On exact counting and quasi-quantum complexity

Niel de Beaudrap∗
Department of Computer Science, University of Oxford
25 September 2015

Abstract

We present characterisations of “exact” gap-definable classes, in terms of indeterministic models of computation which slightly modify the standard model of quantum computation. This follows on work of Aaronson [1], who shows that the counting class \( \text{PP} \) can be characterised in terms of bounded-error “quantum” algorithms which use invertible (and possibly non-unitary) transformations, or postselections on events of non-zero probability. Our work considers similar modifications of the quantum computational model, but in the setting of exact algorithms, and algorithms with zero error and constant success probability. We show that the gap-definable [21] counting classes which bound exact and zero-error quantum algorithms can be characterised in terms of “quantum-like” algorithms involving nonunitary gates, and that postselection and nonunitarity have equivalent power for exact quantum computation only if these classes collapse.

1 Introduction

The relationship between quantum computation (as captured by the class \( \text{BQP} \)), and complexity classes involving classical nondeterminism, is still unclear. It is known that \( \text{BPP} \subseteq \text{BQP} \); is this containment strict? Furthermore, it is not known whether \( \text{NP} \subseteq \text{BQP} \) or \( \text{BQP} \subseteq \text{NP} \) hold, and it is conjectured (see for example Refs. [2, 3]) that neither containment holds: can this be shown? If \( \text{NP} \) and \( \text{BQP} \) are indeed incomparable, is there any natural relationship between quantum computation and nondeterministic Turing machines? These problems are expected to be very difficult. In fact, the best known lower bounds on quantum complexity classes are the classes \( \text{P} \) and \( \text{BPP} \), and the best known upper bounds are gap-definable classes [21] (described below), most of which also do not have a well-understood relationship to \( \text{NP} \).

To look for complexity-theoretic lower bounds for \( \text{BQP} \), one may consider exact or zero-error quantum algorithms. An exact quantum algorithm is one in which the output bit is in one of the pure states \(|0\rangle\) or \(|1\rangle\), according to whether the input is a no or a yes instance. For zero-error quantum algorithms, we allow a bounded probability that the algorithm fails to decide between yes and no instances (indicated by a measurement on some non-output qubit), but require otherwise that the algorithm indicates the correct answer. The sets of problems which can be decided by such algorithms (represented by polynomial-time uniform circuit families), using finite gate sets, are the classes \( \text{EQP} \) [13] and \( \text{ZQP} \) [32, 15] respectively. The class \( \text{BQP} \) is the class of problems which can be decided by such circuits, with bounded error: we then have \( \text{EQP} \subseteq \text{ZQP} \subseteq \text{BQP} \).

One might hope that the exactness constraint might yield an elegant theory of such algorithms, as is the case for other problems in quantum informatics involving one-sided error, such as the zero-error classical capacity of quantum channels [27, 18, 12].

∗niel.debeaudrap@gmail.com
Figure 1: The previously known relations between gap-definable classes [21] (contained in the grey box), and quantum computational complexity classes. The left-most column consists of exact counting classes, including UP (problems decided by NTMs according to whether the number of accepting branches is zero or one) and P (which further requires that there be a total of one branch) for context. The classes SPP $\subseteq$ LWPP $\subseteq$ WPP $\subseteq$ coC $= P$ distinguish between no and yes instances $x \in \{0, 1\}^*$ according to whether the acceptance gap is zero or positive. For WPP, if the gap is positive, we require that it equal some $h(x) > 0$ which is computable from $x$ in deterministic time $O(\text{poly } |x|)$. For LWPP, we further require that $h(x)$ depend only on $|x|$, and for SPP we require that $h$ be constant. Here, $\Delta \text{C}.P := \text{C}.P \cap \text{coC}.P$. The equalities are results of Fenner et al. [23] and Aaronson [1], and the containments EQP $\subseteq$ LWPP and BQP $\subseteq$ AWPP are proven by Fortnow and Rogers [24].

Each of EQP, ZQP, and BQP are bounded above by gap-definable classes [21], which distinguish between yes and no instances via differences between the number of accepting/rejecting branches of nondeterministic Turing machines (NTMs). Figure 1 presents the best previously upper known bounds for these quantum classes, which are EQP $\subseteq$ LWPP, ZQP $\subseteq$ C.P $\cap$ coC.P, and BQP $\subseteq$ AWPP, which follow from Refs. [23, 24]. For the class C.P, we require there be an NTM for which this gap is precisely zero for yes instances, and strictly positive for no instances. For $\Delta \text{C}.P := \text{C}.P \cap \text{coC}.P$, we require that there also be an NTM with the gap conditions reversed: these represent a pair of zero-error randomised algorithms in which exactly one has any bias towards accepting a given input. (Whichever algorithm has a bias, indicates whether the input is a yes or no instance.) For LWPP, we further require that these NTMs have an acceptance gap which is either zero, or an efficiently computable function of the length of the input; the class AWPP is a sort of bounded-error version of LWPP [20].

1 The bound on ZQP follows from ZQP $\subseteq$ NQP $\cap$ coNQP (Proposition 3 on page 6). A problem is in NQP if there are quantum circuits which yield an output of 1 with non-zero probability precisely for the yes instances (in analogy to a probabilistic formulation of NP). Fenner et al. [23] show that NQP = coC.P.
These gap-definable classes may seem quite technical on first encounter. Still, given that BPP can be characterised in terms of the number of just the accepting branches of NTMs, this approach to relate quantum computation to nondeterministic complexity may seem promising if one considers the cancellation of amplitudes (i.e. destructive interference) to be the key distinction between quantum computation and randomised computation. However, as UP is a subset of each of these gap-definable classes, we would have NP ⊆ BQP by Valiant–Vazirani [37] if any of the quantum classes were equal to such a gap-definable class. Thus we might expect these bounds to be loose.

How might we explore the relationship between gap-functions and quantum computation, to find better upper bounds on the quantum complexity classes? One approach is to modify the standard quantum computational model, characterise the power of the modified model, and consider the role played by the modifications in the characterisation. For instance, if one substitutes the real or complex numbers in quantum operations by elements of the ring $\mathbb{Z}_k$ for $k$ a prime power, one obtains new models of computation [9] in which exact or bounded-error computation both efficiently decide any $L \in \text{Mod}_k\text{P}$ [11], classes which again contain UP. This suggests that the complex amplitudes in quantum computation serve to constrain the power of the computational model as much as to empower it, assuming that UP $\not\subseteq$ EQP. Another modification, considered by Aaronson [1], is bounded-error “quantum” algorithms involving invertible (possibly non-unitary) transformations. He shows that for bounded-error computation, this is equivalent to allowing the algorithm to postselect (condition its output on measurement events of non-zero probability), and suffices to decide problems in PP with bounded error. This indicates the computational power of postselection, and suggests that any attempt to separate BQP from PP must somehow account for the unitarity of the operations in quantum algorithms.

**Results.** Extending Aaronson [1], we characterise the power of invertible transformations and of postselection for exact and zero-error quantum algorithms. We first describe a potentially tighter bound EQP $\subseteq$ LPWPP by accounting for the finite gate set of EQP-type algorithms. (This again suggests that finite gate-sets in the zero-error setting may be a strong restriction.) We then characterise LPWPP and LWPP in terms of exact “quantum-like” algorithms, using invertible gates (from an infinite but polynomial-time specifiable [9, §3.3 A] gate-set in the case of LWPP). We also show that $\Delta C. P$ can be characterised either through zero-error quantum-like algorithms using invertible gates, or exact quantum algorithms using postselection. (Thus, postselection and non-unitary operations have equivalent power for exact algorithms only if LWPP = $\Delta C. P$.)

Our results demonstrate that the exact gap-definable classes LWPP and $\Delta C. P$ (and the new class LPWPP) can be described simply in terms of quasi-quantum computation, in which we allow invertible non-unitary gates. At the same time, this shows that the exactness condition per se does not represent a barrier to exact quantum algorithms, as these exact quantum-like classes contain problems of interest (such as GRAPH ISOMORPHISM [7]). Our results also show that unitarity is a significant constraint on quantum algorithms even in the exact setting, where the Born rule plays no role.

**Structure of the paper.** Section 2 presents some preliminaries on quantum complexity and counting classes, defining all of the complexity classes which we use. Section 3 contains a few technical results in gap-function complexity. Section 4 presents our two characterisations of $\Delta C. P$, while Section 5 presents our results on exact quantum-like algorithms with invertible gates. We conclude in Section 6 with some remarks about other gap-definable classes, and open questions.
2 Preliminaries

2.1 Definitions in counting complexity

We begin with some basic terminology relating to the accepting and rejecting branches of poly-time nondeterministic Turing machines (NTMs).

We consider NTMs in normal form [21], in which each nondeterministic transition selects from two possible transitions, and in which the number of nondeterministic transitions made in any computational branch on input $x \in \{0, 1\}^*$ is $q(|x|)$, for some fixed polynomial $q$. We may then represent a computational branch of the NTM by a boolean string $b \in \{0, 1\}^{q(|x|)}$, which we call a branching string of the NTM.

