Quantum Circuits, Polynomials, and Entanglement Measures
WORKING DRAFT

Kenneth Regan
University at Buffalo (SUNY)

Amlan Chakrabarti
University of Calcutta

July 8, 2012

Abstract

We extend the polynomial construction of Dawson et al. [DHH$^+$04] so that, besides working for any "balanced" set of quantum gates, it produces a single polynomial over any sufficiently structured field or ring. We give polynomials that like theirs treat the phases additively in modular rings, and others that treat them multiplicatively over any ring with enough roots of unity. The former appear best for practical algebraic simulation of quantum computations, along lines of Gerdt and Severyanov [GS06]. The latter may have nicer theoretical properties, which we explore with focus on the wide-open problem of quantifying the power of quantum circuits to effect multi-partite entanglements.

1 Quantum Circuits and Polynomials

A *quantum circuit* C has some number n of qubits pictured as "lines" running from left to right, and some number s of gates. Each gate g takes some m-subset S of the lines for its inputs, and has the same m-subset S as its outputs. It can be defined by a sequence $g^1_{S_1}, \dots, g^m_{S_s}$ specifying the type and attached set of lines for each gate. Every gate computes a unitary linear transformation of its inputs, making the whole circuit representable by a unitary transformation U on $\mathbf{C}^{2^n}$. A binary string a of length n is encoded by the n-qubit state $|a\rangle$, which is *completely separable* as the tensor product $|a_1\rangle \otimes |a_2\rangle \otimes \cdots \otimes |a_n\rangle$, and thus is a canonical basis vector in $\mathbf{C}^{2^n}$. The output of the circuit is determined by applying a measurement M to $C_n |a\rangle$, which can be regarded as having probabilistic outcomes 0 or 1. A family of circuits C_n can be said to recognize a language L with bounded error if for all n and $a \in \{ 0, 1 \}^n$, $\Pr[M(C_n |a\rangle) = L(a)] > 2/3$. We model this measurement directly, but note as in [DHH$^+$04] that it can also be deduced from value(s) of the triple product $\langle a| \; C_n \; |b\rangle$ for a modified C_n and suitably chosen basis vector(s) b. Initially we model $\langle a| \; C_n \; |b\rangle$.

Dawson et al. [DHH$^+$04] translated quantum circuits of Hadamard, CNOT, and Toffoli gates, and with a final measurement in the canonical basis, into systems of polynomial equations over the finite field F_2. Their construction yields a clear simulation of BQP by so-called GapP functions and containment of BQP in the complexity class PP. For rotation gates typified by $R^{1/8}$, however, they gave only a sketch using addition modulo 8 and multiplication (perhaps unnecessarily) stated as being modulo 2.

We present variants of their construction that work over any field or ring that contains an image of the kth roots of unity, provided all phase angles in the gates are multiples of $2\pi/k$. Ours build a single polynomial p with parameters $a_1, \dots, a_n$ standing for the input lines at the left, $b_1, \dots, b_n$ standing for the *measurement target* at the right, and variables z^j_i for the interior of the circuit. For our "additive" construction extending that of [DHH$^+$04] for a kind of gate G_j not considered there and $k = 2^r$, we also employ extra variables w_ℓ^j, $1 \leq \ell \leq r$. Here i runs from 1 to n and j runs from 1 to $s-1$. If we want to think of the input and output parameters as variables, we rename a_i to z_i^0 and b_i to z_i^s. Later we show how to reduce the interior variable set to z_i^j where line i comes out of an Hadamard or other "splitting" gate. Given any basis vector $|a\rangle$ as input and possible measurement outcome $|b\rangle$, the polynomial $p_{a,b}$ is obtained by substituting the corresponding classical string values $a, b \in \{ 0, 1 \}^n$ for the parameters a_i and b_i into p.

Our simulation theorems in Section 2 calculate $\langle a| \; C \; |b\rangle$ in terms of the numbers of 0-1 solutions to $p_{a,b}(\vec{z}_{i,j}) = e_\ell$, where e_ℓ ranges over the embedding of the kth roots of unity. The numbers of such solutions may be exponential, and for *general* polynomials p computing such numbers is #P-complete, hence NP-hard. This is not to say that the task is NP-hard for the particular polynomials p constructed from the quantum circuits, and certainly not that the languages accepted by the circuits can be NP-hard: it is commonly believed that BQP does not contain any NP-complete languages. However, it does say—as treated at further length in [DHH$^+$04]—that the polynomial construction does not imply an asymptotically feasible classical simulation of quantum circuits. Whether it facilitates simulation of moderately small interesting quantum circuits is the concrete practical question. Sections 3 and 4 give a catalog of gate polynomials and examples of simulating circuits.

On the upside, the different structure of our polynomials p may raise the hope in [DHH$^+$04] of shedding new analytical light on the complexity of quantum circuits. Moreover, we adduce that the translation into polynomial algebra will enable other important properties of quantum circuits to be characterized in classical mathematics. The most important property we seek to quantify is the circuit's capacity to produce entanglements. This appears to entail defining a measure of entanglement for n-partite quantum systems, and we note that for $n \geq 3$ there is yet no agreement in the literature on such a measure (see [HH08] and references therein). Our aim in Section 6 is to find a salient mathematical invariant E of our circuit polynomials p_C, show that $E(p_C)$ satisfies natural axioms of an entanglement measure, and argue that E is compelling enough to be the unique victor. The ulterior motive is to position such an E as a complexity measure, perhaps reflecting the degree of physical effort that would be needed to combat decoherence in any physical implementation of C. Finally Section 7 gives conclusions and further tasks.

2 Simulation Theorems

Following [DHH$^+$04], say a quantum gate g_j is *balanced* if all nonzero entries in its $2^m \times 2^m$ unitary matrix have the same magnitude r_j. Note that the same applies to the $2^n \times 2^n$ matrix U_j of the operation that acts as the identity on the other $n-m$ qubit lines, with the same r_j. The Hadamard, CNOT, Toffoli, and rotation gates described in Section 1 are all balanced. Automatically every unitary matrix with one nonzero entry per row is balanced, with $r = 1$, so the condition is meaningful only for gates such as Hadamard that intuitively "split" the incoming quantum signal. A quantum circuit is *balanced* if all of its gates are balanced. This is not a great restriction—in fact, it is hard to find examples of useful quantum circuits int he literature that aren't balanced, and the universality theorems mentioned above imply that they can be efficiently simulated by balanced circuits anyway. Define $R = R(C)$ to be the product of r_j over all gates in C.

Also define $k = k(C)$ to be the least integer such that all angles θ in entries $re^{i\theta}$ of gates in C are integer multiples of $2\pi/k$. For example if C has only Hadamard, CNOT, and Toffoli gates then $k(C) = 2$; if it adds the T gate which has an entry $e^{\pi i/4}$, then $k(C) = 8$. Then say a ring is (multiplicatively) *adequate for* C if it admits a 1-1 mapping e from the complex k-th roots of unity such that for all such roots a, b, $e(ab) = e(a)e(b)$. In the additive case we will limit attention to rings extending $\mathbf{Z}_k$, which have an additive embedding e_+ such that $e_+(ab) = e_+(a) + e_+(b)$. In the multiplicative case we also define $e(0) = 0$, but for the additive case we will map 0 to a set of variables.

For a polynomial p in variables a_i, b_i, z_i^j ($1 \leq i \leq n$; $1 \leq j \leq s-1$), and arguments $a, b \in \{\ 0, 1\ \}^n$, $p_{a,b}$ denotes the polynomial in variables z_i^j resulting from substituting the arguments. Then $N_B[p_{a,b}(z_i^j) = v]$ denotes the number of *binary* solutions to the equation, i.e. with an assignment from $\{\ 0, 1\ \}^{n(s-1)}$ to the z_i^j variables. We state and prove the "multiplicative" version of our simulation first.

Theorem 2.1 *There is an efficient uniform procedure that transforms any balanced n-qubit quantum circuit C with s gates into a polynomial p such that for all $a, b \in \{\ 0, 1\ \}^n$:*

$$\langle a|\ C\ |b\rangle = R \sum_{\ell=0}^{k-1} \omega^\ell N_B[p_{a,b}(z_i^j) = e(\omega^\ell)] \tag{1}$$

over any adequate ring. The size of p as a product-of-sums-of-products of z_i^j and $(1 - z_i^j)$ is $O(2^{2m}ms)$ where m is the maximum arity of a gate in C, and the time to write p down is the same ignoring factors of $\log n$ *and* $\log s$ *for variable labels.*

When C is a circuit of Hadamard and Toffoli gates, and any other gates with real entries, we can take $\ell = 2$, and all we need in the target field or ring F is that -1 is different from $+1$. Equation 1 then simplifies to

$$\langle a|\ C\ |b\rangle = R(N_B[p_{a,b}(z) = 1] - N_B[p_{a,b}(z) = -1]).$$

Dawson et al. [DHH+04] achieve the same effect with an additive embedding of $\{\ -1, 1\ \}$ into $\{\ 0, 1\ \}$, with the polynomial(s) over Z_2, and their p is a simple sum of products of similarly-bounded degree. To model gates with complex entries such as $e^{\pi i/4}$, however, they resort to arithmetic with addition modulo a different base (here, 8) than multiplication. We find, however, that the multiplicative theorem better streamlines the issues in its proof, which is then "re-usable" for our generalized additive theorem below.

