A linear-optical proof that the permanent is #P-hard

A Linear-Optical Proof that the Permanent is #P-Hard

Scott Aaronson*

For Les Valiant, on the occasion of his Turing Award

Abstract

One of the crown jewels of complexity theory is Valiant's 1979 theorem that computing the permanent of an $n \times n$ matrix is #P-hard. Here we show that, by using the model of *linear-optical quantum computing*—and in particular, a universality theorem due to Knill, Laflamme, and Milburn—one can give a different and arguably more intuitive proof of this theorem.

1 Introduction

Given an $n \times n$ matrix $A = (a_{i,j})$, the *permanent* of A is defined as

$$\mathrm{Per}\,(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)}.$$

A seminal result of Valiant [15] says that computing $\mathrm{Per}\,(A)$ is #P-hard, if A is a matrix over (say) the integers, the nonnegative integers, or the set $\{0,1\}$.[1] Here #P means (informally) the class of *counting problems*—problems that involve summing exponentially-many nonnegative integers—and #P-hard means "at least as hard as any #P problem."[2,3]

More concretely, Valiant gave a polynomial-time algorithm that takes as input an instance $\varphi(x_1, \ldots, x_n)$ of the Boolean satisfiability problem, and that outputs a matrix A_φ such that $\mathrm{Per}\,(A_\varphi)$ encodes the number of satisfying assignments of φ. This means that computing the permanent is at least as hard as counting satisfying assignments.

Unfortunately, the standard proof that the permanent is #P-hard is notoriously opaque; it relies on a set of gadgets that seem to exist for "accidental" reasons. Could there be an alternative proof that gave more, or at least different, insight? In this paper, we try to answer that question by giving a new, quantum-computing-based proof that the permanent is #P-hard. In particular, we will derive the permanent's #P-hardness as a consequence of the following three facts:

*MIT. Email: aaronson@csail.mit.edu. This material is based upon work supported by the National Science Foundation under Grant No. 0844626. Also supported by a DARPA YFA grant and a Sloan Fellowship.

[1] See Hrubes, Wigderson, and Yehudayoff [7] for a recent, "modular" presentation of Valiant's proof (which also generalizes the proof to the noncommutative and nonassociative case).

[2] See the Complexity Zoo (www.complexityzoo.com) for the definitions of #P and other complexity classes used in this paper.

[3] If A is a nonnegative integer matrix, then $\mathrm{Per}\,(A)$ is *itself* a #P function, which implies that it is #P-*complete* (the term for functions that are both #P-hard and in #P). If A can have negative or fractional entries, then strictly speaking $\mathrm{Per}\,(A)$ is no longer #P-complete, but it is still #P-hard and computable in the class $\mathsf{FP}^{\#\mathsf{P}}$.

(1) *Postselected linear optics* is capable of universal quantum computation, as shown in a celebrated 2001 paper of Knill, Laflamme, and Milburn [9] (henceforth referred to as KLM).[4]

(2) Quantum computations can encode #P-hard quantities in their amplitudes.

(3) Amplitudes in n-photon linear-optics circuits can be expressed as the permanents of $n \times n$ matrices.

Even though our proof is based on quantum computing, we stress that we have made it *entirely self-contained*: all of the results we need (including the KLM Theorem [9], and even the construction of the Toffoli gate from 1-qubit and CSIGN gates) are proved in this paper for completeness. We assume some familiarity with quantum computing notation (e.g., kets and quantum circuit diagrams), but not with linear optics.

1.1 Motivation

If one counts the complexity of all of the individual pieces we use—especially the universality results for quantum gates—then our reduction from #P to the permanent ends up being at least as complicated as Valiant's, and probably more so. In our view, however, this is similar to how writing a program in C++ tends to produce a longer, more complicated executable file than writing the same program in assembly language. Normally, one also cares about the length and readability of the *source code*! Our purpose in this paper is to illustrate how quantum computing provides a powerful "high-level programming language" in which one can, among other things, easily rederive the most celebrated result in the theory of #P-hardness.

But why does the world need a new proof that the permanent is #P-hard—especially a proof invoking what some might consider to be exotic concepts? Let us offer several answers:

- Any theorem as basic as the #P-hardness of the permanent deserves several independent proofs. And our proof really is "independent" of the standard one: rather than composing variable and clause gadgets,[5] we *multiply matrices* corresponding to quantum gates, and use ideas from linear optics to keep track of how such multiplications affect the permanent. One way to see the difference is that our proof never uses the notion of a cycle cover.