Probabilities can be modelled by NTMs by counting the number of branches (conceived of as arising from unbiased coin-flips) which terminate either in the “accept” or “reject” state. To instead model amplitudes which can destructively interfere, we represent negative contributions to amplitudes by rejecting branches of an NTM, and positive contributions by accepting branches; the difference in the numbers of these represent the accumulated amplitude. This motivates studying complexity classes defined in terms of gap functions [21]:

Definition I. GapP is set of integer-valued functions $g$ on finite input-strings, for which there is a poly-time NTM $N$, such that the difference between the number of accepting and rejecting branches of $N$ is $g(x)$ for each input $x$. We call $g$ the gap function of $N$.

For $g$ the gap function of an NTM in normal form, $g(x)$ is always even; however, by Ref. [21, Lemma 4.3], there will be a gap function $h \in \text{GapP}$ of some other NTM not in normal form, such that $g(x) = 2h(x)$. While we present our results for NTMs $N$ in normal form, we take the liberty of considering “half-gap functions” $h(x) = \frac{1}{2}g(x)$, for $g$ the gap function of $N$.

A counting class is a class of languages which can be decided in terms of the number of accepting or rejecting branches of a poly-time NTM. The class $C_P$ is sometimes referred to as the “exact counting” class, as it may be defined in terms of polytime NTMs, for which an input is a yes instance if and only if the number of its accepting branches is equal to a given polytime-computable function. In this article, we call any complexity class an exact counting class if it distinguishes between yes and no instances according to whether the number of its accepting branches, or its gap function, equals some efficiently computable function. Following Ref. [21], we may define the exact counting classes relevant to our results as follows:

Definition II. We define the classes $C_P$, $\text{coC}_P$, $WPP$, $\text{LWPP}$, and $\text{SPP}$ as follows.

- $C_P$ is the class of languages $L$ for which there is $g \in \text{GapP}$, such that $x \in L$ if and only if $g(x) = 0$; $\text{coC}_P$ is the class of problems for which there is $g \in \text{GapP}$, such that $x \in L$ if and only if $g(x) \neq 0$. We denote $\Delta C_P := C_P \cap \text{coC}_P$.

- $WPP$ is the class of $L \in \Delta C_P$ for which, in addition to the above (for $g$ the gap function of an NTM in normal form), there exists a poly-time computable integer function $h$, such that either $x \notin L$ and $g(x) = 0$ or $x \in L$ and $g(x) = 2h(x) \neq 0$.

- $\text{LWPP}$ is the class of $L \in WPP$ for which, in addition to the above, we may require that $h(x)$ depend only on $|x|$; we then say that $h$ is length-dependent.

- $\text{SPP}$ is the class of $L \in \text{LWPP}$ for which, furthermore, we may require $h(x) = 1$ (or more generally $h(x) \in O(\text{poly } |x|)$ by [21, Theorem 5.9]).
2.2 Upper bounds on quantum complexity

While $C_mP$ is a subject of some interest in counting complexity [11, 35, 21, 23, 33], the classes LWPP and WPP appear to be of more technical importance [24, 34], largely for their relationships to quantum complexity. In each case, the polytime computable function $h$ represents a sort of “normalising factor” for the gap-function $g$, and we distinguish between no and yes instances according to whether $\frac{1}{2}g(x)/h(x) = 0$ or $\frac{1}{2}g(x)/h(x) = 1$. This intuition motivates the following “approximate counting” class corresponding to LWPP and WPP [22]:

**Definition III.** For $\varepsilon > 0$, let $a \approx_b b$ if and only if $|a - b| \leq \varepsilon$. Then $AWPP$ is the class of languages $L$ such that, for any $\varepsilon \in 2^{-O(poly \ n)}$, there is a gap-function $g \in GapP$ and a poly-time computable length-dependent function $h$, such that for all inputs $x$ we have $0 \leq g(x) \leq h(x)$, and either $x$ is a no instance and $g(x)/h(x) \approx_0 0$ or $x$ is a yes instance and $g(x)/h(x) \approx_\varepsilon 1$. Furthermore, we obtain the same class if there are such gap-functions $g$ even if we restrict to $\varepsilon = \frac{1}{3}$ and power functions $h(x) = 2^{f(|x|)}$ [20].

We consider quantum circuits whose gates involve only algebraic coefficients (without loss of generality [4]), given exactly as rational combinations of products of independent algebraic numbers, thus admitting an efficient algorithm for deciding equality. Representing amplitudes by “normalised” gap functions provides intuition for the following:

**Proposition 1** (Fortnow and Rogers [24]). EQP $\subseteq$ LWPP and $BQP \subseteq$ AWPP.

We may refine the upper bound on EQP by defining an intermediate class to SPP and LWPP, in which the length-dependent function evaluates powers of some fixed integer, following the characterisation of AWPP by Fenner [20] described in Definition III:

**Definition IV.** LPWPP is the class of problems in LWPP for which there exists an integer $M \geq 1$, a poly-time computable length-dependent function $h$, and a gap-function $g \in GapP$, such that for all inputs $x$ we have $h(x) = M^t$ for some $t \geq 0$; and either $x$ is a no instance and $g(x) = 0$, or $x$ is a yes instance and $\frac{1}{2}g(x) = h(x)$.

We have SPP $\subseteq$ LPWPP by definition (take $M = 1$ or $t = 0$). As is usual in complexity theory, it is not obvious whether either of the containments SPP $\subseteq$ LPWPP $\subseteq$ LWPP are strict. It is quite plausible that LPWPP $\neq$ LWPP; though perhaps the restriction in the definition of LPWPP of the half-gap functions $h$ to perfect powers may allow such problems to subsumed by SPP.

**Proposition 2.** EQP $\subseteq$ LPWPP.

*Proof.* This containment is implicit in Ref. [24, Theorem 3.8]: the length-dependent poly-time computable function in their proof computes powers of some $M \geq 1$, where $M$ is the common denominator of unitary gate coefficients expressed in rationalised form (i.e. with positive integer denominators). \qed

For zero-error quantum computations, the best known bounds follow from the unbounded-error case. Let NQP be the set of problems for which there is a uniform family $\{C_n\}_{n \geq 1}$ of unitary circuits, for which $C_n \ket{x} = \ket{\psi(x)} \ket{0}$ for some $\ket{\psi(x)}$ if $x$ is a no instance, and where $C_n \ket{x}$ does not factor in this way (i.e. yields the output $\ket{1}$ with non-zero probability) if $x$ is a yes instance. We may then show:
**Proposition 3.** \( \text{ZQP} \subseteq \Delta \text{C}.P \).

*Proof.* Consider a \( \text{ZQP} \) algorithm for a problem, which succeeds with probability \( p \in \Omega(1) \), indicated by the outcome \(|1\rangle\) on the measurement of some qubit \( s \). If \( a \) is the output qubit, there is a non-zero probability that \((s, a)\) is in the state \(|0\rangle\) for \( n \) \( \text{no} \) instances, with no probability of this outcome for \( \text{yes} \) instances; and a non-zero probability that \((s, a)\) is in the state \(|1\rangle\) for \( \text{yes} \) instances, with no probability of this outcome for \( \text{no} \) instances. We may use this to produce \( \text{NQP} \) and \( \text{coNQP} \) algorithms for the problem, so that \( \text{ZQP} \subseteq \text{NQP} \cap \text{coNQP} \). The proposition then follows from \( \Delta \text{C}.P := \text{C}.P \cap \text{coC}.P \) and the result of Fenner et al. [23] that \( \text{NQP} = \text{coC}.P \). \( \square \)

The class \( \text{BQP} \) is widely accepted in the literature as a quantum complexity class of interest. Even the less-studied class \( \text{ZQP} \) contains problems not expected to be solvable with bounded-error by classical algorithms. (As Nishimura and Ozawa [31] point out, using a zero-error primality testing algorithm and Shor’s algorithm as subroutines, we may test whether an integer has a prime factor larger than some threshold \( k \) with zero error and at least \( \frac{1}{2} \) probability of success with quantum algorithms.) However, there does not yet appear to be any problem known to be in \( \text{EQP} \) which is thought to lie outside of \( \text{P} \). \( 2 \) One might suspect that the restrictions imposed on \( \text{EQP} \) (to problems that can be decided exactly, using circuit families \( \{ C_n \}_{n \geq 1} \) composed of unitary gates, drawn from a single finite basis) may be too restrictive to allow an interesting theory of non-classical algorithms. If we suppose that unitarity is necessary for a physically-motivated model of computation, but we are still interested in principle in which problems could be decided exactly, we may consider families of circuits which are not constructed from a single finite basis. We may consider circuits with “potentially infinite” gate-sets, using the framework of Ref. [9, §3.3 A]. \( 3 \)

**Definition V.** Unitary\( \text{P}_C \) is the class of problems for which there is a circuit family \( \{ C_n \}_{n \geq 1} \) with unitary gates and preparation of fresh qubits in the state \(|0\rangle\), which

- is polynomial-time uniform: the structure of the circuit \( C_n \) and the labels of its gates can be computed in time \( O(\text{poly } n) \);
- has polynomial-time specifiable gates: the coefficients in \( \mathbb{C} \) of each gate in \( C_n \), can be computed from the label of the gate in time \( O(\text{poly } n) \);
- decides \( L \) exactly: for any input \( x \) of size \( n \), \( C_n |x\rangle = |\psi(x)\rangle |0\rangle \) for \( \text{no} \) instances, and \( C_n |x\rangle = |\psi(x)\rangle |1\rangle \) for \( \text{yes} \) instances, where \( |\psi(x)\rangle \) is a pure state.