Proof: Let $c^0 = a, c^1, \ldots, c^{s-1}, c^s = b$ stand for basis elements in $\{\ 0, 1\ \}^n$. For each j, let $U_j(c^{j-1}, c^j)$ stand for the entry of the $2^n \times 2^n$ operator matrix with row indexed by c^{j-1} and column indexed by c^j. By assumption this entry is either 0 or has the form $r_j e^{i\theta}$ where r_j depends only on j and θ is an integer multiple of $2\pi/k$. In either case we may write $u_j(c^{j-1}, c^j) =$

$U_j(c^{j-1}, c^j)/r_j$. Then:

$$\begin{aligned}
\langle a|\ C\ |b\rangle &= \sum_{c^1,\ldots,c^{s-1}} \langle a|\ U_1\ |c^1\rangle \langle c^1|\ U_2\ |c^2\rangle \cdots \langle c^{s-1}|\ U_s\ |b\rangle \\
&= \sum_{c^1,\ldots,c^{s-1}} \prod_{j=1}^{s} U_j(c^{j-1}, c^j) \\
&= r_1 r_2 \cdots r_s \sum_{c^1,\ldots,c^{s-1}} \prod_{j=1}^{s} u_j(c^{j-1}, c^j) \\
&= R \sum_{\ell=0}^{k-1} P_\ell, \qquad \text{where} \\
P_\ell &= \sum_{\vec{c}:\prod_j u_j(c^{j-1},c^j)=\omega^\ell} \omega^\ell.
\end{aligned}$$

We first construct a huge polynomial $\hat{p}$ such that for each ℓ, and any a, b, the 0-1 solutions to $\hat{p}_{a,b}(z_i^j) = e(\omega^\ell)$ are precisely the values of $c_i^1, \ldots, c_i^{s-1}$ $(1 \leq i \leq n)$ under the sum in the definition of P_ℓ. Then we show how to simplify $\hat{p}$ to a polynomial p of the desired size without changing the number of solutions. For each j and each $c \in \{\ 0, 1\ \}^n$ define the "indicator" of c by

$$I_c^j = \prod_{i=1}^{n} (c_i z_i^j + (1 - c_i)(1 - z_i^j)),$$

which becomes a product of z_i^j or $(1 - z_i^j)$ according to the bits of c. Then $I_c^j(\vec{z}) = 1$ if $z_1^j, \ldots, z_n^j$ are assigned the respective bits of c, and 0 otherwise. Now define $\hat{p} = \prod_{j=1}^{s} P_j$, where

$$P_j = \sum_{c,d \in \{\ 0,1\ \}^n} I_c^{j-1} I_d^j e(u_j(c, d)).$$

For any assignment c^{j-1} to the z_i^{j-1} variables and c^j to the z_i^j, all terms in this sum are 0 except the one for $c = c^{j-1}$ and $d = c^j$, which has value $e(u_j(c^{j-1}, c^j))$. Thus the only nonzero values of $\hat{p}_{a,b}(\vec{z})$, indeed of $\hat{p}(z_1^0, \ldots, z_n^0, z_1^1, \ldots, z_n^s)$, are products of the form

$$\prod_{j=1}^{s} e(u_j(c^{j-1}, c^j)) = e(\omega^{\sum_j u'_j(c^{j-1},c^j) \bmod k})$$

where (since the value is nonzero) we may write $u'_j(c, d)$ for the integer by which $u_j(c, d)$ is a multiple of ω. It is now clear that terms under the sum defining P_ℓ are in 1-1 correspondence with solutions to $\hat{p}_{a,b}(\vec{z}) = e(\omega^\ell)$, for each ℓ, from which (1) follows.

It remains to reduce $\hat{p}$ down to a polynomial p of the stated size, without changing the number of solutions. Consider any qubit line i that is not involved in gate g_j, so that U_j acts as the identity on i. The product terms in P_j divide into four groups with $z_i^{j-1} z_i^j$, $(1 - z_i^{j-1})(1 - z_i^j)$, $z_i^{j-1}(1 - z_i^j)$, and $(1 - z_i^{j-1}) z_i^j$, respectively. Because U_j acts as the identity on line i, the latter two groups occur only for entries $u_j(c, d)$ that are 0, so they vanish. Since having $z_i^{j-1} = 0$ while $z_i^j = 1$ or vice-versa zeroes out the former two groups as well, any 0-1 solution to $\hat{p}_{a,b}(\vec{z}) = e(\omega^\ell)$ must have $z_i^j = z_i^{j-1}$. Hence without changing the number of binary solutions, we may for each such i substitute $z_i^j = z_i^{j-1}$, delete the terms for the vanishing groups, and make the factors on the surviving groups just z_i^{j-1} and $(1 - z_i^{j-1})$, respectively. Doing so cuts the size of P_j down by a factor of 2^{n-m}. But since P_j is a sum of 2^{2n} terms, each a product of $2n$-many z or $(1-z)$ factors, this is not yet good enough.

Again focusing on qubit line i, the remaining terms have the forms

$$z_i^{j-1} H_1 \qquad \text{and} \qquad (1 - z_i^{j-1}) H_2.$$

Because U_j acts as the identity on qubit i, every entry $U(c,d)$ where $c_i = d_i = 1$ equals the entry $U(c',d')$ where $c'_i = d'_i = 0$ with the other bits the same as in c and d. Hence terms in H_1 pair off with equal terms in H_2. We claim that we can replace the remaining terms in P_j by just H_1 $(= H_2)$. Doing this does not add any new solutions, because if a solution makes $z_i^{j-1} = 1$ then the original P_j got the same contribution from H_1 as it gets now (with H_2 being zeroed), and similarly for $z_i^{j-1} = 0$. Nor does doing this remove any solutions—nor does it remove all dependence on z_i^{j-1} because z_i^j for which it was substituted may be involved in U^{j+1}. Applying this second process cuts the number of terms in P_j down by another factor of (at least) 2^{n-m}, and also cuts the degrees of terms down from n to m. The polynomial p obtained by doing this for all j thus has size $O(s2^m m)$ as claimed. □

Variations: Given any ring R, we can adjoin an element u with minimum polynomial $u^k - 1$, and make this work over the extension $R[u]$. We can also treat u as a variable, writing u^ℓ in place of $e(\omega^\ell)$, and add $u^k = 1$ as a second equation. For gates such as CNOT and Toffoli that do not involve splitting, we can also do substitution on the lines that are involved in the gate, exactly as in [DHH+04]. We prefer, however, to define "the" circuit polynomial p_C as p above without doing so, reserving $p'_C, p''_C, \ldots$ for versions that do substitution and/or reduction modulo the "Boolean ideal" generated by $\{ (z_i^j)^2 - z_i^j \}$.

2.1 Additive representations

Here we take a 1-1 mapping e from the kth roots of unity that satisfies $e(\omega^\ell \omega^m) = e(\omega^{\ell+m}) = e(\omega^\ell) + e(\omega^m)$. This entails $e(1) = 0$, and we may suppose that the target ring is $\mathbf{Z}_k$, as in [DHH+04]. The intent is to employ the same phase-indicator terms P_j as above and write

$$q(\vec{z}) = \sum_{j=1}^{s} P_j.$$

The problem is the handling of cases where P_j evaluates to 0 for reasons other than $U(c,d) = 1$. These are covered by substitution for mismatches between d_i and c_i and cases like CNOT and Toffoli and tensor products with Hadamard gates where the number of non-zero entries in each row is a power of 2. However, we do not know how to apply substitution cleanly or implement the suggestions in [DHH+04] to handle multi-qubit gates like the following:

$$A = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & -1 & 1 & 0 \\ 1 & 1 & 0 & -1 \end{bmatrix}.$$

To solve the problem—and optionally handle the other cases when substitution is undesired, we can employ variables that take values in the range of e. When k is a power of 2, i.e. $k = 2^r$, and the embedding is into $\mathbf{Z}_k$, we can employ variables $w^j_0, \ldots, w^j_{r-1}$ with 0-1 arguments. Then we define

$$E^j(0) = \sum_{m=0}^{r-1} 2^s w^j_m; \qquad E^j(\omega^\ell) = e(\omega^\ell) = \ell.$$

It is important to have distinct variables w^j_m for each gate g_j that needs them. The revised phase-indicator term now becomes

$$Q_j = \sum_{c,d \in \{\ 0,1\ \}^n} I^{j-1}_c I^j_d E^j(u_j(c,d)).$$

Theorem 2.2 *There is an efficient uniform procedure that transforms any balanced n-qubit quantum circuit C with s gates, whose nonzero entries have phase a multiple of* $2\pi/k$ *for k a power* 2^r*, into a polynomial* $q(\vec{a}, \vec{b}, \vec{z}, \vec{w})$ *over* $\mathbf{Z}_k$ *such that for all* $a, b \in \{\ 0,1\ \}^n$*:*

$$\langle a|\ C\ |b\rangle = Rk^{-s} \sum_{\ell=0}^{k-1} \omega^\ell N_B[q_{a,b}(z^j_i, w^j_s) = e(\omega^\ell)], \tag{2}$$

with R and the size of q the same as for p in Theorem 2.1.

Proof: Instead of a product, this time the huge polynomial analogous to $\hat{p}$ is

$$\hat{q}(z^0_0, \ldots, z^s_n, w^1_1, \ldots, w^s_r) = \sum_{j=1}^{s} Q_j \pmod{k}.$$

The reductions by substitution to the final polynomial $q = q_C$ are the same, and the extra contribution to size from the w^j_m variables is ignorable. It remains to verify (2) after the changes.

We call a 0-1 assignment α to the z^j_i variables "novel" if it makes $I^{j-1}_c I^j_d$ nonzero for some j, c, d for which $U_j(c,d) = 0$. We denote by j_α the least such j. All novel assignments make the corresponding term $P_j = \sum_{c,d} I^{j-1}_c I^j_d e(u_j(c,d))$ in the proof of Theorem 2.1 zero, since there can be only one possible c, d making $I^{j-1}_c(\alpha) I^j_d(\alpha) \neq 0$ and $u_j(c,d) = 0$ for that pair. Hence they zero out the product over P_j, and so are not solutions to $p_{a,b}(\vec{z}) = e(\omega^\ell)$ for any ℓ. Thus all solutions counted in Theorem 2.1 are not novel, and in particular have no dependence on the w^j_m variables, for any j. Each such solution thus corresponds to k^s solutions to $q_{a,b}(\vec{z}, \vec{w}) = e(\omega^\ell)$, and the factor of k^{-s} in (2) restores the original count. Now it remains to argue that the novel solutions, when extended to the w^j_m variables, make a net-zero contribution to the expression for $\langle a|\ C\ |b\rangle$.