- While our proof, like the standard one, requires "gadgets" (one to simulate a Toffoli gate using CSIGN gates, another to simulate a CSIGN gate using postselected linear optics), the connection to quantum computing gives those gadgets a natural *semantics*. In other words, the gadgets were introduced for "practical" reasons having nothing to do with proving the permanent #P-hard, and can be motivated independently of that goal. *If* one already knows the quantum universality gadgets, *then* we offer what seems like a major advance in complexity-theoretic pedagogy: a proof that the permanent is #P-hard that can be reproduced on-the-spot from memory!

[4] KLM actually prove the stronger (and more practically-relevant) result that linear optics with *adaptive measurements* is capable of universal quantum computation. For our purposes, however, we only need the weaker fact that *postselected* measurements suffice for universal QC, which KLM prove as a lemma along the way to their main result.

[5] Indeed, our proof does not even go through the Cook-Levin Theorem: it reduces a #P computation directly to the permanent, without first reducing #P to $\#3SAT$.

- As Kuperberg [10] pointed out, by their nature, any #P-hardness proofs (including ours) that are based on "quantum postselection" almost *immediately* yield hardness of approximation results as well.

- We expect that the quantum postselection approach used here could lead to #P-hardness proofs for many other problems—including problems not already known to be #P-hard by other means. In this direction, one natural place to look would be special cases of the permanent.

1.2 Related Work

By now, there are many examples where quantum computing has been used to give new or simpler proofs of classical complexity theorems; see Drucker and de Wolf [6] for an excellent survey. Within the area of counting complexity, Aaronson [1] showed that the class PP is equal to PostBQP (quantum polynomial-time with postselection), and then used that theorem to give a simpler proof of the landmark result of Beigel, Reingold, and Spielman [4] that PP is closed under intersection. Later, also using the PostBQP = PP theorem, Kuperberg [10] gave a "quantum proof" of the result of Jaeger, Vertigan, and Welsh [8] that computing the Jones polynomial is #P-hard, and even showed that a certain approximate version is #P-hard (which had not been shown previously). Kuperberg's argument for the Jones polynomial is conceptually similar to our argument for the permanent.

There is also precedent for using linear optics as a tool to prove theorems about the permanent. Scheel [14] observed that the unitarity of linear-optical quantum computing implies the interesting fact that $|\mathrm{Per}\,(U)| \leq 1$ for all unitary matrices U.

Rudolph [13] showed how to encode quantum amplitudes directly as matrix permanents, and in the process, gave a "quantum-computing proof" that the permanent is #P-hard. However, a crucial difference is that Rudolph starts with Valiant's proof based on cycle covers, then recasts it in quantum terms (with the goal of making Valiant's proof more accessible to a physics audience). By contrast, our proof is independent of Valiant's; the tools we use were invented for separate reasons in the quantum computing literature.

There has been a great deal of work on linear-optical quantum computing, beyond the seminal KLM Theorem [9] on which this paper relies. Recently, Aaronson and Arkhipov [2] studied the complexity of sampling from a linear-optical computer's output distribution, assuming no adaptive measurements are available. By using the #P-hardness of the permanent as an "input axiom," they showed that this sampling problem is classically intractable unless $\mathsf{P}^{\#\mathsf{P}} = \mathsf{BPP}^{\mathsf{NP}}$. More relevant to this paper is an alternative *proof* that Aaronson and Arkhipov gave for their result. Inspired by work of Bremner, Jozsa, and Shepherd [5], the alternative proof combines Aaronson's PostBQP = PP theorem [1] with the fact that postselected linear optics is universal for PostBQP, and thereby avoids any direct appeal to the #P-hardness of the permanent. In retrospect, that proof was already much of the way toward a linear-optical proof that the permanent is #P-hard; this paper simply makes the connection explicit.

2 Background

Not by accident, this section constitutes the bulk of the paper. First, in Section 2.1, we fix some facts and notation about standard (qubit-based) quantum computing. Then, in Section 2.2, we

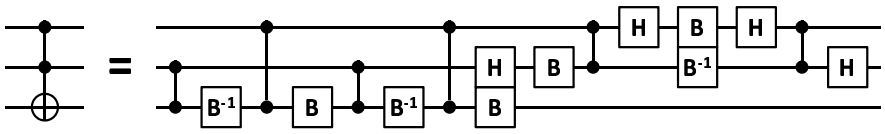

Figure 1: Simulating a Toffoli gate using CSIGN and 1-qubit gates.

give a short overview of those aspects of linear-optical quantum computing that are relevant for us, and (for completeness) prove the KLM Theorem in the specific form we will need.