Clearly \( \text{EQP} \subseteq \text{Unitary}\text{P}_C \) (by restricting to circuit-families with “constant-time specifiable” gates); one may also show that \( \text{Unitary}\text{P}_C \subseteq \text{BQP} \) (see Ref. [9, §3.3 C]). A simple modification of Ref. [4, Theorem 6.2] will show that the gates of the circuits of \( \text{Unitary}\text{P}_C \) algorithms may be restricted to algebraic numbers. As circuits with polynome-specifiable gates are considerably more flexible than those with constant-time specifiable gates — for example, the former include quantum Fourier transforms of arbitrary order, while the latter does not — we conjecture that \( \text{EQP} \) is strictly contained in \( \text{Unitary}\text{P}_C \).

---

1. The closest result known to the author is a circuit family for the discrete logarithm problem due to Mosca and Zalka [29]: however, this result involves the preparation of input-dependent quantum superpositions, which cannot be realised in any obvious way using gates from a fixed finite set acting on standard basis states.

2. Similar families of circuits are those described simply as “uniform” by Nishimura and Ozawa [31]; however, while they suppose that coefficients are specified by rational approximations, we require an exact representation for which there exist efficient algorithms for arithmetic operations.

---

6
We consider this evidence that allowing “potentially infinite”, but efficiently specifiable, of some other qubit being in the state PostEQP the output. The class UnitaryP defines non-unitary transformations of state vectors, following Aaronson [1]. Consider the following “quasi-quantum” models of computation, in which we allow gate-sets for quantum algorithms is not computationally extravagant. (We present a foundational argument for the study of circuit families in the Appendix.)

\textbf{Proposition 4.} UnitaryP \subseteq \text{LWPP.}

We consider this evidence that allowing “potentially infinite”, but efficiently specifiable, gate-sets for quantum algorithms is not computationally extravagant. (We present a foundational argument for the study of circuit families in the Appendix.)

\section{2.3 Quasi-quantum complexity}

Consider the following “quasi-quantum” models of computation, in which we allow non-unitary transformations of state vectors, following Aaronson [1].

\textbf{Definition VI.} Define the following classes in analogy to EQP, UnitaryP, and ZQP:

- **EQP_{GL}** is the set of problems which may be decided by polytime-uniform circuit families \( \{ C_n \} \), over a finite set of invertible gates, such that \( C_n |x\rangle = |\psi(x)\rangle |0\rangle \) if \( x \) is a no instance and \( C_n |x\rangle = |\psi(x)\rangle |1\rangle \) if \( x \) is a yes instance.

- **GLP_{C}** is the set of problems which may be decided by polytime-uniform circuit families \( \{ C_n \} \), with polytime-specifiable invertible gates, such that \( C_n |x\rangle = |\psi(x)\rangle |0\rangle \) if \( x \) is a no instance and \( C_n |x\rangle = |\psi(x)\rangle |1\rangle \) if \( x \) is a yes instance.

- **ZQP_{GL}** is the set of problems which may be decided with zero error by polytime-uniform circuit families \( \{ C_n \} \), over a finite set of invertible gates, and two sets of standard-basis projectors \( \{ \Pi_F, \Pi_S \} \) and \( \{ \Pi_0, \Pi_1 \} \) on distinct qubits, and with constant success probability in that for the normalised state-vector \( |\Psi(x)\rangle = C_n |x\rangle / \sqrt{\langle x|C_n^n |x\rangle} \),
  - \( \langle \Psi(x)|\Pi_S |\Psi(x)\rangle \geq \frac{1}{2} \) for both yes and no instances \( x \);
  - \( \langle \Psi(x)|\Pi_0\Pi_S |\Psi(x)\rangle = 0 \) for yes instances;
  - \( \langle \Psi(x)|\Pi_1\Pi_S |\Psi(x)\rangle = 0 \) for no instances.

- **PostEQP** is the set of problems which may be decided by polytime-uniform circuit families \( \{ C_n \} \), over a finite set of unitary gates, and two sets of standard-basis projectors \( \{ \Pi_F, \Pi_S \} \) and \( \{ \Pi_0, \Pi_1 \} \) on distinct qubits, and exactly with postselection in the sense that for \( |\Psi(x)\rangle = C_n |x\rangle \),
  - \( \langle \Psi(x)|\Pi_S |\Psi(x)\rangle > 0 \) for both yes and no instances \( x \);
  - \( \langle \Psi(x)|\Pi_0\Pi_S |\Psi(x)\rangle = 0 \) for yes instances;
  - \( \langle \Psi(x)|\Pi_1\Pi_S |\Psi(x)\rangle = 0 \) for no instances.

The classes EQP_{GL}, GLP_{C}, and ZQP_{GL} are invertible-gate analogues of EQP, UnitaryP_{C}, and ZQP respectively, in which we renormalise the state before producing the output. The class PostEQP is a variant of EQP, in which we post-select on the value of some other qubit being in the state \( |1\rangle \) (ignoring the output in all other branches).
prior to producing the output, and renormalise the state conditioned on the postselected outcome. For those classes which decide languages exactly, the renormalisation has no effect on the decomposition of the result as a tensor product of the answer and the remaining qubits, and so is omitted from the definitions of those classes.

**Remark.** The notations for the classes above are slightly non-uniform. We use the notation GLP\(_C\) (as well as the notation UnitaryP\(_C\)) as part of the framework for quasi-probabilistic (or “modal”) computational models defined in Ref. [9]. The “GL” refers to the fact that the gates are elements of GL\(_k\)(C) for various values of \(k\). We use the notations EQP\(_{GL}\) and ZQP\(_{GL}\) as a compromise between this convention and the notation EQP\(_{nu}\) and ZQP\(_{nu}\) which would extend the notation suggested by Aaronson [1] for bounded-error “quantum” algorithms using invertible non-unitary gates. One could analogously consider (non-unitary) quasi-quantum “algorithms”, consisting of circuit families in which the polytime-specifiable gates are affine operators over \(\mathbb{C}\), i.e. which conserve the sum of the coefficients of the distributions on which they act (treating them as quasi-probability distributions). One might then denote the corresponding complexity classes by Affine\(_C\), EQP\(_{Aff}\), and ZQP\(_{Aff}\). (We do not study the latter classes here, but note their existence to justify the notation used for EQP\(_{GL}\) and ZQP\(_{GL}\).)

As the proofs of Propositions 2 and 4 do not depend in any way on the transformations involved being unitary, the following proposition is implicit in Refs. [4, 24]:

**Proposition 5.** EQP\(_{GL}\) \(\subseteq\) LPWPP and GLP\(_C\) \(\subseteq\) LWPP.

That is, allowing non-unitary operations does not allow such exact “quasi-quantum” algorithms to exceed the known upper bounds on exact quantum complexity. Similarly, the results of Fenner et al. [23] that NQP = coC\(_C\)P does not depend on the gates being unitary; we then have

**Proposition 6.** ZQP\(_{GL}\) \(\subseteq\) \(\Delta\)C\(_C\)P.

Finally, given a PostEQP algorithm for a problem, we may implement an NQP and a coNQP algorithm for the same problem along the lines described in the proof of Proposition 3, so that we have:

**Proposition 7.** PostEQP \(\subseteq\) \(\Delta\)C\(_C\)P.

The main results of this article are to show that the containments of the above three propositions all hold with equality.

### 3 Technical definitions in exact counting complexity

We now define some concepts relating to algorithms for exact gap-definable classes. This will simplify the analysis of simulations of these algorithms by quasi-quantum algorithms, of the sort described as part of Definition VI.

#### 3.1 Dual nondeterministic machines

It will prove helpful to describe algorithms in terms of *dual nondeterministic Turing machines*: a pair of normal-form NTMs \((N_0, N_1)\) for a language \(L\), such that \(N_0\) represents a C\(_C\)P algorithm for \(L\) (whose gap-function \(g_0\) evaluates to zero for \(x\in L\)) and \(N_1\) represents a coC\(_C\)P algorithm for \(L\) (whose gap-function \(g_1\) evaluates to zero for \(x\notin L\)). We further require that for any input \(x\in\{0,1\}^*\), pairs of dual NTMs make the same number of nondeterministic choices as one another.
Consider a poly-time NTM $\text{LPWPP}$ suppose that $f$ be by $f$ let $N$ in normal form, rather referring to than the NTMs themselves. In particular, if $N$ is odd. The resulting gap-function $g_j$ forms the non-deterministic transitions corresponding to the final $|q_1(|x|) - q_0(|x|)|$ bits of $b$, recording those bits as it does so. If those bits are of the form $1^*(0|1)$, then $N'_j$ accepts if and only if $N_j$ accepts; otherwise $N'_j$ accepts if the parity of the substring is odd. The resulting gap-function $g'_j$ then satisfies $g'_j(x) = 2g_j(x)$ for all $x$, and the machines $N'_j$ and $N_{(1-j)}$ perform the same number of nondeterministic transitions.