We do this by exhibiting 1-1 correspondences between the novel solutions to $q_{a,b}(\vec{z}, \vec{w}) = e(\omega^\ell)$ and $q_{a,b}(\vec{z}, \vec{w}) = e(\omega^{\ell+1})$ for any ℓ. Namely, given a solution α to the former, let α' be obtained by incrementing the binary string $a_m a_{m-1} \cdots a_1$ obtained from the respective assigned values $a_1, \ldots, a_m$ of $w^{j_\alpha}_1, \ldots, w^{j_\alpha}_m$ in α, wrapping 1^m to 0^m. Doing so makes $E^{j_\alpha}(0)(a'_1, \ldots, a'_m) = E^{j_\alpha}(0)(a_1, \ldots, a_m) + 1 \pmod{k}$. It also makes $q_{a,b}(\alpha') = q_{a,b}(\alpha) + 1 \pmod{k}$ because α makes only one indicator term in Q_j nonzero, while α' makes the same term nonzero and has no effect on terms $Q_{j'}$ for $j' \neq j$. Hence α' is a solution to $q_{a,b}(\vec{z}, \vec{w}) = e(\omega^{\ell+1})$. Moreover $j_{\alpha'} = j_\alpha$, and by the way this is defined, there is no other assignment β whose incrementing yields α'. Hence the numbers N'_ℓ of *novel* solutions to $q_{a,b}(\vec{z},\vec{w}) = e(\omega^\ell)$ are all equal. This makes

$$\begin{aligned} & Rk^{-s}\sum_{\ell=0}^{k-1}\omega^\ell N_B[q_{a,b}(z_i^j, w_s^j) = e(\omega^\ell)] \\ = & \sum_{\ell=0}^{k-1}\omega^\ell N_B[p_{a,b}(z_i^j) = e(\omega^\ell)] + \sum_{\ell=0}^{k-1}\omega^\ell N'_\ell \\ = & \langle a|\ C\ |b\rangle + 0 \end{aligned}$$

because the sum of the complex kth roots of unity cancels to zero. This proves (2) and hence the theorem. ☐

Variations: When k is not a power of 2, we could extend (and simplify) the proof by having single variables w^j that range over $\mathbf{Z}_k$ rather than $\{\ 0, 1\ \}$. We could also extend the z_i^j variables to $\mathbf{Z}_k$ by adding the equations $(z_i^j)^2 - z_i^j = 0$ to the system, or retain a single polynomial equation by working over the quotient ring by the ideal generated by the $(z_i^j)^2 - z_i^j$.

2.2 Measurements and conjugation

First we observe that for both kinds of representations, substitution for a_i and/or b_i equals taking an inner product with a standard basis vector, and undoing such a substitution corresponds to summing. In particular,

$$\begin{aligned} \sum_b \langle a|\ C\ |b\rangle &= R\sum_{\ell=0}^{k-1}\omega^\ell \sum_b N_B[p_{a,b}(z_i^j) = e(\omega^\ell)] \\ &= R\sum_{\ell=0}^{k-1}\omega^\ell N_B[p_a(z_i^j) = e(\omega^\ell)] \end{aligned}$$

However, it is not the case that summing $N_B[p\ldots]$ over all $b = 0c$ that begin with 0 gives the amplitude of measuring 0 on the first qubit line. Instead, writing $\alpha_c = \langle a|\ C\ |0c\rangle$ and $\beta_c = \langle a|\ C\ |1b\rangle$ gives us a formula for the classical probability: $\sum_c |\alpha_c|^2 =$

$$\begin{aligned} \sum_c \alpha_c\bar{\alpha}_c &= \sum_c R^2\left(\sum_{\ell=0}^{k-1}\omega^\ell N_B[p_{a,0c}(z_i^j) = e(\omega^\ell)]\right)\cdot\left(\sum_{m=0}^{k-1}\omega^{k-m}N_B[p_{a,0c}(z_i^j) = e(\omega^m)]\right) \\ &= R^2\sum_{\ell,m=0}^{k-1}\omega^{\ell-m}\sum_c N_B[p_{a,0c}(z_i^j) = e(\omega^\ell)]\cdot N_B[p_{a,0c}(z_i^j) = e(\omega^m)] \end{aligned}$$

with $\ell - m$ taken modulo k. To process this further, we first want to find a polynomial $\bar{p}$ such that for each m,

$$N_B[\bar{p}_{a,0c}(z_i^j) = e(\omega^{-m})] = N_B[p_{a,0c}(z_i^j) = e(\omega^m)]. \tag{3}$$

We call $\bar{p}$ the *classical conjugate* of p. The property (3) is multiplicative for multiplicative representations and additive for additive ones, so we may focus on conjugating the individual phase terms for each gate. This can be done on a case-by-case basis. Thus for the multiplicative representations,

$$\begin{aligned}
\Pr(0) &= R^2 \sum_{\ell,m=0}^{k-1} \omega^{\ell-m} \sum_c N_B[p_{a,0c}(z_i^j) = e(\omega^\ell)] \cdot N_B[\bar{p}_{a,0c}({z'}_i^j) = e(\omega^{-m})] \\
&= R^2 \sum_c \sum_{r=0}^{k-1} \omega^r \sum_{\ell,m:\ell-m=r} N_B[p_{a,0c}(z_i^j) = e(\omega^\ell)] \cdot N_B[\bar{p}_{a,0c}({z'}_i^j) = e(\omega^{-m})] \\
&= R^2 \sum_c \sum_{r=0}^{k-1} \omega^r N_B[p_{a,0c}(z_i^j)\bar{p}_{a,0c}({z'}_i^j) = e(\omega^r)] \\
&= R^2 \sum_{r=0}^{k-1} \omega^r \sum_c N_B[(p \cdot \bar{p})_{a,0c}(z_i^j, {z'}_i^j) = e(\omega^r)] \\
&= R^2 \sum_{r=0}^{k-1} \omega^r N_B[(p \cdot \bar{p})_{a,0}(z_i^j, {z'}_i^j) = e(\omega^r)] \\
&= R^2 (N_B[(p \cdot \bar{p})_{a,0}(z_i^j, {z'}_i^j) = 1] - N_B[(p \cdot \bar{p})_{a,0}(z_i^j, {z'}_i^j) = -1]).
\end{aligned}$$

Here the ${z'}_i^j$ are independent copies of the z_i^j variables; if $\bar{p}$ could use the same variables as p then BQP $\subseteq \Delta_2^p$ would follow—see discussion later in Section 5. Also $p_{a,0}$ means that the classical value 0 is substituted for b_1, i.e. for z_1^s, and $(p \cdot \bar{p})_{a,0}$ means that 0 is also substituted for ${z'}_1^s$. The last line follows because $\Pr(0)$ is a real number and so the values for other ω^r must cancel. Thus in fact we obtain a probability of acceptance by measuring a single qubit line as a difference of two #P functions, regardless of the phases of the intervening gates.

Representing the state of the remaining lines $2, \dots, n$ after such a measurement may be more cumbersome. If the measurement outcome is 0, then the remaining lines have state

$$\begin{aligned}
\sqrt{1/\Pr(0)} \sum_c \alpha_c \left|c\right\rangle &= \sqrt{1/\Pr(0)} \sum_c \left\langle a\right| C \left|0c\right\rangle \left|c\right\rangle \\
&= \sqrt{1/\Pr(0)} R \sum_c \sum_{\ell=0}^{k-1} \omega^\ell N_B[p_{a,0c}(z_i^j) = e(\omega^\ell)] \left|c\right\rangle \\
&= R' \sum_c \sum_{\ell=0}^{k-1} \omega^\ell N_B[p'_{a,c}(z_i^j) = e(\omega^\ell)] \left|c\right\rangle
\end{aligned}$$

where $p' = p[z_1^s := 0]$ and i runs from 2 to n. It does not seem possible or useful to try to simplify this further, but we do obtain for any $b' \in \{\, 0, 1 \,\}^{n-1}$:

$$\begin{aligned}
\langle \textstyle\sum_c \alpha_c c |\, b' \,\rangle &= \alpha_{b'} = \left\langle a\right| C \left|0b'\right\rangle \\
&= R \sum_{\ell=0}^{k-1} \omega^\ell N_B[p_{a,0b'}(z_i^j) = e(\omega^\ell)] \\
&= R \sum_{\ell=0}^{k-1} \omega^\ell N_B[p'_{a,c}(z_i^j) = e(\omega^\ell)]
\end{aligned}$$

This at least shows that the polynomial p' obtained by substituting the classical result of the measurement into p becomes the operative one, though one must still keep track of the difference between R and $R' = R/\sqrt{\Pr(0)}$.

2.3 Circuit manipulations

First we note a consequence of the bi-directional symmetry in the definitions of p_C and q_C, whereby no substitutions have been performed. $C*$ denotes the mirror image of C with the former outputs $b_1, \dots b_n$ now being designated the inputs $a_1, \dots, a_n$, and with each gate G reversed by substituting its adjoint G^*.