2.1 Quantum Circuits

Abusing notation, we will often identify a quantum circuit Q with the unitary transformation that it induces: for example, $\langle 0\cdots 0|\, Q\, |0\cdots 0\rangle$ represents the amplitude with which Q maps its initial state to itself. We use $|Q|$ to denote the number of gates in Q.

The first ingredient we need for our proof is a convenient set of quantum gates (in the standard qubit model). Thus, let $\mathcal{G}$ be the set of gates consisting of (1) all 1-qubit gates, and (2) the 2-qubit *controlled-sign* gate

$$\text{CSIGN} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix},$$

which flips the amplitude if and only if both qubits are $|1\rangle$.[6] Then Barenco et al. [3] showed that $\mathcal{G}$ is a *universal* set of quantum gates, in the sense that $\mathcal{G}$ generates *any* unitary transformation on any number of qubits (without error). For our purposes, however, the following weaker result suffices.

Lemma 1 *$\mathcal{G}$ generates the Toffoli gate, the 3-qubit gate that maps each basis state $|x, y, z\rangle$ to $|x, y, z \oplus xy\rangle$.*

Proof. The circuit can be found in Nielsen and Chuang [11] for example, but we reproduce it in Figure 1 for completeness. In the diagram,

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

is the Hadamard gate,

$$B = \frac{1}{2} \begin{pmatrix} \sqrt{2+\sqrt{2}} & i\sqrt{2-\sqrt{2}} \\ i\sqrt{2-\sqrt{2}} & \sqrt{2+\sqrt{2}} \end{pmatrix}$$

is another 1-qubit gate, and the six vertical bars represent CSIGN gates.

■

[6] A more common 2-qubit gate than CSIGN is the *controlled-NOT* (CNOT) gate, which maps each basis state $|x, y\rangle$ to $|x, y \oplus x\rangle$. However, CSIGN is more convenient for linear-optics purposes, and is equivalent to CNOT by conjugating the second qubit with a Hadamard gate.

2.2 Linear-Optical Quantum Computing

We now give a brief overview of *linear-optical quantum computing* (LOQC), an alternative quantum computing model based on identical photons rather than qubits. For a detailed introduction to LOQC from a computer science perspective, see Aaronson and Arkhipov [2].

In LOQC, each basis state of our quantum computer has the form $|S\rangle = |s_1, \ldots, s_m\rangle$, where $s_1, \ldots, s_m$ are nonnegative integers summing to n. Here s_i represents the number of photons in the i^{th} location or "mode," and the fact that $s_1 + \cdots + s_m = n$ means that photons are never created or destroyed. One should think of m and n as both polynomially-bounded. For this paper, it will be convenient to assume that m is even, that $n = m/2$, and that the initial state has the form $|I\rangle = |0, 1, 0, 1, \ldots, 0, 1\rangle$: that is, one photon in each even-numbered mode, and no photons in the odd-numbered modes.

Let $\Phi_{m,n}$ be the set of nonnegative integer tuples $S = (s_1, \ldots, s_m)$ such that $s_1 + \cdots + s_m = n$, and let $\mathcal{H}_{m,n}$ be the Hilbert space spanned by basis states $|S\rangle$ with $S \in \Phi_{m,n}$. Then a general state in LOQC is just a unit vector in $\mathcal{H}_{m,n}$:

$$|\psi\rangle = \sum_{S \in \Phi_{m,n}} \alpha_S |S\rangle$$

with $\sum_{S \in \Phi_{m,n}} |\alpha_S|^2 = 1$.

To transform $|\psi\rangle$, one can select any $m \times m$ unitary transformation $U = (u_{ij})$. This U then induces a larger unitary transformation $\varphi(U)$ on the Hilbert space $\mathcal{H}_{m,n}$ of n-photon states. There are several ways to define $\varphi(U)$, but perhaps the simplest is the following formula:

$$\langle S|\varphi(U)|T\rangle = \frac{\mathrm{Per}(U_{S,T})}{\sqrt{s_1! \cdots s_m! t_1! \cdots t_m!}} \qquad (*)$$

for all tuples $S = (s_1, \ldots, s_m)$ and $T = (t_1, \ldots, t_m)$ in $\Phi_{m,n}$. Here $U_{S,T}$ is the $n \times n$ matrix obtained from U by taking s_i copies of the i^{th} row of U and t_j copies of the j^{th} column, for all $i, j \in [m]$. To illustrate, if

$$U = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

and $|S\rangle = |T\rangle = |2, 1\rangle$, then

$$U_{S,T} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}.$$

Intuitively, the reason the permanent arises in formula (*) is that there are $n!$ ways of mapping the n photons in basis state $|S\rangle$ onto the n photons in basis state $|T\rangle$. Since the photons are *identical bosons*, quantum mechanics says that each of those $n!$ ways contributes a term to the total $\langle S|\varphi(U)|T\rangle$, with the contribution given by the product of the transition amplitudes u_{ij} for each of the n photons individually.