It will also be useful to consider dual LWPP machines (and dual LPWPP machines): a pair $(N_0, N_1)$ of dual machines for some $L \in \text{LWPP}$ (or $L \in \text{LPWPP}$ respectively), for which there is a single poly-time computable, length-dependent, non-zero function $h$ (whose values are all powers of a fixed $M \geq 1$ for $L \in \text{LPWPP}$) such that either $\frac{1}{2}g_0(x) = h(x)$ or $\frac{1}{2}g_1(x) = h(x)$.

**Lemma 8.** Every $L \in \text{C.} \cap \text{coC.} \cap \text{P}$ has a pair of dual NTMs.

**Proof.** Let $N_0$ be an NTM performing a C. algorithm for $L$ with $q_0(|x|)$ nondeterministic transitions in each branch, and $N_1$ be an NTM performing a coC. algorithm for $L$ with $q_1(|x|)$ nondeterministic transitions in each branch. We augment whichever machine $N_j$ makes fewer transitions, as follows. Construct an NTM $N'_j$ which has branching strings $b \in \{0,1\}^{\max\{q_0(|x|),q_1(|x|)\}}$, which first simulates $N_j$, and then performs the non-deterministic transitions corresponding to the final $|q_1(|x|) - q_0(|x|)|$ bits of $b$, recording those bits as it does so. If those bits are of the form $1^*(0|1)$, then $N'_j$ accepts if and only if $N_j$ accepts; otherwise $N'_j$ accepts if the parity of the substring is odd. The resulting gap-function $g'_j$ then satisfies $g'_j(x) = 2g_j(x)$ for all $x$, and the machines $N'_j$ and $N_{(1-j)}$ perform the same number of nondeterministic transitions.

It will also be useful to consider dual LWPP machines (and dual LPWPP machines): a pair $(N_0, N_1)$ of dual machines for some $L \in \text{LWPP}$ (or $L \in \text{LPWPP}$ respectively), for which there is a single poly-time computable, length-dependent, non-zero function $h$ (whose values are all powers of a fixed $M \geq 1$ for $L \in \text{LPWPP}$) such that either $\frac{1}{2}g_0(x) = h(x)$ or $\frac{1}{2}g_1(x) = h(x)$.

**Lemma 9.** Every $L \in \text{LWPP}$ has a pair of dual LWPP machines, and every $L \in \text{LPWPP}$ has a pair of dual LPWPP machines.

**Proof.** Consider a poly-time NTM $N$ in normal form with branching strings $b \in \{0,1\}^{|q(|x|)|}$, representing an LWPP algorithm to decide a language $L$, in that the gap-function $g$ of $N$ satisfies $g(x) = 0$ for $x \notin L$ and $g(x) = 2h(x)$ for $x \in L$, where $h$ is non-zero, efficiently computable, and length-dependent. We may form a dual pair of NTMs $(N_0, N_1)$, as follows. We construct $N_0$ to make a non-deterministic transition corresponding to a bit $\beta$. If $\beta = 0$, it non-deterministically guesses an integer $0 \leq b' < 2^{q(|x|)}$, and rejects if $b' < 2^{q(|x|)} - h(x)$, accepting otherwise. Otherwise, for $\beta = 1$, $N_0$ simulates $N$, and accepts if and only if $N$ rejects. If $g$ is the gap function for $N$, the gap function for $N_0$ is then $-g(x) + 2h(x)$, which is equal to $2h(x)$ if $x \notin L$ and $0$ otherwise. We similarly construct $N_1$ to make a non-deterministic transition corresponding to a bit $\beta$: if $\beta = 0$ it accepts on exactly half of the branching strings, and if $\beta = 1$ it accepts if and only if a simulation of $N$ accepts. Then $(N_0, N_1)$ are dual LWPP machines, with branching strings in $\{0,1\}^{|q(|x|)|+1}$, and gap functions governed by the same poly-time computable function $h$ which governs $N$. In particular, if $L \in \text{LPWPP}$, we may require that $h$ computes powers of some fixed integer $M \geq 1$, in which case $(N_0, N_1)$ are dual LPWPP machines.

### 3.2 Verifier functions and gap amplitudes

It will prove more convenient in our analysis to refer to polytime-computable functions $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$, which compute the acceptance conditions of NTMs in normal form, rather referring to than the NTMs themselves. In particular, if $N$ is a poly-time NTM in normal form which halts in $q(n)$ steps for inputs of length $n \in \mathbb{N}$, we let $f(x,b) \in \{0,1\}$ represent the acceptance condition of $N$ in the branch represented by $b$ on the input $x$. (For a boolean string $b$ which is longer or shorter than $q(|x|)$, we suppose that $f(x,b) = 0$.) We then define:

$$A(x,f,q) = \#\left\{ b \in \{0,1\}^{q(|x|)} \mid f(x,b) = 1 \right\}, \quad (1a)$$

$$R(x,f,q) = \#\left\{ b \in \{0,1\}^{q(|x|)} \mid f(x,b) = 0 \right\}, \quad (1b)$$
After the second round of Hadamards on the branching register (not counting the which allows one to neglect unbounded error. We show in this section, for zero-error which compute the respective acceptance conditions of a pair of dual NTMs where \( f \) computes the acceptance condition of one of a pair of dual NTMs. We will typically consider “dual pairs” of verifier functions: boolean functions

\[
V_n^{(0)} : \{0, 1\}^n \times \{0, 1\}^q(n) \to \{0, 1\}, \quad V_n^{(1)} : \{0, 1\}^n \times \{0, 1\}^q(n) \to \{0, 1\}
\]

(2) which compute the respective acceptance conditions of a pair of dual NTMs \((N_0, N_1)\) on inputs \(x \in \{0, 1\}^n\). When the pair of dual NTMs are defined by context (with \(q\) implicitly depending on them), we then write

\[
\alpha_x^{(c)} = \frac{A(x, V^{(c)}, q)}{2^{q(|x|)}}, \quad \rho_x^{(c)} = \frac{R(x, V^{(c)}, q)}{2^{q(|x|)}}, \quad \delta_x^{(c)} = \frac{\Delta(x, V^{(c)}, q)}{2^{q(|x|)}}
\]

(3)

for the sake of brevity. (Note that \(\rho_x^{(c)} + \alpha_x^{(c)} = 1\) by construction.)

We refer to \(\alpha_x^{(0)}, \alpha_x^{(1)}\) as the gap amplitudes of the NTMs \(N_0\) and \(N_1\); our simulation techniques largely concern evaluating such gap amplitudes.

4 Quasi-quantum algorithms for \(\Delta C_P\)

The definitions of the classes \(\text{ZQP}_{\text{GL}}\) and \(\text{PostEQP}\) in Definition VI are quite similar: the notional “exactness” of \(\text{PostEQP}\) belies the fact that it involves a projective operation, which allows one to neglect unbounded error. We show in this section, for zero-error quantum-like algorithms, how these two varieties of non-unitarity are equivalent.

4.1 A unitary circuit to construct gap amplitudes

Consider a dual pair of verifier functions \(V_n^{(0)}\) and \(V_n^{(1)}\) as in Section 3.2, corresponding to dual NTMs \((N_0, N_1)\) acting on inputs of length \(n\), and let \(m = q(n)\) be the size of the branching strings accepted by these verifiers. Figure 2 presents a unitary circuit \(U_n\) to produce the gap amplitudes of \((N_0, N_1)\) as amplitudes of a pure state. Consider the evolution of the qubits in the circuit when presented with an input of \(|x\rangle |0^{m+2}\rangle\). After the first round of Hadamard operations, followed by coherently evaluating both verifier circuits, the state of the system may be expressed by

\[
|\Psi_1(x)\rangle = \frac{1}{\sqrt{2^{m+1}}} \sum_{b \in \{0, 1\}^m} \sum_{c \in \{0, 1\}} |x\rangle |b\rangle |c\rangle |V^{(c)}(x, b)\rangle.
\]

(4)

After the second round of Hadamards on the branching register (not counting the Hadamard on the final bit), we obtain the state

\[
|\Psi_2(x)\rangle = \frac{1}{2^m \sqrt{2}} \sum_{b \in \{0, 1\}^m} \sum_{c \in \{0, 1\}} (-1)^x \cdot b |x\rangle |z\rangle |c\rangle |V^{(c)}(x, b)\rangle
\]

\[
= |\Psi_d(x)\rangle + \frac{1}{2^m \sqrt{2}} \sum_{b \in \{0, 1\}^m} |x\rangle |0^m\rangle |c\rangle |V^{(c)}(x, b)\rangle,
\]