Corollary 2.3 *For every quantum circuit C, $p_{C^*} = p_C$ and $q_{C^*} = q_C$ (up to renaming of variables).*

Proof: Since the adjoint of a "bra" is a "ket," $\langle a|\ C^*\ |b\rangle = \langle b|\ C\ |a\rangle$, so we may picture the original C running right-to-left or with a and b interchanged. Since the construction in the proof of Theorem 2.1 is symmetrical for each gate until the substitution step, and the only substitution is to equate two variables, the resulting p_{C^*} is the same polynomial, up to interchanging the substituted variables and a with b (i.e., each z_i^0 with z_i^s). The same goes for q_C. □

Before treating tensor products, we need to emphasize points about equivalence of gate polynomials and finding distinguished representatives. Suppose we have a circuit C composed of two gates, say H (Hadamard) on line 1 followed by CNOT on $1, 2$ (meaning 1 is the control). Under the above notation, p_C and q_C employ interior variables z_1^1 and z_2^1, and while z_2^1 can be substituted by a_2 ($= z_2^0$) since the H gate does not involve line 2, no substitution is prescribed for z_1^1. Likewise [DHH$^+$04] introduces the same new variable. However, we can combine the gates into one by multiplying the matrices for $H \otimes I$ and CNOT, call this U. Then we obtain equivalent polynomials p_U and q_U that do not involve z_1^1.

Moreover, note that the CNOT gate offers the substitution $z_1^1 + a_2 - 2z_1^1 a_2$ on the second line, as we compute expressly for p_C in the next section. Were it followed by a third gate, we could use this expression in place of z_2^2. However the terms of our theorems do not allow such "zapping" of b_2—rather what happens in the proof is that b_2 gets substituted for the nominal variable "z_2^s" $= z_2^2$. As also observed in [DHH$^+$04], we can mitigate this issue by appending two more Hadamard gates on the second line. This produces an equivalent circuit C', but by both our rules and theirs, produces a variable z_2^4 that has no indicated substitution, and that can then be "cleanly" identified with b_2. (Note that our constructions above already eliminate the n equations collectively called "B" in [DHH$^+$04].) Moreover, the two extra Hadamard gates make the constant R' for C' equal to $R/2$. The polynomial $p_{C'}$ thus is overtly different from p_C, and gives different raw numbers of solutions, though they scale by R' to give the same amplitudes. Absent an immediate way to recognize the ostensible equivalence of such $p_C, p_U, p_{C'}$, we resort to verbs like "represents" and "can be taken as."

That said, our tensor-product theorem holds for any of the polynomial forms, after substitution as well as reduction. Given two quantum circuits C^1, C^2 on n_1 and n_2 qubit lines, respectively, $C^1 \otimes C^2$ is representable as a circuit on $n_1 + n_2$ lines that puts the gates of C_2 "below" those for C_1. Any merge of the ordered lists of gates in C_1, respectively C_2, is fine—or one may pair up gates into tensor products. Hence the need for our language about equivalence, in what is otherwise a short-and-sweet statement:

Theorem 2.4 *For any quantum circuits C^1 and C^2, $p_{C^1 \otimes C^2}$ can be taken as $p_{C^1} \cdot p_{C^2}$, and $q_{C^1 \otimes C^2}$ can be taken as $q_{C_1} + q_{C_2}$ (mod k).*

Proof: For any $a = a^1 \otimes a^2$ and $b = b^1 \otimes b^2$ on the same respective indices,

$$\begin{aligned}
\langle a|\; C^1 \otimes C^2 \;|b\rangle &= \langle a^1|\; C^1 \;|b^1\rangle \cdot \langle a^2|\; C^2 \;|b^2\rangle \\
&= \left(R_1 \sum_{\ell} \omega^{\ell} N_B[p^1_{a^1,b^1}(\vec{z}^1) = e(\omega^{\ell})]\right)\left(R_2 \sum_{m} \omega^{m} N_B[p^2_{a^2,b^2}(\vec{z}^2) = e(\omega^{m})]\right) \\
&= R_1R_2 \sum_{\ell,m} \omega^{\ell+m} N_B[p^1_{a^1,b^1}(\vec{z}^1) = e(\omega^{\ell})] \cdot N_B[p^2_{a^2,b^2}(\vec{z}^2) = e(\omega^{m})] \\
&= R_1R_2 \sum_{\lambda} \omega^{\lambda} N_B[p^1_{a^1,b^1}(\vec{z}^1) p^2_{a^2,b^2}(\vec{z}^2) = e(\omega^{\lambda})].
\end{aligned}$$

The last line follows because any 0-1 assignment $u = (u^1, u^2)$ that makes p^1p^2 evaluate to $e(\omega^{\lambda})$ must (given that the values of the polynomials are always units) make $p^1(u^1) = \omega^{\ell}$ and $p^2(u^2) = \omega^{m}$ such that $\ell + m = \lambda \pmod{k}$. By the same token, writing S_1, S_2 for the extra scaling factors in Theorem 2.2, we have

$$\begin{aligned}
\langle a|\; C^1 \otimes C^2 \;|b\rangle &= R_1S_1R_2S_2 \sum_{\ell,m} \omega^{\ell+m} N_B[q^1_{a^1,b^1}(\vec{z}^1, \vec{w}^1) = e(\omega^{\ell})] \cdot N_B[q^2_{a^2,b^2}(\vec{z}^2, \vec{w}^2) = e(\omega^{m})] \\
&= R_1R_2 \sum_{\lambda} \omega^{\lambda} N_B[q^1_{a^1,b^1}(\vec{z}^1, \vec{w}^1) + q^2_{a^2,b^2}(\vec{z}^2, \vec{w}^2) = e(\omega^{\lambda})]
\end{aligned}$$

because here $e(\omega^{\lambda}) = e(\omega^{\ell}) + e(\omega^{m}) \pmod{k}$, and the same 1-1 breakdown of solutions applies. $\square$

Indeed, when the circuits are single gates F and G, the polynomial $p_{F\otimes G}$ winds up being expressly defined as the product of the phase-indicator terms P_F and P_G. Multiplying out this product gives indicator factors of the form $I(c^1)I(c^2)I(d^1)I(d^2)$ that are formally identical to the post-tensoring factors $I(c)I(d)$, and the entries $e((F \otimes G)(c,d))$ are just $e(F(c^1,d^1))e(G(c^2,d^2))$. However, taking the product of q_F and q_G, while giving the correct indicator factors, fails because now we need $E((F \otimes G)(c,d)) = E(F(c^1,d^1)) + E(G(c^2,d^2)) \pmod{k}$. It is interesting that taking the sum yields correct results without yielding indicator factors of degree $n^1 + n^2$. This nice behavior is ultimately a feature of the algebra of indicator factors and reflects the way that the indices decompose. The degree savings also help computer algebra systems. Still, the lack of immediate correspondence to the directly-defined $q_{F\otimes G}$ is part of our general suspicion that certain algebraic properties of quantum circuits will emerge more readily from the multiplicative representations.

2.4 Controlled Gates

For any m-qubit gate G, C-G is a gate on $m + 1$ lines that behaves like G on the m "target" lines for the $\langle 1|$ value of the new "control" line, and behaves like the identity on all lines for the $\langle 0|$ value. It is always the identity on the control line. Letting y, z stand for the before- and after- variables on the control line, with the substitution $z := y$, we have the identities:

$$\begin{aligned}
P_{C\text{-}G} &= (1-y)(1-z)P_{I^{\otimes m}} + yzP_G \\
P'_{C\text{-}G} &= (1-y)P'_{I^{\otimes m}} + yP'_G \\
Q_{C\text{-}G} &= w[(1-y)z + y(1-z)] + yzQ_G \\
Q'_{C\text{-}G} &= yQ'_G
\end{aligned}$$

3 Gate Polynomials

We use the capitalized labels P, Q for the phase terms P_j, Q_j obtained for single gates, and P', Q' when the gates allow substitution for involved qubit lines. For all gates except the T-gate, P'' denotes P' over $\mathbf{Z}_2[u]$ where the adjoined element u is supposed only to satisfy $u^4 = 1$. The symbol $\mapsto$ denotes the further reduction of P' modulo the Boolean ideal. Overbars denote conjugate polynomials, when different. For an m-qubit gate, we rename the "z_i^{j-1}" variables on its input lines to $y_1, \dots, y_m$, and its output lines to $z_1, \dots, z_m$. For single-qubit gates, we just write y and z. All examples have dyadic phases, so k is always a power of 2.

3.1 Multiplicative polynomials P

In the multiplicative representations, we write simply 1 and -1 in place of $e(1)$ and $e(-1)$, and when the target ring R has two more fourth roots of unity, we name one of them i and the other $-i$, and so on. We begin with single-qubit gates.

Identity Gate

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\begin{aligned} P_I &= (1-y)(1-z) + yz = 2yz - y - z + 1 \\ P'_I &= P_I[z := y] = 2y^2 - 2y + 1 \\ &\mapsto 1 \\ P''_I &= 1 \end{aligned}$$

Not Gate—Pauli X

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{aligned} P_X &= (1-y)z + y(1-z) = y + z - 2yz \\ P'_X &= P_X[z := (1-y)] = 2y^2 - 2y + 1 \\ &\mapsto 1 \\ P''_X &= 1 \end{aligned}$$

Note that P' is the same for I and X even before the reduction—the substitution handles the entire difference between the two gates.

Pauli Y Gate—half phase

$$Y_0 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{aligned} P_{Y_0} &= y(1-z) - (1-y)z = (y - z) \\ P'_{Y_0} &= P_Y[z := (1-y)] = 2y - 1 \\ P''_{Y_0} &= y^2 + u^2(1-y)^2 = (1+u^2)y^2 - 2u^2y + u^2 \\ &\mapsto u^2 + u^2y + y \end{aligned}$$

Pauli Y Gate—standard phase

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

$$\begin{aligned}
P_Y &= iy(1-z) - i(1-y)z = i(y-z) \\
\bar{P}_Y &= i(z-y) \\
P'_Y &= P_Y[z := (1-y)] = i(2y-1) \\
\bar{P}'_Y &= i(1-2y) \\
P''_Y &= uy(1-z) + u^3(1-y)z[z := 1-y] \\
&= uy^2 + u^3(1-y)^2 = (u+u^3)y^2 - 2u^3y + u^3 \\
&\mapsto u^3 + u^3y + uy \\
\bar{P}''_Y &= u + u^3y + uy
\end{aligned}$$

Note that for P'' we are not allowed to assume $u + u^3 = 0$. Compared to Y_0, the global phase multiplies the polynomials by i and u, respectively. The multiplier of u can be pulled all the way outside the circuit polynomial, but still shows up in the value of $\langle a| \, C \, |b\rangle$, so we do not ignore it.

