5:456–466, 2005. Initial version appeared online in 2001.

[Cho75] M. D. Choi. Completely positive linear maps on complex matrices. *Linear Algebra and Applications*, 10, 1975.

[CLN05a] A. M. Childs, D. W. Leung, and M. A. Nielsen. Unified derivations of measurement-based schemes for quantum computation. *Physical Review A*, 71, 2005. quant-ph/0404132.

[CLN05b] A. M. Childs, D. W. Leung, and M. A. Nielsen. Unified derivations of measurement-based schemes for quantum computation. *Physical Review A*, 71:032318 [14 pages], 2005.

[DAB03] W. Dür, H. Aschauer, and H. J. Briegel. Multiparticle entanglement purification for graph state. *Physical Review Letters*, 91, 2003. quant-ph/0303087.

[Deu85] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. In *Proceedings of the Royal Society of London*, volume A400, 1985.

[Deu89] D. Deutsch. Quantum computational networks. *Proc. Roy. Soc. Lond A*, 425, 1989.

[DK06] V. Danos and E. Kashefi. Determinism in the one-way model. *Physical Review A*, 74:052310 [6 pages], 2006.

[DKP05] V. Danos, E. Kashefi, and P. Panangaden. Parsimonious and robust realizations of unitary maps in the one-way model. *Physical Review A*, 72, 2005.

[DKP07] V. Danos, E. Kashefi, and P. Panangaden. The measurement calculus. *Journal of ACM*, 54:8 [45 pages], 2007.

[DS96] C. Dürr and M. Santha. A decision procedure for unitary linear quantum cellular automata. In *Proceedings of FOCS'96 – Symposium on Foundations of Computer Science*. LNCS, 1996. quant-ph/9604007.

[Fei86] J. Feigenbaum. Encrypting problem instances: Or ... can you take advantage of someone without having to trust him? In *Proceedings of Advances in Cryptology—CRYPTO 85*, pages 477–488, 1986.

[GC99] D. Gottesman and I. L. Chuang. Quantum teleportation is a universal computational primitive. *Nature*, 402, 1999.

[GKR08] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. Delegating computation: interactive proofs for muggles. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 113–122, 2008.

[GOD$^+$06] A. D. Greentree, P. Olivero, M. Draganski, E. Trajkov, J. R. Rabeau, P. Reichart, B. C. Gibson, S. Rubanov, S. T. Huntington, D. N. Jamieson, and S. Prawer. Critical components for diamond-based quantum coherent devices. *Journal of Physics: Condensed Matter*, 18:825–842, 2006.

[Got97] D. Gottesoman. *Stabilizer codes and quantum error correction.* PhD thesis, California Institute of Technology, 1997.

[Got00] D. Gottesman. Fault-tolerant quantum computation with local gates. *Journal of Modern Optics*, 47:333–345, 2000.

[HEB04] M. Hein, J. Eisert, and H. J. Briegel. Multi-party entanglement in graph states. *Physical Review A*, 69, 2004.

[Kit97] A. Y. Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52:1191–1249, 1997.

[KL00] E. Knill and R. Laflamme. A theory of quantum error-correcting codes. *Physical Review Letters*, 84:2525, 2000.

[MS08] D. Markham and B. C. Sanders. Graph states for quantum secret sharing. *Physical Review A*, 78:042309 [17 pages], 2008.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[ND05] M. A. Nielsen and C. M. Dawson. Fault-tolerant quantum computation with cluster states. *Phys. Rev. A*, 71, 2005.

[Nie03] M. A. Nielsen. Universal quantum computation using only projective measurement, quantum memory, and preparation of the 0 state. *Physical Review A*, 308, 2003.

[Per95] A. Peres. *Quantum Theory: Concepts and Methods.* Kluwer Academic, 1995.

[Pre98] J. Preskill. Fault-tolerant quantum computation. In H. K. Lo, S. Popescu, and T. P. Spiller, editors, *Introduction to Quantum Computation and Information.* World Scientific, 1998.

[RB01] R. Raussendorf and H. J. Briegel. A one-way quantum computer. *Physical Review Letters*, 86, 2001.

[RBB03] R. Raussendorf, D. E. Browne, and H. J. Briegel. Measurement-based quantum computation on cluster states. *Physical Review A*, 68, 2003.

[RHG06] R. Raussendorf, J. Harrington, and K. Goyal. A fault-tolerant one-way quantum computer. *Annals of Physics*, 321:2242–2270, 2006.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

[Sel04] P. Selinger. Towards a quantum programming language. *Mathematical Structures in Computer Science*, 14(4), 2004.

[Sel05] P. Selinger, editor. *Proceedings of the 3nd International Workshop on Quantum Programming Languages*, Electronic Notes in Theoretical Computer Science, 2005.

[Sho97] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26:1484–1509, 1997. First published in 1995.

[SW04] B. Schumacher and R. F. Werner. Reversible quantum cellular automata. quant-ph/0405174, 2004.

[Unr05] D. Unruh. Quantum programs with classical output streams. In Selinger [Sel05].

[vD96] W. van Dam. Quantum cellular automata. Master's thesis, Computer Science Nijmegen, 1996.

[Wat95] J. Watrous. On one-dimensional quantum cellular automata. In *Proceedings of FOCS'95 – Symposium on Foundations of Computer Science*. LNCS, 1995.

[ZCC07] B. Zeng, A. Cross, and I. L. Chuang. Transversality versus universality for additive quantum codes. Available as arXiv:0706.1382v3 [quant-ph], 2007.