(5a)
To produce the gap amplitude in the amplitudes of the components, we perform the Hadamard operation on the final bit, yielding 

\[ V_n(0) \] and \[ V_n(1) \]

In particular, we have

\[ \langle \psi \rangle \]

and the final bit is \[ |1⟩ \]. The component in which the "branching register" is \[ |0⟩ \] and the final bit is \[ |1⟩ \] then has an amplitude proportional to a gap function.

where \[ |Ψ'_x⟩ \] is simply the contributions to \[ |Ψ_2⟨⟩ \] for \[ z \neq 0^n \]:

\[
|Ψ'_x⟩ = \frac{1}{2^m} \sum_{c \in \{0,1\}} \sum_{x,b \in \{0,1\}^n} (−1)^{c,b} |x⟩ |z⟩ |c⟩ |V(c)(x,b)⟩ .
\] \hfill (5b)

In particular, we have \((|0⟩ ⊗ |0⟩) ⊗ |Ψ'_x⟩ = 0\). Evaluating the sum over \(b \in \{0,1\}^m\) in the final term of Eqn. (5a), we may then express

\[
|Ψ_2(x)⟩ := |Ψ'_x⟩ + \frac{1}{\sqrt{2}} |x⟩ |0^m⟩ \sum_{c \in \{0,1\}} \left( R(x,V(c),m) |c0⟩ + \frac{A(x,V(c),m)}{2^m} |c1⟩ \right)
\]

\[
= |Ψ'_x⟩ + \frac{1}{\sqrt{2}} |x⟩ |0^m⟩ \sum_{c \in \{0,1\}} \left( \rho_x^c |c0⟩ + \alpha_x^c |c1⟩ \right).\]
\hfill (6)

To produce the gap amplitude in the amplitudes of the components, we perform the Hadamard operation on the final bit, yielding

\[
|ψ'_x⟩ := |Ψ_3(x)⟩ = |x⟩ |0^m⟩ \sum_{c \in \{0,1\}} \left( \left[ \frac{\rho_x^c + \alpha_x^c}{2} \right] |c0⟩ + \left[ \frac{\rho_x^c - \alpha_x^c}{2} \right] |c1⟩ \right)
\]

\[
+ \left( |1⟩ ⊗ |0⟩^n+1 ⊗ H \right) |Ψ'_x⟩
\]

\[
= |x⟩ |0^m⟩ \left( \frac{1}{2} |00⟩ + \frac{1}{2} |10⟩ + \delta_x^{(0)} |01⟩ + \delta_x^{(1)} |11⟩ \right) + |ψ'_x⟩,\]
\hfill (7)
where $|\psi'_x\rangle = (1^{\otimes n+m+1} \otimes H) |\Psi'_2(x)\rangle$, and satisfies $(1^{\otimes n} \otimes (0^n \otimes 1^{\otimes 2}) |\psi'_x\rangle = 0$. Thus, in the component of $|\psi_x\rangle = U_n |x\rangle |0^{n+2}\rangle$ where the branching register is in the state $|0^{n}\rangle$ and the final bit is in the state $|1\rangle$, the amplitudes are given by the gap amplitudes of the machines $N_0$ and $N_1$. Furthermore, by the fact that $(N_0, N_1)$ are dual, exactly one of $\delta_x^{(0)}$, $\delta_x^{(1)}$ is non-zero. Specifically, $\delta_x^{(0)} = 0$ if $x \in L$, and $\delta_x^{(1)} = 0$ if $x \notin L$. If we use the notation $L(x) = 0$ to indicate $x \notin L$ and $L(x) = 1$ to indicate $x \in L$, we then have $\delta_x^{(L(x))} \neq 0$, and we may express

$$|\psi_x\rangle := |x\rangle |0^n\rangle \left( \frac{1}{2} |00\rangle + \frac{1}{2} |10\rangle + \delta_x^{(L(x))} |L(x)\rangle |1\rangle \right) + |\psi'_x\rangle .$$

(8)

Note that $\langle \psi_x | \psi_x \rangle = 1$, as $U_n$ is unitary, and that measuring $|\psi_x\rangle$ in the standard basis yields $|x\rangle |0^n\rangle |00\rangle$ or $|x\rangle |0^n\rangle |10\rangle$ with probability $\frac{1}{2}$ each. It then follows that $\langle \psi'_x | \psi'_x \rangle < \frac{1}{2}$.

### 4.2 Quasi-quantum algorithms by amplitude forcing

The above construction produces a state in which the gap amplitudes indicate (albeit possibly with low probability) whether $x \in L$ for an input $x$. This will allow us to easily prove that $L \in ZQP_{GL}$ and $L \in PostEQP$, and that therefore $\Delta C_P = ZQP_{GL} = PostEQP$.

The second-to-last bit of $|\psi_x\rangle$ stores $L(x)$ in the component where the branching register takes the state $|0^n\rangle$ and the final bit is in the state $|1\rangle$. By using a multiply-controlled NOT conditioned on these states, we may indicate in a single bit whether...
or not \(L(x)\) has been successfully computed. Consider the circuit in Figure 3, which computes such a bit, performs a non-unitary single-bit operation \(N = \text{diag}(p, 1)\) for some \(0 \leq p < 1\), and performs a permutation so that the final bit presents (with some probability) the outcome \(L(x)\). Given an input of \(|x\rangle|0^{m+3}\rangle\) the output of this circuit is

\[
|\Psi_{L,x}\rangle = |x\rangle|0^m\rangle \left( \frac{1}{2} |000\rangle + \frac{p}{2} |001\rangle + \delta_x^{(L(x))} |11\rangle |L(x)\rangle \right) + p|\psi_x''\rangle, \tag{9}
\]

where \(|\psi_x''\rangle\) is the result of cyclically permuting the final three qubits of \(|\psi_x'\rangle|0\rangle\). (In particular, the second-to-last qubit is in the state \(|0\rangle\).) We may then describe \(\text{ZQP}_{GL}\) and \(\text{PostEQP}\) algorithms for \(L\), as follows.

### 4.2.1 Zero-error algorithms by invertible gap amplification

To consider a zero-error algorithm, consider the projectors \(\Pi_F = |0\rangle\langle 0|\) and \(\Pi_S = |1\rangle\langle 1|\) performed on the second-to-last qubit, and consider the effect of a projective \(\{\Pi_F, \Pi_S\}\) measurement. We interpret an outcome of \(\Pi_F\) as a failure of the algorithm to produce an answer; on an outcome of \(\Pi_S\) we produce the value of the final bit as output of the algorithm. By construction, when we obtain the outcome \(\Pi_S\), the value of the final bit will be \(|L(x)\rangle\). This is then a zero-error algorithm for any value of \(p\). To ensure that this zero-error algorithm has a bounded probability of failure, we need to bound the probability of obtaining the outcome \(\Pi_F\) from above.

To determine the probability of any particular outcome from \(|\Psi_{L,x}\rangle\), we renormalise the vector and then apply the Born rule. Thus the probability of any measurement outcome is determined by the ratio of the modulus-squared of its amplitude, relative to that of all other amplitudes. We have \(\langle \psi''_x | \psi''_x \rangle < \frac{1}{2}\) from the remarks at the end of Section 4.1; thus

\[
\left\| \Pi_F |\Psi_{L,x}\rangle \right\|^2 < \frac{p^2}{4} + \frac{p^2}{4} + \frac{p^2}{2} = p^2, \tag{10}
\]

whereas \(\left\| \Pi_S |\Psi_{L,x}\rangle \right\|^2 = \delta_x^{(L(x))} 2\). Thus we have

\[
\left\| \Pi_F |\Psi_{L,x}\rangle \right\|^2 / \left\| \Pi_S |\Psi_{L,x}\rangle \right\|^2 < \left( p / \delta_x^{(L(x))} \right)^2. \tag{11}
\]

If we bound the right-hand side from above by 1, the probability of failure will then be less than \(\frac{1}{2}\). Given that \(\delta_x^{(L(x))} \geq 2^{-m}\), we may let \(p \leq 2^{-m}\), which suffices to ensure \(p \leq \delta_x^{(L(x))}\). If we let \(N = B^m\), where

\[
B = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix}, \tag{12}
\]

then \(p = 2^{-m}\), so that \(\Pi_F\) occurs with probability less than \(\frac{1}{2}\). Thus we have a zero-error polytime algorithm (using \(2m \in O(\text{poly} \ n)\) non-unitary invertible gates) with a bounded probability of failure, so that \(L \in \text{ZQP}_{GL}\). By Proposition 6, we then have:

**Theorem 10.** \(\text{ZQP}_{GL} = \Delta \text{C}_{\text{P}}\).

### 4.2.2 Forcing exactness by post-selection

The above zero-error algorithm suppresses the probability of failure by a scalar factor \(p \ll 1\). Taking this idea to its logical limit (i.e. setting \(N = B^t\) and letting \(t \to \infty\))
is equivalent to postselecting the value $|1\rangle$ on the second-to-last qubit of $|\Psi_{L,x}\rangle$. In other words, taking $N = |1\rangle\langle 1|$ yields an algorithm with a well-defined final state after renormalisation, and in which the final bit is certain to be in the state $|L(x)\rangle$. This yields an exact polynomial-time (but postselective) algorithm for $L$, so that $L \in \text{PostEQP}$. Together with Proposition 7, we then have:

**Theorem 11.** PostEQP = $\Delta_C \cdot P$.

## 5 Quasi-quantum algorithms for LWPP and LPWPP

Postselected quantum computations can achieve exactness, in effect, by the ignoring the probability of failure of a zero-error algorithm. Achieving exact quantum algorithms with invertible operations is subtler, as we show in this section. The set of problems which exact nonunitary algorithms can solve is apparently smaller, even if one considers algorithms with a possibly infinite (but polynomial-time specifiable) gate-set.