Pauli Z Gate

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$\begin{aligned}
P_Z &= (1-y)(1-z) - yz = 1 - y - z \\
P'_Z &= P_Z[z := y] = 1 - 2y \\
P''_Z &= (1-y)^2 + u^2y^2 = 1 - 2y + (1+u^2)y^2 \\
&\mapsto 1 + y + u^2y
\end{aligned}$$

With $u^4 = 1$, note that $P''_Z = u^2 P''_{Y_0}$ even though the gates are not phase translates of each other—the nub is that they use different subsitutions. Also note that one cannot assume $u^2 = -1$ when defining P''_Z, since with coefficients modulo 2 that would make $u^2 = 1$, violating injectivity of the embedding.

Phase Gate

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$

$$\begin{aligned}
P_S &= (1-y)(1-z) + iyz = (1+i)yz - y - z + 1 \\
\bar{P}_S &= (1-i)yz - y - z + 1 \\
P'_S &= P_S[z := y] = (1+i)y^2 - 2y + 1
\end{aligned}$$

$$\bar{P}'_S = (1-i)y^2 - 2y + 1$$

$$\begin{aligned}
&\mapsto iy - y + 1 = \sqrt{2}\omega^3 y + 1 \\
P''_S &= (1-y)^2 + y^2u = 1 - 2y + y^2(1+u) \\
&\mapsto 1 + y(1+u) \\
\bar{P}''_S &= 1 + y + yu^3
\end{aligned}$$

T **Gate** With $\omega = e^{\pi i/4} = +\sqrt{i}$,

$$T = \begin{bmatrix} 1 & 0 \\ 0 & \omega \end{bmatrix}$$

$$\begin{aligned}
P_T &= (1-y)(1-z) + \omega yz = (1+\omega)yz - y - z + 1 \\
\bar{P}_T &= (1+\omega^7)yz - y - z + 1 \\
P'_T &= P_T[z := y] = (1+\omega)y^2 - 2y + 1 \\
\bar{P}'_T &= (1+\omega^7)y^2 - 2y + 1 \\
&\mapsto \omega y - y + 1 \\
P''_T &= P'_T/\text{n.a.}
\end{aligned}$$

Hadamard Gate

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

$$\begin{aligned}
P_H &= (1-y)(1-z) + (1-y)z + y(1-z) - yz = 1 - 2yz \\
P'_H &= P_H \\
P''_H &= 1 - yz + yzu^2 \qquad \text{(no substitution)}
\end{aligned}$$

$\sqrt{\text{NOT}}$ **Gate**

$$V = \frac{1}{2} \begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} \omega & \omega^{-1} \\ \omega^{-1} & \omega \end{bmatrix}$$

$$\begin{aligned}
P_V &= (1-y)(1-z)\omega + (1-y)z\omega^{-1} + y(1-z)\omega{-}1 + yz\omega \\
&= (1-y-z+2yz)\omega + (y+z-2yz)\omega^{-1} \\
&= \omega - y(\omega^3+\omega) - z(\omega^3+\omega) + 2yz(\omega^3+\omega) \\
&= \omega - (y+z)i\sqrt{2} + 2yzi\sqrt{2} \\
\bar{P}_V &= \omega^7 + (y+z)i\sqrt{2} - 2yzi\sqrt{2} \\
P'_V &= P_V
\end{aligned}$$

It is also OK to do the calculation without first dividing out by $\sqrt{2}$, provided one remembers it at the end. Thus

$$\begin{aligned}
P_V &= [(1-y)(1-z)(1+i) + (1-y)z(1-i) + y(1-z)(1-i) + yz(1+i)]/sqrt2 \\
&= [1 + i - 2iy - 2iz + 4iyz]/\sqrt{2}
\end{aligned}$$

which agrees with the above. However, it is not allowed to create P''_V over $\mathbf{Z}_2[u]$ by cancelling the corresponding terms $4uyz$, $-2uy$, and $-2uz$.

CNOT **Gate**

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\begin{aligned}
P_{\mathrm{CNOT}} &= (1-y_1)(1-z_1)P_I(y_2,z_2) + y_1 z_1 P_{\mathrm{NOT}}(y_2,z_2) \\
&= -2y_1y_2z_2 + y_1y_2 + y_1z_1 + y_1z_2 - y_1 - 2y_2z_1z_2 + y_2z_1 + 2y_2z_2 - y_2 \\
&\quad +z_1z_2 - z_1 - z_2 + 1 \\
P'_{\mathrm{CNOT}} &= P_{\mathrm{CNOT}}[z_1 := y_1,\ z_2 := y_1 + y_2 - 2y_1y_2] \\
&= 8y_1^2y_2^2 - 8y_1^2y_2 + 3y_1^2 - 8y_1y_2^2 + 8y_1y_2 - 3y_1 + 2y_2^2 - 2y_2 + 1 \\
&\mapsto 1 \\
P''_{\mathrm{CNOT}} &= 1
\end{aligned}$$

Alternately, $P_{\mathrm{CNOT}} = (1-y_1)(1-y_2)(1-z_1)(1-z_2) + (1-y_1)y_2(1-z_1)z_2 + y_1(1-y_2)z_1z_2 + y_1y_2z_1(1-z_2)$ and $P'_{\mathrm{CNOT}} = (1-y_1)P_I(y_2) + y_1P'_{\mathrm{NOT}}(y_2) = 1 - y_1 + y_1 = 1$. If one substitutes for z_1 but not for z_2, one obtains

$$P_{\mathrm{CNOT}}[z_1 := y_1] \mapsto 1 - y_1 - y_2 - z_2 + 2y_1y_2 + 2y_1z_2 + 2y_2z_2 - 4y_1y_2z_2.$$

Controlled-Z Gate

$$\mathrm{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

$$\begin{aligned}
P_{CZ} &= (1-y_1)(1-z_1)P_I(y_2,z_2) + y_1z_1P_Z(y_2,z_2) \\
&= 2y_1y_2z_1z_2 - 2y_1y_2z_1 - 2y_1y_2z_2 - 2y_1z_1z_2 - 2y_2z_1z_2 + y_1y_2 + 2y_1z_1 \\
&\quad +y_2z_1 + y_1z_2 + 2y_2z_2 + z_1z_2 - y_1 - y_2 - z_1 - z_2 + 1 \\
P'_{CZ} &= P_{CZ}[z_1 := y_1,\ z_2 := y_2] \\
&= 2y_1^2y_2^2 - 4y_1^2y_2 - 4y_1y_2^2 + 2y_1^2 + 4y_1y_2 + 2y_2^2 - 2y_1 - 2y_2 + 1 \\
&\mapsto 1 - 2y_1y_2 \\
P''_{CZ} &= 1 - y_1y_2 + u^2y_1y_2
\end{aligned}$$

The polynomial P_{CZ} is invariant under swapping y_1, z_1 with y_2, z_2 respectively. The other two polynomials more obviously reflect the indifference under which qubit line is the "control" and which the "target."

Swap Gate

$$\mathrm{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned}
P_{\mathrm{SWAP}} &= (1-y_1)(1-y_2)(1-z_1)(1-z_2) + (1-y_1)y_2z_1(1-z_2) + y_1(1-y_2)(1-z_1)z_2 \\
&\quad +y_1y_2z_1z_2 \\
&= 4y_1y_2z_1z_2 - 2y_1y_2z_1 - 2y_1y_2z_2 + y_1y_2 - 2y_1z_1z_2 + y_1z_1 \\
&\quad +2y_1z_2 - y_1 - 2y_2z_1z_2 + 2y_2z_1 + y_2z_2 - y_2 + z_1z_2 - z_1 - z_2 + 1 \\
P'_{\mathrm{SWAP}} &= P_{\mathrm{SWAP}}[z_1 := y_2,\ z_2 := y_1] \\
&= 4y_1^2y_2^2 - 4y_1^2y_2 + 2y_1^2 - 4y_1y_2^2 + 4y_1y_2 - 2y_1 + 2y_2^2 - 2y_2 + 1 \\
&\mapsto 1 \\
P''_{\mathrm{SWAP}} &= 1
\end{aligned}$$

Shor Gate

$$\textsc{Shor} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \omega \end{bmatrix}$$

where $\omega = e^{i\pi/2^{k-j}}$ with the gate acting on qubits j and k of a designated quantum register.

$$\begin{aligned}
P_{\text{SHOR}} &= (1-y_1)(1-y_2)(1-z_1)(1-z_2) + (1-y_1)y_2(1-z_1)z_2 + y_1(1-y_2)z_1(1=z_2) \\
&\quad + y_1y_2z_1z_2\omega \\
&= \omega y_1y_2z_1z_2 + 3y_1y_2z_1z_2 - 2y_1y_2z_1 - 2y_1y_2z_2 + y_1y_2 - 2y_1z_1z_2 \\
&\quad + 2y_1z_1 + y_1z_2 - y_1 - 2y_2z_1z_2 + y_2z_1 + 2y_2z_2 - y_2 + z_1z_2 - z_1 - z_2 + 1 \\
P'_{\text{SHOR}} &= P_{\text{SHOR}}[z_1 := y_1,\ z_2 := y_2] \\
&= \omega y_1^2y_2^2 + 3y_1^2y_2^2 - 4y_1^2y_2 + 2y_1^2 - 4y_1y_2^2 + 4y_1y_2 - 2y_1 + 2y_2^2 - 2y_2 + 1 \\
&\mapsto \omega y_1y_2 - y_1y_2 + 1 \\
P''_{\text{SHOR}} &= P'_{\text{SHOR}}/\text{n.a.}
\end{aligned}$$

The conjugates are obtained by conjugating ω. Note that when $k-j$ approaches n, exponentially many tiny phases are summed over in (2.1), removing all question of its becoming a polynomial-sized formula. Shor's algorithm proper aprpoximates the effect of the tiny phases via standard gates. We wonder whether translating the proper circuits into polynomials will involve effects shown for the V-gate above, but where the magnitudes of (changes to) the normalization constants themselves become an issue.