It turns out that $\varphi(U)$ is always unitary and that φ is a homomorphism. Both facts seem surprising viewed purely as algebraic consequences of formula (*), but of course they have natural physical interpretations: $\varphi(U)$ is unitary because it represents an actual physical transformation that can be applied, and φ is a homomorphism because generalizing from one photon to n photons must commute with composing beamsplitters. In this paper, we will not need that $\varphi(U)$ is unitary; see Aaronson and Arkhipov [2] for a proof of that fact. Below we prove that φ is a homomorphism.

Lemma 2 *φ is a homomorphism.*

Proof. We want to show that for all tuples $S, T \in \Phi_{m,n}$ and all $m \times m$ unitaries U, V,

$$\langle S|\varphi(VU)|T\rangle = \langle S|\varphi(V)\,\varphi(U)|T\rangle = \sum_{R\in\Phi_{m,n}} \langle S|\varphi(V)|R\rangle\langle R|\varphi(U)|T\rangle.$$

By equation (*), the above is equivalent (after multiplying both sides by $\sqrt{s_1!\cdots s_m! t_1!\cdots t_m!}$) to the identity

$$\mathrm{Per}\left((VU)_{S,T}\right) = \sum_{R\in\Phi_{m,n}} \frac{\mathrm{Per}(V_{S,R})\,\mathrm{Per}(U_{R,T})}{r_1!\cdots r_m!}. \qquad (**)$$

We will prove identity (**) in the special case $n = m$ and $S = T = I = (1, 1, \ldots, 1)$, since the general case is analogous. We have

$$\begin{aligned}
\mathrm{Per}(VU) &= \sum_{\sigma\in S_n}\prod_{i=1}^{n}(VU)_{i,\sigma(i)} \\
&= \sum_{R\in\Phi_{n,n}}\left(\frac{1}{r_1!\cdots r_n!}\sum_{\tau,\xi\in S_n}\prod_{i=1}^{n}(V_{I,R})_{i,\xi(i)}\,(U_{R,I})_{i,\tau(i)}\right) \\
&= \sum_{R\in\Phi_{n,n}}\frac{\mathrm{Per}(V_{I,R})\,\mathrm{Per}(U_{R,I})}{r_1!\cdots r_n!}.
\end{aligned}$$

In the second line above, we decomposed the sum by thinking about each permutation $\sigma \in S_n$ as a product of *two* permutations: one, τ, that maps n particles in the initial configuration $|I\rangle$ to n particles in the intermediate configuration $|R\rangle$ when U is applied, and another, ξ, that maps n particles in the intermediate configuration $|R\rangle$ to n particles in the final configuration $|I\rangle$ when V is applied. This yields the same result, as long as we remember to sum over all possible intermediate configurations $R \in \Phi_{n,n}$, and also to divide each summand by $r_1!\cdots r_n!$, which is the size of R's automorphism group (i.e., the number of ways to permute the n particles within $|R\rangle$ that leave $|R\rangle$ unchanged). ■

In the standard qubit model, every unitary transformation can be decomposed as a product of *gates*, each of which acts nontrivially on only 1 or 2 qubits. Similarly, in LOQC, every unitary transformation can be decomposed as a product of *linear-optics gates*, each of which acts nontrivially on only 1 or 2 modes. Then a *linear-optics circuit* is simply a list of linear-optics gates applied to specified modes (or pairs of modes) starting from the initial state $|I\rangle = |0, 1, \ldots, 0, 1\rangle$.[7]

The last notion we need is that of *postselected LOQC*. In our context, postselection simply means measuring the number of photons in a given mode i, and conditioning on a particular result (for example, 0 photons, or 1 photon). After we postselect on the number of photons in some

[7] A crucial difference between standard quantum circuits and linear-optics circuits is that, whereas a standard quantum gate is the *tensor product* of a small (say 4×4) unitary matrix with an exponentially-large (say $2^{n-2} \times 2^{n-2}$) identity matrix, a linear-optics gate is the *direct sum* of a small (say 2×2) unitary matrix with a *polynomially*-large (say $(m-2) \times (m-2)$) identity matrix. It is only the homomorphism $U \to \varphi(U)$ that produces exponentially-large matrices. One consequence, pointed out by Reck et al. [12], is that, whereas most n-qubit unitary transformations require $\Omega\left(2^{2n}\right)$ gates to implement (as follows from an easy dimension argument), every m-mode unitary transformation U can be implemented using only $O\left(m^2\right)$ linear-optics gates.