### 5.1 An invertible circuit to construct gap amplitudes

We first define a circuit $W_n$ as in Figure 4, which builds upon the unitary circuit $U_n$ described in Figure 2 and uses a single non-unitary gate

$$S = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{13}$$

This circuit constructs the gap amplitude $\delta_z^{(L(x))}$ for a component of its output in much the same way that $U_n$ does, and also uses $U_n^\dagger$ to uncompute the “garbage” components contained in the state $|\Psi_{L,x}\rangle$ described in Eqn. (9). This will simplify the analysis to follow for exact invertible algorithms.

As before, consider a dual pair of verifier functions $V_n^{(0)}$ and $V_n^{(1)}$, corresponding to dual NTMs $(N_0, N_1)$ acting on inputs of length $n$, and let $m = q(n)$ be the size of the branching strings accepted by these verifiers. Let $U_n$ be as described in Figure 2, defined in terms of functions $V_n^{(0)}$ and $V_n^{(1)}$. On an input of $|x\rangle |0\rangle |0\rangle$, the circuit first introduces several ancilla bits to produce $|x\rangle |0^n, y\rangle$. Acting on this with the operator $U_n \otimes 1$ then yields a state

$$|\Phi_1(x)\rangle = |\psi_x\rangle |0\rangle = |x\rangle |0^n\rangle \left( \frac{1}{2} |00\rangle + \frac{1}{2} |10\rangle + \delta_z^{(L(x))} |L(x)\rangle |1\rangle \right) |0\rangle + |\psi'_x\rangle |0\rangle, \tag{14}$$

following Eqn. (8). In particular, we have $(1 \otimes^n \otimes |0\rangle \otimes 1^{\otimes 2}) |\psi'_x\rangle = 0$. Then after the multiply controlled-NOT gate, we obtain the state

$$|\Phi_2(x)\rangle = |x\rangle |0^n\rangle \left( \frac{1}{2} |00\rangle + \frac{1}{2} |10\rangle \right) |0\rangle + |\psi'_x\rangle |0\rangle + \delta_z^{(L(x))} |x\rangle |0^n\rangle |L(x)\rangle |1\rangle |1\rangle. \tag{15}$$

The $S$ gate linearly (but non-unitarily) maps $|0\rangle \mapsto |0\rangle$ and $|1\rangle \mapsto |1\rangle$; thus the state after the $S$ operation on the final bit is

$$|\Phi_3(x)\rangle = |x\rangle |0^n\rangle \left( \frac{1}{2} |00\rangle + \frac{1}{2} |10\rangle + \delta_z^{(0)} |01\rangle + \delta_z^{(1)} |11\rangle \right) |0\rangle + |\psi'_x\rangle |0\rangle + \delta_z^{(L(x))} |x\rangle |0^n\rangle |L(x)\rangle |1\rangle |1\rangle,$$

$$= |\psi_x\rangle |0\rangle + \delta_z^{(L(x))} |x\rangle |0^n\rangle |L(x)\rangle |1\rangle |1\rangle. \tag{16}$$
Figure 4: Diagram for a circuit $W_n$, consisting of unitary gates, together with one non-unitary operation $S$, to evaluate gap functions corresponding to two distinct poly-time nondeterministic Turing machines $N_0$ and $N_1$. The circuit $U_n$, and its relationship to the two nondeterministic machines $N_0$ and $N_1$, are presented in Figure 2. The $S$ gate is a non-unitary transformation given by $S = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$. Operations with controls marked by white circles are (coherently) conditioned on the qubit being in the state $|0\rangle$; operations with controls marked by black circles are conditioned on the qubit being in the state $|1\rangle$. The qubits with explicitly indicated input and output states are ancillas.

Conditioned on the final bit being $|0\rangle$, we coherently perform $U_n^\dagger$, and flip the second-to-last qubit conditioned on the final bit being $|1\rangle$. This produces the state

$$|\Phi_4(x)\rangle = |x\rangle |0^m\rangle |000\rangle + \delta_x^{(L(x))} |x\rangle |0^m\rangle |L(x)\rangle |0\rangle |1\rangle, \quad (17)$$

removing the ancilla qubits, this yields a final state of

$$|\varphi_x\rangle = |x\rangle \left( |00\rangle + \delta_x^{(L(x))} |L(x)\rangle |1\rangle \right). \quad (18)$$

A single non-unitary operation thus suffices to uncompute all components of the output, except for the $|00\rangle$ component and a component which indicates whether $x \in L$.

## 5.2 Exact invertible algorithms

If $(N_0, N_1)$ are dual LWPP machines, there is a poly-time computable and length-dependent function $h$, for which $\delta_x^{(L(x))} = \frac{1}{2^m} h(x)$. The ability to compute $h(x)$ exactly and in polynomial time allows us to suppress the amplitude of the $|00\rangle$ component precisely, by interfering it with a (non-unitary) contribution from the component which indicates $L(x)$. This allows us to easily prove that $L \in \text{GLP}_\mathbb{C}$, and that furthermore $L \in \text{EQP}_{\text{GL}}$ in the case that $L \in \text{LPWPP}$.

### 5.2.1 Polytime-specifiable invertible circuits for $L \in \text{LWPP}$

Figure 5 presents a polytime uniform (and polytime-specifiable) invertible circuit to exactly decide a language $L \in \text{LWPP}$, in the case that $(N_0, N_1)$ are dual LWPP machines. In addition to the subroutine $W_n$ presented in Figure 4, this circuit uses one
Figure 5: Diagram for a circuit consisting of invertible gates to exactly decide a language \( L \in \text{LWPP} \). The invertible circuit \( W_n \) is as depicted in Figure 4, and depends on nondeterministic Turing machines representing \( \text{LWPP} \) algorithms for \( L \) and \( \overline{L} \). The gate \( B \) is the same as in Eqn. (12), and \( D \) is given by Eqn. (20). The gate \( A_n \) acts on a single bit, and is polytime-specifiable from the input size \( n \): in case we have \( L \in \text{LPWPP} \), we have \( A_n = G^{O(\text{poly } n)} \) for some single-qubit gate \( G \) which is constant in the input size. On input \(|x\rangle|0\rangle|0\rangle\), the output of this circuit is proportional to \(|x\rangle|1\rangle L(x)\rangle\).

By the choice of \( h \), this is a family of invertible single-bit operations whose coefficients are computable in time \( O(\text{poly } n) \); it is therefore poly-time specifiable in the sense of Definition V. The algorithm also involves the two-bit invertible gate

\[
D = \begin{bmatrix}
1 & 0 & -1 & -1 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix},
\] 

(20)

which linearly maps \(|0x\rangle \mapsto |0x\rangle \) and \(|1x\rangle \mapsto |1x\rangle - |00\rangle\); and the invertible operator \( B = \text{diag}(\frac{1}{2}, 1) \) described in Eqn. (12).