A Gate

$$A = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & -1 & 1 & 0 \\ 1 & 1 & 0 & -1 \end{bmatrix}$$

$$\begin{aligned}
P_A &= 2y_1y_2z_1z_2 - 4y_1y_2z_1 + y_1y_2 - 2y_1z_1z_2 + 2y_1z_1 - y_1z_2 - 4y_2z_1z_2 \\
&\quad + 3y_2z_1 + 2y_2z_2 - y_2 + 3z_1z_2 - 2z_1 - z_2 + 1 \\
P'_A &= P_A \\
P''_A &= y_1y_2 + y_1z_2 + y_2z_1 + z_1z_2 + y_2 + z_2 + 1
\end{aligned}$$

In contrast to P''_V, it is OK to cancel the multiples of 2 in the coefficients because the $\sqrt{3}$ factor does not interact with the normalization of individual entries to be units.

Toffoli Gate

$$\textsc{Tof} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$
\begin{aligned}
P_{\mathrm{TOF}} &= (1-y_1)(1-z_1)P_{I^{\otimes 2}} + y_1 z_1 P_{\mathrm{CNOT}}(y_2,y_3,z_2,z_3)\\
&= 4y_1y_2y_3z_1z_2z_3 - 2y_1y_2y_3z_1z_2 - 4y_1y_2y_3z_1z_3\\
&\quad +2y_1y_2y_3z_1 - 4y_1y_2y_3z_2z_3 + 2y_1y_2y_3z_2 + 2y_1y_2y_3z_3\\
&\quad -y_1y_2y_3 - 2y_1y_2z_1z_2z_3 + 3y_1y_2z_1z_2 + 2y_1y_2z_1z_3 - 2y_1y_2z_1\\
&\quad +2y_1y_2z_2z_3 - 2y_1y_2z_2 - y_1y_2z_3 + y_1y_2 - 4y_1y_3z_1z_2z_3\\
&\quad +2y_1y_3z_1z_2 + 4y_1y_3z_1z_3 - 2y_1y_3z_1 + 2y_1y_3z_2z_3 - y_1y_3z_2 - 2y_1y_3z_3\\
&\quad +y_1y_3 + 2y_1z_1z_2z_3 - 2y_1z_1z_2 - 2y_1z_1z_3 + 2y_1z_1 - y_1z_2z_3\\
&\quad +y_1z_2 + y_1z_3 - y_1 - 4y_2y_3z_1z_2z_3 + 2y_2y_3z_1z_2 + 2y_2y_3z_1z_3\\
&\quad -y_2y_3z_1 + 4y_2y_3z_2z_3 - 2y_2y_3z_2 - 2y_2y_3z_3 + y_2y_3 + 2y_2z_1z_2z_3\\
&\quad -2y_2z_1z_2 - y_2z_1z_3 + y_2z_1 - 2y_2z_2z_3 + 2y_2z_2 + y_2z_3 - y_2 + 2y_3z_1z_2z_3\\
&\quad -y_3z_1z_2 - 2y_3z_1z_3 + y_3z_1 - 2y_3z_2z_3 + y_3z_2 + 2y_3z_3 - y_3 - z_1z_2z_3\\
&\quad +z_1z_2 + z_1z_3 - z_1 + z_2z_3 - z_2 - z_3 + 1\\
P'_{\mathrm{TOF}} &= (1-y_1) + y_1 P'_{\mathrm{CNOT}}(y_2,y_3,z_2,z_3) = 1\\
P''_{\mathrm{TOF}} &= P_{\mathrm{TOF}}[z_1 := y_1,\ z_2 := y_2\ z_3 := y_1y_2 + y_3] = 1
\end{aligned}
$$

Writing $P'_{\mathrm{TOF}} = P_{\mathrm{TOF}}[z_1 := y_1,\ z_2 := y_2,\ z_3 := y_1y_2 + y_3 - 2y_1y_2y_3]$ reduces to the above, and importantly, shows the required substitutions. Note that even over $\mathbf{Z}_2[u]$, the substitution for z_3 is non-linear.

3.2 Additive representations

Rather than take the minimum k for a particular gate, we leave k general, thus writing $k/2$ for the additively-embedded value $e(-1)$, rather than 1 assuming $k = 2$. Likewise we write $k/4$ for $e(i)$ and $k/8$ for $e(\sqrt{i})$.

Recall w stands for the new variable for each gate. Q' stands for the substitution version of Q, while Q'' is Q or Q' for $k = 4$ *provided* it is invariant under its arguments being 2 versus 0, and 3 versus 1. Conjugation of the *value* of Q'' is still an issue, so conjugates for all three polynomials are shown when they differ from the respective originals.

Identity Gate

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$
\begin{aligned}
Q_I &= (1-y)zw + y(1-z)w = w(y + z - 2yz)\\
Q'_I &= Q_I[z := y] = w(2y - 2y^2)\\
&\mapsto 0\\
Q''_I &= 0
\end{aligned}
$$

To justify the last line, we need to note that $2y - 2y^2$ is 0 modulo 4 not only when $y = 0, 1$ but also when $y = 2, 3$.

Not Gate—Pauli X

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{aligned}
Q_X &= (1-y)(1-z)w + yzw = w(1-y-z+2yz)\\
Q'_X &= Q_X[z := (1-y)] = w(2y-2y^2)\\
&\mapsto 0\\
Q''_X &= 0
\end{aligned}$$

Again the substitution handles the entire difference between I and X.

Pauli Y Gate—half phase

$$Y_0 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{aligned}
Q_{Y_0} &= (1-y)(1-z)w + (1-y)z(k/2) + yzw\\
&= w(1-y-z+2yz) + zk/2 - yzk/2\\
Q'_{Y_0} &= Q_{Y_0}[z := (1-y)] = 0w + k/2 - yk - y^2k/2\\
&\mapsto k/2 - (k/2)y\\
&= 2y+2 \pmod 4\\
&= 4y+4 \pmod 8\\
Q''_{Y_0} &= 2y+2
\end{aligned}$$

The substitution $z := (1-y)$ leaves w multiplied by $2y-2y^2$, which as we have seen is 0 mod 4 even when $y=2$ or $y=3$.

Pauli Y Gate

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

$$\begin{aligned}
Q_Y &= (1-y)(1-z)w + (1-y)z(3k/4) + y(1-z)(k/4) + yzw\\
&= w(1-y-z+2yz) + (3k/4)z + (k/4)y\\
\bar{Q}_Y &= w(1-y-z+2yz) + (k/4)z + (3k/4)y\\
Q'_Y &= Q_Y[z := (1-y)] = 2w(y-y^2) + (3k/4) - (k/2)y\\
&\mapsto 3k/4 - (k/2)y = (k/4) + Q'_{Y_0}\\
&= 2y+3 \pmod 4\\
&= 4y+6 \pmod 8\\
\bar{Q}'_Y &= k/4 - (k/2)y = k/4 + (k/2)y\\
Q''_Y &= 2y+3\\
\bar{Q}''_Y &= 2y+1
\end{aligned}$$

As expected, the extra factor of i corresponds to adding 1 to the representations modulo 4, or adding $k/4$ in general.

Pauli Z Gate

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$\begin{aligned}
Q_Z &= (1-y)zw + y(1-z)w + yz(k/2)\\
&= w(y+z-2yz) + (k/2)yz\\
Q'_Z &= Q_Z[z := y] = w(2y-2y^2) + (k/2)y^2\\
&\mapsto (k/2)y\\
&= y \pmod 2\\
&= 2y \pmod 4\\
&= 4y \pmod 8\\
Q''_Z &= 2y
\end{aligned}$$

Phase Gate

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$

$$\begin{aligned}
Q_S &= (1-y)zw + y(1-z)w + yz(k/4)\\
&= w(y+z-2yz) + (k/4)yz\\
\bar{Q}_S &= w(y+z-2yz) + (3k/4)yz\\
Q'_S &= Q_S[z := y] = w(2y-2y^2) + (k/4)y^2\\
&\mapsto (k/4)y\\
&= y \pmod 4\\
&= 2y \pmod 8\\
\bar{Q}'{}_S &= (3k/4)y\\
Q''_S &= y^2\\
\bar{Q}''{}_S &= 3y^2
\end{aligned}$$

Note that for Q''_S we do *not* reduce modulo the Boolean ideal, because we envision values $y = 2, 3$ as well as $0, 1$ modulo 4. That $2^2 = 0$ and $3^2 = 1$ modulo 4 is the point.

T **Gate** With $\omega = e^{\pi i/4} = +\sqrt{i}$,

$$T = \begin{bmatrix} 1 & 0 \\ 0 & \omega \end{bmatrix}$$

$$\begin{aligned}
Q_T &= (1-y)zw + y(1-z)w + yz(k/8)\\
&= w(y+z-2yz) + (k/8)yz\\
\bar{Q}_T &= w(y+z-2yz) + (7k/8)yz\\
Q'_T &= Q_T[z := y] = w(2y-2y^2) + (k/8)y^2\\
&\mapsto (k/8)y\\
&= y \pmod 8\\
\bar{Q}'{}_T &= (7k/8)y\\
Q''_T &= \text{n.a.?}
\end{aligned}$$

The odd values all square to 1 modulo 8, but the even values 2 and 6 square to 4 rather than 0. It is not clear whether this promotes or inhibits applications for circuits with T-gates and $k = 8$ similar to what we show for stabilizer circuits with $k = 4$.