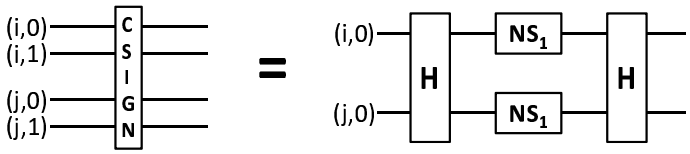

Figure 2: Simulating CSIGN by NS_1 and Hadamard.

mode, we will never use that mode for further computation.[8] For this reason, without loss of generality, we can defer all postselected measurements until the end of the computation.

Our #P-hardness proof will fall out as a corollary of the following universality theorem, which is implicit in the work of KLM [9]. Indeed, we *could* just appeal to the KLM construction as a "black box," but we choose not to do so, since the properties of the construction that we want are slightly different from the properties KLM want, and we wish to verify in detail that the desired properties hold.

Theorem 3 (following KLM [9]) *Postselected linear optics can simulate universal quantum computation. More concretely: there exists a polynomial-time classical algorithm that converts a quantum circuit Q over the gate set $\mathcal{G}$ into a linear-optics circuit L, so that*

$$\langle I| \varphi(L) |I\rangle = \frac{\langle 0\cdots 0| Q |0\cdots 0\rangle}{4^{\Gamma}},$$

where Γ is the number of CSIGN *gates in Q and $|I\rangle = |0,1,\ldots,0,1\rangle$ is the standard initial state.*

Proof. To encode a (qubit-based) quantum circuit by a postselected linear-optics circuit, KLM use the so-called *dual-rail representation* of a qubit using two optical modes. In this representation, the qubit $|0\rangle$ is represented as $|0,1\rangle$, while the qubit $|1\rangle$ is represented as $|1,0\rangle$. Thus, to simulate a quantum circuit that acts on k qubits, we need $2k$ optical modes. (We will also need additional modes to handle postselection, but we can ignore those for now.) Let the modes corresponding to qubit i be labeled $(i,0)$ and $(i,1)$ respectively. Notice that the initial state $|0\cdots 0\rangle$ in the qubit model maps onto the initial state $|I\rangle$ in the optical model.

Since φ is a homomorphism by Lemma 2, to prove the theorem it suffices to show how to simulate the gates in $\mathcal{G}$. Simulating a 1-qubit gate is easy: simply apply the appropriate 2×2 unitary transformation to the Hilbert space spanned by $|0,1\rangle$ and $|1,0\rangle$. The interesting part is how to simulate a CSIGN gate. To do so, KLM use another gate that they call NS_1, which applies the following unitary transformation to a single mode:

$$\mathrm{NS}_1 : \alpha_0|0\rangle + \alpha_1|1\rangle + \alpha_2|2\rangle \rightarrow \alpha_0|0\rangle + \alpha_1|1\rangle - \alpha_2|2\rangle.$$

(We do not care how NS_1 acts on $|3\rangle$, $|4\rangle$, and so on, since those basis states will never arise in our simulation.) Using NS_1, it is not hard to simulate CSIGN on two qubits i and j. The procedure, shown in Figure 2, is this: first apply a Hadamard transformation to modes $(i,0)$ and $(j,0)$. One

[8] In physics language, all photon-number measurements are assumed to be "demolition" measurements.

can check that this induces the following transformation on the state of $(i,0)$ and $(j,0)$:

$$\begin{aligned}
|0,0\rangle &\rightarrow |0,0\rangle \\
|1,0\rangle &\rightarrow \frac{|1,0\rangle + |0,1\rangle}{\sqrt{2}} \\
|0,1\rangle &\rightarrow \frac{|1,0\rangle - |0,1\rangle}{\sqrt{2}} \\
|1,1\rangle &\rightarrow \frac{|2,0\rangle - |0,2\rangle}{\sqrt{2}}
\end{aligned}$$

The key point is that we get a state involving 2 photons in the same mode, *if and only if* the modes $(i,0)$ and $(j,0)$ *both* contained a photon. Next, apply NS_1 gates to both $(i,0)$ and $(j,0)$. This flips the amplitude if and only if we started with $|1,1\rangle$. Finally, apply a second Hadamard transformation to $(i,0)$ and $(j,0)$, to complete the implementation of CSIGN.