Let \(|\varphi'_x\rangle\) represent the result of swapping the final two bits of the state \(|\varphi_x\rangle\) given in Eqn. (18). The effect of performing the circuit of Figure 5 on the input state \(|x\rangle|0\rangle|0\rangle\) is then to produce the state

\[
|\Phi_{L,x}\rangle = \left(\mathbb{1} \otimes^n [D(A_n B^m \otimes \mathbb{1})]\right) |\varphi'_x\rangle \\
= |x\rangle \otimes \left[D(A_n B^m \otimes \mathbb{1})(|0\rangle |0\rangle + \delta_x^{L(x)} |1\rangle |L(x)\rangle)\right].
\] 

(21)

The operator \( A_n B^m \otimes \mathbb{1} \) multiplies the amplitude of \(|0\rangle |0\rangle\) by \( h(1^n)/2^m = h(x)/2^m \), leaving the component \(|1\rangle |L(x)\rangle\) unaffected. By construction, we have \( \delta_x^{L(x)} = h(x)/2^m \), so it follows that

\[
|\Phi_{L,x}\rangle = |x\rangle \otimes \frac{h(x)}{2^m} \left[D\left(|0\rangle |0\rangle + |1\rangle |L(x)\rangle\right)\right] = \frac{h(x)}{2^m} |x\rangle |1\rangle |L(x)\rangle.
\] 

(22)

Thus the value of the final bit of the output is \(|L(x)\rangle\), with certainty. With Proposition 5, we then have:

**Theorem 12.** \( \text{LWPP} = \text{GLP}_C \).
5.2.2 Circuits for \( L \in \text{LPWPP} \) with finite invertible gate-sets

In the case that \( L \in \text{LPWPP} \), the function \( h(x) \) for which \( \delta^{(L(x))} = h(x)/2^m \) may be taken to be a power of some constant, \( h(x) = M^t(x) \) for some \( M \geq 1 \) and poly-time computable \( t(x) \in O(\text{poly} |x|) \). From this it follows that

\[
A_n = G^t(x), \quad \text{where } G = \text{diag}(M, 1). \tag{23}
\]

Thus the \( A_n \) gate in Figure 5 may be simulated by polynomially many \( G \) gates. The analysis of the preceding Section then suffices to show that \( L \in \text{LPWPP} \) may be exactly decided by a circuit over the gate-set \( \{ X, \text{CNOT}, \text{TOFFOLI}, H, S, B, G, D \} \). Together with Proposition 5, this proves:

**Theorem 13.** \( \text{LPWPP} = \text{EQP}_{\text{GL}} \).

6 Remarks

The results of this article are summarised in Figure 6. We have demonstrated that the classes \( \text{LWPP} \) and \( \Delta^C P \), which may seem to be of mainly technical interest in gap-definable complexity, can be defined quite naturally in terms similar to quantum circuit families. Specifically, they may be characterised in terms of acyclic directed networks of invertible tensors. The same perspective also motivates a gap-definable class \( \text{LPWPP} \) which provides a tighter upper bound for \( \text{EQP} \).

Using techniques similar to that of Theorem 12, one can describe a similar characterisation of the gap-definable class \( \text{WPP} \), in terms of a family of circuit with input-dependent gates in place of the length-dependent gate \( A_n \). These may be considered to represent a sort of input-dependent “just-in-time” circuit, in which an invertible-gate circuit which is computed depending on an input \( x \), and then immediately used as a subroutine to solve a decision or promise problem involving \( x \). Such a family of circuits would not satisfy any uniformity condition in the sense that “uniformity” is usually understood, and would represent a departure from the usual approach to computational complexity; nevertheless, this is a way in which \( \text{WPP} \) may also be characterised in terms of tensor networks. The problems solvable by unitary “just-in-time” circuit families of this sort would then be bounded by \( \text{WPP} \). (It seems likely that the algorithm of Mosca and Zalka [29] for \text{DISCRETE-LOG} is an example of such a just-in-time quantum algorithm: this would depend on a careful examination of how to exactly represent the state preparations involved through algebraic coefficients.)

Other gap-definable classes can also be described using tensor networks (albeit over fields other than \( \mathbb{C} \)), also representing models of indeterministic, “quasi-quantum” computation [9]. Perhaps all of counting complexity can be recast in terms of tensor-like structures over semirings, in a manner not unlike Valiant’s matchgates [36], but with the added intuition that the networks represent modes of indeterministic computation.

Our analysis suggests that the technical aspects of the definitions of \( \text{LWPP} \) and \( \text{WPP} \) (and analogously, \( \text{AWPP} \)) merely serve to capture indeterministic computations involving algebraic numbers, represented using natural numbers. Viewed in this light, it is not obvious that those definitions could be made simpler while still being presented in terms of gap-functions, rather than efficiently specified tensor networks. It also demonstrates that the existing upper bounds on quantum complexity classes make no use of the unitarity of the transformations involved. It is not obvious how a condition such as unitarity would be represented by gap-functions — except in the same technical manner as in the traditional definitions of \( \text{LWPP} \), \( \text{WPP} \), and \( \text{AWPP} \).
The question remains as to whether any of the bounds $\text{EQP} \subseteq \text{LPWPP}$, $\text{UnitaryP}_C \subseteq \text{LWPP}$, or $ZQP_c \subseteq C \cap \text{coC}_C$, are strict. However, we at least see that exact quantum-like classes can solve problems which are not expected to be in $P$; thus the constraint of exact decision does not in itself prevent a fruitful theory of indeterministic algorithms. Furthermore, as non-trivial techniques will be required to simulate unitary circuits with polynomial-time coefficients using circuits built from a finite gate-set, it seems likely that $\text{EQP} \neq \text{UnitaryP}_C$.

We ask whether $\text{SPP}$ has an interpretation as a quantum-like class in the manner of $\text{LWPP}$ and $\Delta C$, and whether $\text{SPP}$ is strictly contained in $\text{EQP}_{GL} = \text{LPWPP}$. A proof that $\text{SPP} = \text{LPWPP}$ could be taken as evidence that $\text{EQP} = P$; we consider either of these possible equalities to be worthy topics of investigation. Finally, we ask whether results similar to these also hold for other quasi-quantum models of computation, such as transformation by affine operators (which we allude to on page 8). A result implicit in Ref. [8] is that the problems which may be solved with bounded error by polynomial-time affine circuit families over a finite gate set is the entire class $\text{AWPP}$; we expect that the class of problems exactly solvable by such circuits is again $\text{LPWPP}$, and that extending to polytime-specifiable gate sets again recovers $\text{LWPP}$. 
Acknowledgements. This work was performed in part at the CWI, with support from a Vidi grant from the Netherlands Organisation for Scientific Research (NWO) and the European Commission project QALGO. A first draft of this work was completed while a guest of Jane Biggar, during a break from professional academia. I would like to acknowledge support from the UK Quantum Technology Hub project NQIT.

A On quantum algorithms with “infinite” gate sets

It might be that $\text{EQP}$ contains no interesting problems (or even no problems at all) outside of $\text{P}$. The results of this article show that exactness alone is not the cause. The other significant constraints on the study of exact quantum algorithms is the necessary restriction to unitary operations, and the restricting to a fixed finite gate set, both of which were originally motivated by the original definition of $\text{EQP}$ in terms of quantum Turing machines [13].

As we note on page 6 (and as observed by Nishimura and Ozawa [31]), a prime-factor version of the integer factoring problem is in $\text{ZQP}$, so it is not obvious that these constraints are too strict. However, in the zero-error case, the freedom to fail with constant probability allows us to exploit standard approximation results such as the Solovay–Kitaev Theorem [26, 19] to simulate quantum Fourier transforms (QFTs) of arbitrary order. Such freedom to fail is not available to exact quantum algorithms; nor can QFTs of arbitrary order (which involve algebraic number fields of arbitrarily large degree) be decomposed into a finite gate set, which can only yield coefficients from a fixed algebraic number field. In the case of integer factorisation, this is not the only obstacle for exact quantum algorithms. However, this does seem indicative of the central problem for exact quantum complexity: that quantum-versus-classical speedups require algorithms involving “large” amounts of destructive interference, and that arranging for the destructive interference to be total (in the physicist’s sense) is difficult to arrange with only a finite gate set.

The alternative is to allow infinite gate-sets — or perhaps more appropriately, to allow the basis $G_n$ available to a circuit $C_n$ in a uniform circuit family $\{C_n\}_{n \geq 1}$ not to be bounded in size by a constant. This is not a priori unreasonable from the standpoint of computational complexity: classical $\text{AC}^k$ circuits (with poly-log depth and gates with unbounded fan-in) also have length-dependent gate-sets — albeit ones which may be exactly decomposed by virtue of the existence of a finite universal gate sets for boolean logic. The absence of such finite universal gate sets for quantum computation need not prevent us from considering length-dependent gate-sets; indeed, one might suppose that this absence forces the issue of such gate-sets on us, as we cannot rule them out without loss of generality in the case of exact algorithms.

A consequence of considering such gate-sets is a break from the exact correspondence between quantum Turing machines and quantum circuit families. However, insisting on such a correspondence introduces a distinction in quantum complexity between “algorithms”, and “meta-algorithms” [9, Appendix C] consisting of efficiently computable descriptions of other computations. The standard theory of universal Turing machines prevents such a distinction in classical computational complexity; the absence of an exact (and efficient) universal quantum Turing machine is what makes an algorithm versus meta-algorithm distinction possible in the quantum regime. However, it is not clear that it is productive or meaningful to make such a distinction, especially as

---

4There do exist exact universal quantum Turing machines, if one is willing to dispense with efficient simulation. For example, any deterministic universal Turing machine suffices.

19
unitary circuit families and adiabatic quantum computations are both descriptions of computations whose parameters are computed by a Turing machine.