Hadamard Gate

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

$$\begin{aligned} Q_H &= yz(k/2) \\ Q'_H &= \text{n.a.} \\ Q''_H &= 2yz \end{aligned}$$

Even though we do not have a substitution, the coefficient of 2 makes Q''_H depend only on the parity of its arguments modulo 4.

$\sqrt{\text{NOT}}$ Gate

$$V = \frac{1}{2} \begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} \omega & \omega^{-1} \\ \omega^{-1} & \omega \end{bmatrix}$$

$$\begin{aligned} Q_V &= (k/8)[(1-y)(1-z) + yz] + (7k/8)[(1-y)z + y(1-z)] \\ &= (k/8)[1 - y - z + 2yz + 7z + 7y - 14yz] = (k/8)[1 + 6y + 6z + 12yz] \\ &= \frac{k}{8} + \frac{3k}{4}(y+z) - \frac{k}{2}yz \\ \bar{Q}_V &= \frac{7k}{8} + \frac{k}{4}(y+z) - \frac{k}{2}yz \\ Q'_V &= \text{n.a.} \\ Q''_V &= \text{n.a.} \end{aligned}$$

CNOT Gate

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\begin{aligned} Q_{\text{CNOT}} &= w[2y_1y_2z_2 - y_1y_2 - y_1z_1 - y_1z_2 + y_1 + 2y_2z_1z_2 - y_2z_1 - 2y_2z_2 \\ &\quad + y_2 - z_1z_2 + z_1 + z_2] \\ Q'_{\text{CNOT}} &= Q_{\text{CNOT}}[z_1 := y_1,\ z_2 := y_1 + y_2 - 2y_1y_2] \\ &= w[-8y_1^2y_2^2 + 8y_1^2y_2 - 3y_1^2 + 8y_1y_2^2 - 8y_1y_2 + 3y_1 - 2y_2^2 + 2y_2] \\ &\mapsto 0 \\ Q''_{\text{CNOT}} &= 0 \end{aligned}$$

Besides "substituting away" the CNOT gate, it may be important to compare the two substitutions that result from $z_2 := y_1 + y_2$ versus $z_2 := y_1 + y_2 - 2y_1y_2$, modulo 4. Namely:

$$\begin{aligned} Q_{\text{CNOT}}[z_1 := y_1,\ z_2 := y_1 + y_2] &= w[4y_1^2y_2 - 3y_1^2 + 4y_1y_2^2 - 6y_1y_2 + 3y_1 - 2y_2^2 + 2y_2] \\ &= w[-3y_1^2 - 6y_1y_2 + 3y_1 + 2y_2^2 + 2y_2] \\ &= w[y_1^2 - y_1 + 2y_1y_2]; \\ Q'_{\text{CNOT}} &= w[-3y_1^2 + 3y_1 + 2y_2^2 + 2y_2] \\ &= w[y_1^2 - y_1] \end{aligned}$$

That these terms are multiplied by w may make the difference immaterial. Also, substituting just the first variable gives

$$Q_{\mathrm{CNOT}}[z_1 := y_1] \mapsto w[2y_1y_2z_2 - 2y_1y_2 - 2y_1z_2 - 2y_2z_2 + y_1 + y_2 + z_2].$$

Controlled-Z Gate

$$\mathrm{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

$$\begin{aligned}
Q_{CZ} &= w[1 - ((1-y_1)(1-y_2)(1-z_1)(1-z_2) + (1-y_1)y_2(1-z_1)z_2 + y_1(1-y_2)z_1(1-z_2) \\
&\quad + y_1y_2z_1z_2)] + (k/2)y_1y_2z_1z_2 \\
&= (k/2)y_1y_2z_1z_2 - 4y_1y_2z_1z_2w + 2y_1y_2z_1w + 2y_1y_2z_2w \\
&\quad + 2y_1z_1z_2w + 2y_2z_1z_2w - y_1y_2w - 2y_1z_1w - y_2z_1w - y_1z_2w \\
&\quad - 2y_2z_2w - z_1z_2w + y_1w + y_2w + z_1w + z_2w \\
Q'_{CZ} &= Q_{CZ}[z_1 := y_1,\ z_2 := y_2] \\
&= w[-4y_1^2y_2^2 + 4y_1^2y_2 + 4y_1y_2^2 - 2y_1^2 - 4y_1y_2 - 2y_2^2 + 2y_1 + 2y_2] \\
&\quad + (k/2)y_1^2y_2^2 \\
&\mapsto (k/2)y_1y_2 \\
Q''_{CZ} &= 2y_1y_2
\end{aligned}$$

As with the other stabilizer gates, Q''_{CZ} does not care whether an argument is 0 vs. 2, or 1 vs. 3.

Swap Gate

$$\mathrm{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned}
Q_{\mathrm{SWAP}} &= w[-4y_1y_2z_1z_2 + 2y_1y_2z_1 + 2y_1y_2z_2 - y_1y_2 + 2y_1z_1z_2 \\
&\quad - y_1z_1 - 2y_1z_2 + y_1 + 2y_2z_1z_2 - 2y_2z_1 - y_2z_2 + y_2 - z_1z_2 + z_1 + z_2] \\
Q'_{\mathrm{SWAP}} &= Q_{\mathrm{SWAP}}[z_1 := y_2,\ z_2 := y_1] \\
&= w[-4y_1^2y_2^2 + 4y_1^2y_2 - 2y_1^2 + 4y_1y_2^2 - 4y_1y_2 + 2y_1 - 2y_2^2 + 2y_2] \\
&\mapsto 0 \\
Q''_{\mathrm{SWAP}} &= w[2y_1^2 + 2y_1 + 2y_2^2 + 2y_2] = 0
\end{aligned}$$

Shor Gate

$$\textsc{Shor} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \omega \end{bmatrix}$$

where $\omega = e^{i\pi/2^{\ell-j}}$ with the gate acting on qubits j and ℓ of a designated quantum register. Here we assume $k = 2^{n-1}$ and put $a = k/2^{\ell-j}$ so that $e_+(\omega) = k/a$.

$$\begin{aligned}
Q_{\text{SHOR}} &= w[-4y_1y_2z_1z_2 + 2y_1y_2z_1 + 2y_1y_2z_2 - y_1y_2 + 2y_1z_1z_2 - 2y_1z_1 \\
&\quad -y_1z_2 + y_1 + 2y_2z_1z_2 - y_2z_1 - 2y_2z_2 + y_2 - z_1z_2 + z_1 + z_2] + (k/a)y_1y_2z_1z_2 \\
Q'_{\text{SHOR}} &= Q_{\text{SHOR}}[z_1 := y_1,\ z_2 := y_2] \\
&= w[-4y_1^2y_2^2 + 4y_1^2y_2 - 2y_1^2 + 4y_1y_2^2 - 4y_1y_2 + 2y_1 - 2y_2^2 + 2y_2] + (k/a)y_1^2y_2^2 \\
&\mapsto (k/a)y_1y_2 \\
Q''_{\text{SHOR}} &= \text{n.a.}
\end{aligned}$$

Conjugates replace (k/a) by $k(1 - 1/a)$.

A Gate

$$A = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & -1 & 1 & 0 \\ 1 & 1 & 0 & -1 \end{bmatrix}$$

$$\begin{aligned}
Q_A &= w[(1-y_1)y_2(1-z_1)(1-z_2) + (1-y_1)(1-y_2)(1-z_1)z_2 \\
&\quad +y_1y_2z_1(1-z_2) + y_1(1-y_2)z_1z_2] \\
&\quad +(k/2)[(1-y_1)(1-y_2)z_1(1-z_2) + y_1(1-y_2)(1-z_1)z_2 + y_1y_2z_1z_2] \\
Q'_A &= Q_A \\
Q''_A &= \text{n.a.}
\end{aligned}$$

Modulo 4, the non-'w' portion of Q_A multiplies out to

$$\begin{aligned}
&6y_1y_2z_1z_2 - 2y_1y_2z_2 - 4y_1z_1z_2 + 2y_1z_2 - 2y_2z_1z_2 + 2z_1z_2 \\
=\ &2y_1y_2z_1z_2 - 2y_1y_2z_2 + 2y_1z_2 - 2y_2z_1z_2 + 2z_1z_2
\end{aligned}$$

which owing to the factors of 2 is odd/even invariant, but the portion multiplied by w is

$$\begin{aligned}
&-4y_1y_2z_1z_2 + 2y_1y_2z_1 + 2y_1y_2z_2 - y_1y_2 + 2y_1z_1z_2 - y_1z_2 + 2y_2z_1z_2 - y_2z_1 - 2y_2z_2 \\
&+y_2 - z_1z_2 + z_2 \\
=\ &2y_1y_2z_1 + 2y_1y_2z_2 + 2y_1z_1z_2 + 2y_2z_1z_2 - 2y_2z_2 \\
&+y_2 + z_2 - y_1y_2 - y_1z_2 - y_2z_1 - z_1z_2
\end{aligned}$$

The last line is not odd/even invariant, but the fact that it multiplies w may make this immaterial.