We now explain how to implement NS_1 on a given mode i, using postselection. To do so, we need two additional modes j and k, which are initialized to the states $|0\rangle$ and $|1\rangle$ respectively. First we apply the following 3×3 unitary transformation to i, j, k:

$$W := \begin{pmatrix}
1-\sqrt{2} & \sqrt{\frac{3}{\sqrt{2}}-2} & \frac{1}{2^{1/4}} \\
\sqrt{\frac{3}{\sqrt{2}}-2} & \sqrt{2}-\frac{1}{2} & \frac{1}{2}-\frac{1}{\sqrt{2}} \\
\frac{1}{2^{1/4}} & \frac{1}{2}-\frac{1}{\sqrt{2}} & \frac{1}{2}
\end{pmatrix}.$$

Then we postselect on j and k being returned to the state $|0,1\rangle$. As shown in [9], this postselection always succeeds with amplitude $1/2$ (corresponding to probability $1/4$); and that conditioned on it succeeding, the effect is to apply NS_1 in mode i. To prove this, observe that since the number of photons is conserved, the effect of W on mode i must have the form

$$\alpha_0 |0\rangle + \alpha_1 |1\rangle + \alpha_2 |2\rangle \rightarrow \lambda_0\alpha_0 |0\rangle + \lambda_1\alpha_1 |1\rangle + \lambda_2\alpha_2 |2\rangle,$$

for some $\lambda_0, \lambda_1, \lambda_2$. Using formula (*), we then calculate

$$\begin{aligned}
\lambda_0 &= w_{33} = \frac{1}{2}, \\
\lambda_1 &= \mathrm{Per}\begin{pmatrix} w_{11} & w_{13} \\ w_{31} & w_{33} \end{pmatrix} = \frac{1}{2}, \\
\lambda_2 &= \frac{1}{2}\,\mathrm{Per}\begin{pmatrix} w_{11} & w_{11} & w_{13} \\ w_{11} & w_{11} & w_{13} \\ w_{31} & w_{31} & w_{33} \end{pmatrix} = -\frac{1}{2}.
\end{aligned}$$

This implies that the CSIGN circuit shown in Figure 2 succeeds with amplitude $1/4$ (corresponding to probability $1/16$), and furthermore, we know when it succeeds. ■

In the proof of Theorem 3, the main reason the matrix W looks complicated is simply that it needs to be unitary. However, notice that unitarity is irrelevant for our #P-hardness application—and if we drop the unitarity requirement, then we can replace W by a simpler 2×2 matrix, such as

$$Y := \begin{pmatrix} 1-\sqrt{2} & \sqrt{2} \\ 1 & 1 \end{pmatrix}.$$

To implement NS_1 on a given mode i, we would apply Y to i as well as another mode j that initially contains one photon, then postselect on j still containing one photon after Y is applied. One can verify by calculation that the effect on mode i is

$$\alpha_0 |0\rangle + \alpha_1 |1\rangle + \alpha_2 |2\rangle \rightarrow \lambda_0 \alpha_0 |0\rangle + \lambda_1 \alpha_1 |1\rangle + \lambda_2 \alpha_2 |2\rangle$$

where $\lambda_0 = \lambda_1 = 1$ and $\lambda_2 = -1$.

3 Main Result

In this section we deduce the following theorem, as a straightforward consequence of Theorem 3.

Theorem 4 *The problem of computing* $\mathrm{Per}(A)$*, given a matrix* $A \in \mathbb{Z}^{N \times N}$ *of* $\mathrm{poly}(N)$*-bit integers written in binary, is* #P*-hard under many-one reductions.*

In classical complexity theory, one is often more interested in various corollaries of Theorem 4: for example, that computing $\mathrm{Per}(A)$ remains #P-hard even if A is a *nonnegative* integer matrix, or a $\{-1, 0, 1\}$-valued matrix, or a $\{0, 1\}$-valued matrix. Valiant [15] gave simple reductions by which one can deduce all of these corollaries from Theorem 4. We do not know how to use the linear-optics perspective to get any additional insight into the corollaries.