If one were to consider circuit families, which involve gate sets which scale with the circuit size, one must consider how to do so without trivialising the theory of quantum complexity, for either exact algorithms or for bounded-error algorithms. It seems plausible that restricting them to be polynomial-time specifiable, so that the entire tensor network of the each circuit \( C_n \) can be specified in time \( O(p\text{oly } n) \), is a reasonable first step. In order to avoid subsuming a non-trivial amount of work into any one gate — e.g. polylogarithmic factors, due to sophisticated unitary operators acting on \( O(\log n) \)-qubits — one may impose further restrictions. Given that we require each gate in the circuit to be explicitly expressible in polynomial time, it seems likely that standard exact circuit decomposition techniques \([30]\) would allow us to restrict to gates acting on \( O(1) \) qubits without loss of generality. Furthermore, to represent the cost of calibration of (abstract) devices which control the unitary evolution of the computational state, and also to account for the work performed by a straightforward evaluation of amplitudes through the gap-function of an NTM, it seems reasonable to impose a non-trivial cost on each gate, which depends on the time required to compute its description. (For a gate-set of finite size, this cost would be bounded by a constant, recovering the usual notion of circuit size as the cost of the circuit.) Ref. \([9, \S 3.3\ C]\) sketches how circuits involving such gates may be simulated with bounded error by constant-size gates; and Proposition 4 of this article sketches how the complexity of algorithms using such gate-sets can be bounded by counting classes which are well-known as upper bounds for exact quantum complexity. Thus, such an approach seems likely to leave the existing theory of bounded-error quantum computation largely unchanged, while providing more freedom in the study of exact quantum algorithms.

It seems plausible that there would be problems of interest in \( \text{UnitaryP}_C \) which are not expected to be in \( P \), or even \( \text{BPP} \). The usual approach to decomposing the QFT over the integers modulo \( 2^n \) taught in many introductory courses on quantum algorithms \([17]\), is an example of a uniform circuit family over a polytime-specifiable gate-set. It seems plausible that, by some modification of the approach of Mosca and Zalka \([29]\), some version of \textsc{Discrete-Log} might be contained in the class \( \text{UnitaryP}_C \) of problems which are exactly decidable by such unitary circuits (though this may involve technical arguments on the number of \( O(n) \)-bit primes for which \textsc{Discrete-Log} is expected to be hard). As we show in the main text, \text{UnitaryP}_C \text{ is bounded above both by LWPP (until recently the best known bound on EQP) and BQP, which we consider evidence that this approach to quantum computational complexity is not extravagant. We therefore propose the study of the class UnitaryP}_C \text{ (or some similar notion of exact quantum complexity) as a way to approach the subject of exact quantum algorithms.}

References

[1] S. Aaronson. \textit{Quantum computing, postselection, and probabilistic polynomial-time}. Proc. Roy. Soc. \textbf{A 461} (3473–3482), 2005. arXiv:quant-ph/0412187.

[2] S. Aaronson, A. Ambainis. \textit{The Need for Structure in Quantum Speedups}. Proc. ICS 2011 (pp. 338–352), 2011. arXiv:0911.0996.

[3] S. Aaronson and A. Archipov. \textit{The Computational Complexity of Linear Optics}. Proceedings of the 43rd Annual STOC (pp. 333–342), 2011. arXiv:1011.3245.
[4] L. Adleman, J. DeMarrais, and M. Huang. Quantum computability. SIAM Journal on Computing 26 (pp. 1524–1540), 1997.

[5] A. Ambainis. Superlinear Advantage for Exact Quantum Algorithms. In Proceedings of the 45th Annual STOC (pp. 891–900), 2013. arXiv:1211.0721.

[6] A. Ambainis, J. Iraids, and J. Smotrovs. Exact quantum query complexity of EXACT and THRESHOLD. arXiv:1302.1235.

[7] V. Arvind and P. P. Kurur. Graph isomorphism is in SPP. Proceedings of the IEEE 42nd annual FOCS (pp. 743–750), 2002.

[8] J. Barrett, C. M. Lee, N. de Beaudrap, and M. J. Hoban. The computational landscape of generalised probabilistic theories (to appear).

[9] N. de Beaudrap. On computation with ‘probabilities’ modulo k. arXiv:1405.7381, 2014.

[10] N. de Beaudrap, T. J. Osborne, and J. Eisert. Ground states of unfrustrated spin hamiltonians satisfy an area law. New Journal of Physics 12 (095007), 2010. arXiv:1009.3051.

[11] R. Beigel, J. Gill, U. Hertrampf. Counting classes: Thresholds, parity, mods, and fewness. Proc. 7th Annual STACS, Lecture Notes in Computer Science 415 (pp. 49–57), 1990.

[12] S. Beigi. Entanglement-assisted zero-error capacity is upper-bounded by the Lovász $\vartheta$ function. Physical Review A 82 (010303), 2010. arXiv:1002.2488.

[13] E. Bernstein, U. Vazirani. Quantum Complexity Theory. SIAM Journal on Computing 26 (pp. 1411-1473), 1997.

[14] S. Bravyi. Efficient algorithm for a quantum analogue of 2-SAT. Preprint, quant-ph/0602108v1, 2006.

[15] H. Buhrman and R. de Wolf. Quantum Zero-Error Algorithms Cannot be Composed. Information Processing Letters 87 (pp. 79–84), 2003. arXiv:quant-ph/0211029.

[16] J. Chen, X. Chen, R. Duan, Z. Ji, and B. Zeng. No-go theorem for one-way quantum computing on naturally occurring two-level systems. Physical Review A 83 (050301), 2011. arXiv:1004.3787.

[17] D. Coppersmith. An approximate Fourier transform useful in quantum factoring. Technical Report RC19642, IBM, 1994. arXiv:quant-ph/0201067.

[18] T. S. Cubitt, D. Leung, W. Matthews, and A. Winter. Improving Zero-Error Classical Communication with Entanglement. Physical Review Letters 104 (230503), 2010. arXiv:0911.5300.

[19] C. M. Dawson and M. A. Nielsen. The Solovay–Kitaev algorithm. Quantum Information and Computation 6 (pp. 81–95), 2006. arXiv:quant-ph/0505030

[20] S. A. Fenner. PP-Lowness and a Simple Definition of AWPP. Theory of Computing Systems 36 (pp. 199–212), 2003.
[21] S. Fenner, L. Fortnow, and S. Kurtz. *Gap-definable counting classes.* Journal of Computer and System Sciences 48 (pp. 116–148), 1994.

[22] S. Fenner, L. Fortnow, S. Kurtz, and L. Li. *An oracle builder’s toolkit.* In Proceedings of the 8th IEEE Structure in Complexity Theory Conference (pp. 120–131), New York, 1993.

[23] S. Fenner, F. Green, S. Homer, and R. Pruim (1999). *Determining Acceptance Possibility for a Quantum Computation is Hard for the Polynomial Hierarchy.* Proc. Roy. Soc. A 455 (3953–3966). arXiv:quant-ph/9812056.

[24] L. Fortnow and J. Rogers. *Complexity Limitations on Quantum Computation.* Journal of Computer and System Sciences 59 (pp. 240–252), 1999. arXiv:cs/9811023.

[25] J. Kempe, A. Kitaev, and O. Regev. *The Complexity of the Local Hamiltonian Problem.* SIAM Journal on Computing 35 (pp 1070–1097), 2006. arXiv:quant-ph/0406180.

[26] A. Y. Kitaev, A. Shen, and M. N. Vyalyi. *Classical and Quantum Computation.* Graduate Studies in Mathematics, American Mathematical Society, 2002.

[27] R. A. C. Medeiros and F. M. de Assis. *Quantum zero-error capacity.* International Journal of Quantum Information 3 (pp. 135–139), 2005. arXiv:quant-ph/0611089

[28] A. Montanaro, R. Jozsa, and G. Mitchison. *On Exact Quantum Query Complexity.* Algorithmica 71 (pp. 775–796), 2015. arXiv:1111.0475.

[29] M. Mosca and C. Zalka. *Exact quantum Fourier transforms and discrete logarithm algorithms.* arXiv:quant-ph/0301093

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

[31] H. Nishimura and M. Ozawa. *Computational Complexity of Uniform Quantum Circuit Families and Quantum Turing Machines.* Theoretical Computer Science 276 (pp. 147–181), 2002. arXiv:quant-ph/9906095.

[32] T. Nishino. *Mathematical models of quantum computation.* New Generation Computing 20 (pp. 317–337), 2002.

[33] H. Spakowski, M. Thakur, and R. Tripathi. *Quantum and classical complexity classes: Separations, collapses, and closure properties.* Information and Computation 200 (pp. 1–34), 2005.

[34] H. Spakowski and R. Tripathi. *LWPP and WPP Are Not Uniformly Gap-definable.* Journal of Computer and System Sciences 72 (pp. 660-689), 2006.

[35] S. Toda and M. Ogiwara. *Counting Classes Are at Least As Hard As the Polynomial-time Hierarchy.* SIAM Journal on Computing 21 (pp. 316–328), 1992.

[36] L. G. Valiant. *Holographic Circuits.* In Proceedings of the 32nd annual ICALP (pp. 1–15), 2005.

[37] L. G. Valiant, V. V. Vazirani. *NP is as easy as detecting unique solutions.* Theoretical Computer Science 47 (pp. 85–93), 1986.