Let C be a classical circuit that computes a Boolean function $C : \{0,1\}^n \rightarrow \{-1, 1\}$, and let $\Delta_C := \sum_{x \in \{0,1\}^n} C(x)$. Then computing Δ_C, given C as input, is a #P-hard problem essentially by definition. On the other hand, it is easy to encode Δ_C as an amplitude in a quantum circuit:

Lemma 5 *There exists a classical algorithm that takes a circuit* C *as input, runs in* $\mathrm{poly}(n, |C|)$ *time, and outputs a (qubit-based) quantum circuit* Q*, consisting of gates from* $\mathcal{G}$*, such that*

$$\langle 0 \cdots 0 | Q | 0 \cdots 0 \rangle = \frac{\Delta_C}{2^n}.$$

Proof. Let D_C be a $2^n \times 2^n$ diagonal unitary matrix whose (x, x) entry is $C(x)$. Then since the Toffoli gate is universal for classical computation, a quantum circuit consisting of 1-qubit gates and Toffoli gates can easily apply D_C. To do so, one uses the standard "uncomputing" trick:

$$|x\rangle \rightarrow |x\rangle |h_C(x)\rangle \rightarrow C(x) |x\rangle |h_C(x)\rangle \rightarrow C(x) |x\rangle,$$

where $h_C(x)$ is the complete history of a computation using Toffoli gates that produces $C(x)$. Now let $F = H_2^{\otimes n}$ be the quantum Fourier transform over $\mathbb{Z}_2^n$ (i.e., the Hadamard gate applied to each of n qubits), and let $Q = F D_C F$. Then

$$\langle 0 |^{\otimes n} Q | 0 \rangle^{\otimes n} = \left(\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} \langle x | \right) D_C \left(\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \right) = \frac{\Delta_C}{2^n}.$$

Finally, by Lemma 1, we can simulate each of the Toffoli gates in Q using gates from the set $\mathcal{G}$. ■

Let Q be the quantum circuit from Lemma 5, and assume Q uses $k = \mathrm{poly}(n, |C|)$ qubits. By Theorem 3, we can simulate Q by a linear-optics circuit L such that

$$\langle I | \varphi(L) | I \rangle = \frac{\langle 0 \cdots 0 | Q | 0 \cdots 0 \rangle}{4^\Gamma},$$

where $\Gamma = \text{poly}(n, |C|)$ is the number of CSIGN gates in Q. Furthermore, the circuit L uses $m := 2k + 4\Gamma$ optical modes. Let U be the $m \times m$ unitary matrix induced by L, and let V be the $(m/2) \times (m/2)$ submatrix of U obtained by taking the even-numbered rows and columns only. Then we have

$$\begin{aligned}\text{Per}(V) &= \langle I| \varphi(L) |I\rangle \\ &= \frac{\langle 0\cdots 0| Q |0\cdots 0\rangle}{4^\Gamma} \\ &= \frac{\Delta_C}{2^n 4^\Gamma}\end{aligned}$$

where the first line follows from formula (*) and the third from Lemma 5. Since V can be produced in polynomial time given C, this already shows that computing $\text{Per}(V)$ to sufficient precision is #P-hard.

However, we still need to deal with the issue that the entries of V are real numbers.[9] Let $b := \lceil \log_2(n!) + 2n + 2\Gamma \rceil$. Then notice that truncating the entries of V to b bits of precision produces a matrix $\widetilde{V}$ such that

$$\begin{aligned}\left|\text{Per}(\widetilde{V}) - \text{Per}(V)\right| &\leq n! \left(1 - \left(1 - \frac{1}{2^b}\right)^n\right) \\ &\leq \frac{n! \cdot n}{2^b} \\ &\leq \frac{1}{2^{n+2} 4^\Gamma}\end{aligned}$$

for sufficiently large n, and hence

$$\left\lfloor 2^n 4^\Gamma \text{Per}(\widetilde{V}) \right\rfloor = 2^n 4^\Gamma \text{Per}(V) = \Delta_C.$$

For this reason, we can assume that each entry of V has the form $k/2^b$ for some integer $k \in \left[-2^b, 2^b\right]$. Now set $A := 2^b V$. Then A is an integer matrix satisfying $\text{Per}(A) = 2^{bn} \text{Per}(V)$, whose entries can be specified using $b + O(1) = \text{poly}(n, |C|)$ bits each. This completes the proof of Theorem 4.

We conclude by noticing that our proof yields not only Theorem 4, but also the following corollary:

Corollary 6 *The problem of computing* $\text{sgn}(\text{Per}(A)) := \text{Per}(A) / |\text{Per}(A)|$, *given a matrix* $A \in \mathbb{Z}^{N \times N}$ *of* $\text{poly}(N)$*-bit integers written in binary, is* #P*-hard under Turing reductions.*

Proof. By the above equivalences, it suffices to show that computing $\text{sgn}(\Delta_C)$ is #P-hard. This is true because, given the ability to compute $\text{sgn}(\Delta_C)$, we can determine Δ_C *exactly* using binary search. In more detail, given a positive integer k, let $C[k]$ denote the circuit C modified to contain k additional inputs x such that $C(x) = 1$, and let $C[-k]$ denote C modified to contain k additional x's such that $C(x) = -1$. Then clearly

$$\begin{aligned}\Delta_{C[k]} &= \Delta_C + k, \\ \Delta_{C[-k]} &= \Delta_C - k.\end{aligned}$$

[9] Indeed, the matrices that we multiply to obtain U can be *complex* matrices, but U itself (and hence the submatrix V) will always be real.

Thus we can use the following strategy: compute the signs of $\Delta_{C[1]}, \Delta_{C[-1]}, \Delta_{C[2]}, \Delta_{C[-2]}, \Delta_{C[4]}, \Delta_{C[-4]}$,and so on, increasing k by successive factors of 2, until a k is found such that $\text{sgn}\left(\Delta_{C[k]}\right) \neq \text{sgn}\left(\Delta_{C[2k]}\right)$. At that point, we know that Δ_C must be between k and $2k$. Then by computing $\text{sgn}\left(\Delta_{C[3k/2]}\right)$, we can decide whether Δ_C is between k and $3k/2$ or between $3k/2$ and $2k$, and so on recursively until Δ_C has been determined exactly. ■

Corollary 6 implies, in particular, that approximating $\text{Per}\,(A)$ to within any multiplicative factor is #P-hard—since to output a multiplicative approximation, at the least we would need to know whether $\text{Per}\,(A)$ is positive or negative.

Using a more involved binary search strategy (which we omit), one can show that, for any $\beta\,(N) \in [1, \text{poly}\,(N)]$, even approximating $|\Delta_C|$ or Δ_C^2 to within a multiplicative factor of $\beta\,(N)$ would let one compute Δ_C exactly, and is therefore #P-hard under Turing reductions. It follows from this that approximating $|\text{Per}\,(A)|$ or $\text{Per}\,(A)^2$ to within a multiplicative factor of $\beta\,(N)$ is #P-hard as well. (Aaronson and Arkhipov [2] gave a related but more complicated proof of the #P-hardness of approximating $|\text{Per}\,(A)|$ and $\text{Per}\,(A)^2$, which did not first replace $\text{Per}\,(A)$ with Δ_C.)

4 Acknowledgments

I am grateful to Alex Arkhipov and Michael Forbes for helpful discussions, and to Andy Drucker, Greg Kuperberg, Avi Wigderson, Ronald de Wolf, and the anonymous reviewers for their comments.

References

[1] S. Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proc. Roy. Soc. London*, A461(2063):3473–3482, 2005. quant-ph/0412187.

[2] S. Aaronson and A. Arkhipov. The computational complexity of linear optics. In *Proc. ACM STOC*, 2011. ECCC TR10-170, arXiv:1011.3245.

[3] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Phys. Rev. A*, 52(3457), 1995. quant-ph/9503016.

[4] R. Beigel, N. Reingold, and D. Spielman. PP is closed under intersection. *J. Comput. Sys. Sci.*, 50(2):191–202, 1995.

[5] M. Bremner, R. Jozsa, and D. Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proc. Roy. Soc. London*, A467(2126):459–472, 2010. arXiv:1005.1407.

[6] A. Drucker and R. de Wolf. Quantum proofs for classical theorems. *Theory of Computing Graduate Surveys*, (2):1–54, 2011. arXiv:0910.3376, ECCC TR03-048.

[7] P. Hrubes, A. Wigderson, and A. Yehudayoff. Relationless completeness and separations. In *Proc. IEEE Conference on Computational Complexity*, pages 280–290, 2010. ECCC TR10-040.

[8] F. Jaeger, D. L. Vertigan, and D. Welsh. On the computational complexity of the Jones and Tutte polynomials. *Math. Proc. Cambridge Philos. Soc.*, 108(1):35–53, 1990.

[9] E. Knill, R. Laflamme, and G. J. Milburn. A scheme for efficient quantum computation with linear optics. *Nature*, 409:46–52, 2001. See also quant-ph/0006088.

[10] G. Kuperberg. How hard is it to approximate the Jones polynomial? arXiv:0908.0512, 2009.

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

[12] M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani. Experimental realization of any discrete unitary operator. *Phys. Rev. Lett.*, 73(1):58–61, 1994.

[13] T. Rudolph. A simple encoding of a quantum circuit amplitude as a matrix permanent. arXiv:0909.3005, 2009.

[14] S. Scheel. Permanents in linear optical networks. quant-ph/0406127, 2004.

[15] L. G. Valiant. The complexity of computing the permanent. *Theoretical Comput. Sci.*, 8(2):189–201, 1979.