Toffoli Gate

$$\textsc{Tof} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$
\begin{aligned}
Q_{\mathrm{TOF}} &= w[-3y_1y_2y_3z_1z_2z_3 + y_1y_2y_3z_1z_2 + 3y_1y_2y_3z_1z_3 - y_1y_2y_3z_1 \\
&\quad +3y_1y_2y_3z_2z_3 - y_1y_2y_3z_2 - y_1y_2y_3z_3 + y_1y_2y_3 + y_1y_2z_1z_2z_3 \\
&\quad -2y_1y_2z_1z_2 - y_1y_2z_1z_3 + y_1y_2z_1 - y_1y_2z_2z_3 + y_1y_2z_2 - y_1y_2 \\
&\quad +3y_1y_3z_1z_2z_3 - y_1y_3z_1z_2 - 3y_1y_3z_1z_3 + y_1y_3z_1 - y_1y_3z_2z_3 \\
&\quad +y_1y_3z_3 - y_1y_3 - y_1z_1z_2z_3 + y_1z_1z_2 + y_1z_1z_3 - y_1z_1 + y_1 \\
&\quad +3y_2y_3z_1z_2z_3 - y_2y_3z_1z_2 - y_2y_3z_1z_3 - 3y_2y_3z_2z_3 + y_2y_3z_2 \\
&\quad +y_2y_3z_3 - y_2y_3 - y_2z_1z_2z_3 + y_2z_1z_2 + y_2z_2z_3 - y_2z_2 + y_2 - y_3z_1z_2z_3 \\
&\quad +y_3z_1z_3 + y_3z_2z_3 - y_3z_3 + y_3] \\
Q'_{\mathrm{TOF}} &= Q_{\mathrm{TOF}}[z_1 := y_1,\ z_2 := y_2,\ z_3 := y_1y_2 + y_3 - 2y_1y_2y_3] \\
&= w[6y_1^3y_2^3y_3^2 - 5y_1^3y_2^3y_3 + y_1^3y_2^3 - 12y_1^3y_2^2y_3^2 + 10y_1^3y_2^2y_3 \\
&\quad -2y_1^3y_2^2 + 6y_1^3y_2y_3^2 - 5y_1^3y_2y_3 + y_1^3y_2 - 12y_1^2y_2^3y_3^2 \\
&\quad +10y_1^2y_2^3y_3 - 2y_1^2y_2^3 + 5y_1^2y_2^2y_3^2 - 2y_1^2y_2^2y_3 - 2y_1^2y_2^2 \\
&\quad +2y_1^2y_2y_3^2 - 2y_1^2y_2y_3 + 2y_1^2y_2 - 3y_1^2y_3^2 + 2y_1^2y_3 - y_1^2 \\
&\quad +6y_1y_2^3y_3^2 - 5y_1y_2^3y_3 + y_1y_2^3 + 2y_1y_2^2y_3^2 - 2y_1y_2^2y_3 \\
&\quad +2y_1y_2^2 - 2y_1y_2y_3^2 - y_1y_2 + 2y_1y_3^2 - y_1y_3 + y_1 - 3y_2^2y_3^2 \\
&\quad +2y_2^2y_3 - y_2^2 + 2y_2y_3^2 - y_2y_3 + y_2 - y_3^2 + y_3] \\
&\mapsto 0 \\
Q''_{\mathrm{TOF}} &= \text{n.a.?}
\end{aligned}
$$

As with Q''_{CNOT}, the polynomial multiplied by w does not vanish modulo 4 for all arguments in $\{\ 0, 1, 2, 3\ \}$, but this fact may be immaterial. Of more import is that the substitution $z_3 := y_1y_2 + y_3 - 2y_1y_2y_3$ does not have the same parity as any linear substitution.

4 Circuit Simulations and Equivalence

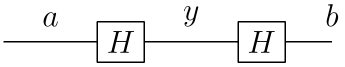
A *classical annotation* of a quantum circuit is an assignment of occurrences of variables or terms to the wires of the circuit diagram. The *minimum annotation* is the assignment z_i^j for $1 \leq i \leq n$ and $0 \leq j \leq s$, possibly renaming z_i^0 to a_i and z_i^s to b_i. All other *legal* annotations are obtained from the minimum one by substituting a term t for some z_i^j, $1 \leq j \leq s-1$ in a way allowed by the gate immediately to its *left*. When no gate is there, the single-qubit identity gate is assumed. Note that we do not allow substitution for b_i—instead we prescribe inserting an extra identity gate. (Dawson et al. suggest inserting two consecutive Hadamard gates for a similar purpose.) There is always a unique *maximum annotation* which applies all legal substitutions.

Given a representation scheme, meaning choice of coefficient ring and additive-or-multiplicative, every annotation A gives rise a unique phase polynomial P_A (or additively, Q_A) obtained via the prescribed terms for each gate, and further a unique reduction P'_A of P_A (or Q'_A of Q_A) modulo the Boolean ideal. In the catalog above, the gate polynomials P give P_A for the minimum annotation A, while P' give $P'_{A'}$ for the maximum A', and similarly for Q, Q'.

Definition. Two polynomials are *equivalent* if they arise from annotations of two equivalent quantum circuits.

The polynomials must have the same variables $a_1, \dots, a_n$ and $b_1, \dots, b_n$ (or their z_i^0, z_i^s namings), but there is no requirement on any other variables, which we call *interior variables*. It is interesting to ask whether there is a simple algebraic characterization of this equivalence relation. Now we give some examples.

Two consecutive Hadamard gates on the same qubit line give the identity, but show some subtleties of the polynomials.



The multiplicative polynomial is

$$P_C = (1 - 2ay)(1 - 2yb) \mapsto 1 - 2ay - 2yb + 4ayb,$$

with a background factor of $R = 1/2$ from the two Hadamard gates. Why is this equivalent to the identity-gate polynomial? The latter is

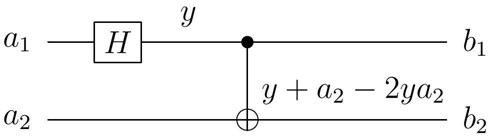
$$P_I = 1 - a - b + 2ab.$$

A clue is to look at what happens to P_C under the "illegal" substitution $b = 1 - a$, when it becomes

$$1 - 2ay - 2y + 2ay + 4ay - 4a^2y \mapsto 1 - 2y.$$

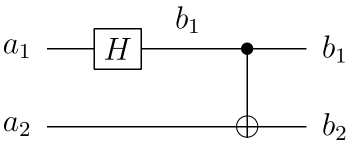
A multiplicative term $(1-2y)$ where this is the only occurrence of the interior variable y behaves much like "w" in the additive representations. It sets up a 1-1 correspondence between solutions for each $e(\omega)$ and $e(-\omega)$, which cancels everything to zero. Thus all assignments into P_C that make $a \neq b$ contribute a net of zero to the complex amplitude. Substituting $a = b$ makes both polynomials reduce to 1. Interestingly, substituting $y = 1/2$ into P_C yields P_I, but this is not a "legal" substitution.

We diagram the next circuit in a way that makes the distinguished role of the a_i and b_i variables clearer:



$$P_A = (1 - 2a_1y)(1 - y - b_1 + 2yb_1)(1 - y - a_2 + 2ya_2 - b_2 + 2yb_2 + 2a_2b_2 - 4ya_2b_2)$$

Here we have implemented a pair of I-gates at the far right, but not shown them in the diagram. It is legal to substitute b_1 for y, but not to wipe out b_2. We may, however, dispense with the substitution for the second CNOT line and annotate this way:



$$P = (1 - 2a_1b_1)(1 - a_2 - b_1 - b_2 + 2b_1b_2 + 2a_2b_2 + 2a_2b_1 - 4a_2b_1b_2)$$

Note that in the second term, we substituted b_1 for both y_1 and z_1 in P_{CNOT} as given in the catalog, then reduced modulo the Boolean ideal—but we did not use P'_{CNOT} because we did not substitute on the second qubit line. To execute the circuit on input $\langle 00|$, now substitute $a_1 = 0$ and $a_2 = 0$ to get

$$(1 - b_1 - b_2 + 2b_1b_2)$$

This gives 1 when $b_2 = b_1$, but 0 otherwise. Hence inner product with $|01\rangle$ and $|10\rangle$ give 0, while recalling $R = 1/\sqrt{2}$ from the Hadamard gate, the inner products with $|00\rangle$ and $|11\rangle$ give $1/\sqrt{2}$. Hence the final state is $(1/\sqrt{2})(|00\rangle + |11\rangle)$. Execution on $\langle 01|$ gives:

$$(1)(1 - 1 - b_1 - b_2 + 2b_1b_2 + 2b_2 + 2b_1 - 4b_1b_2) = b_1 + b_2 - 2b_1b_2$$

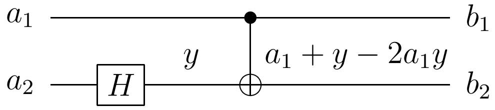
This is 0 when $b_1 = b_2$, so we obtain the similarly-entangled state $(1/\sqrt{2})(|01\rangle + |10\rangle)$. Finally, multiplying the matrices for $H \otimes I$ and CNOT gives:

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 \end{bmatrix}$$

$$P = -4a_1a_2b_1b_2 + 2a_1a_2b_1 + 2a_1b_1b_2 + 4a_2b_1b_2 - 2a_1b_1 - 2a_2b_1 - 2a_2b_2 - 2b_1b_2 + a_2 + b_1 + b_2$$

This equals the reduction of P_A above modulo the Boolean ideal. Thus this reduction will be part of any algebraic characterization of the polynomial equivalence relation.

Putting the Hadamard gate on line 2 instead gives:



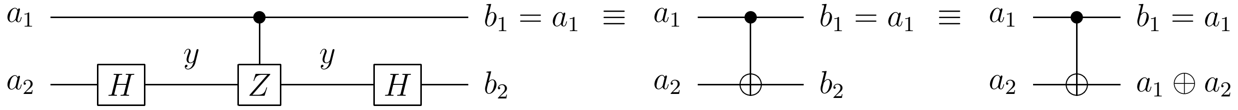
$$P = (1 - 2a_2y)P_I(a_1, b_1)P_I(a_1 + y - 2a_1y, b_2)$$

On input $\langle 00|$ this simplifies to

$$P_I(0, b_1)P_I(y, b_2) = (1 - b_1)(1 - y - b_2 + 2yb_2)$$

This forces $b_1 = 0$. When $b_2 = 0$ we have $1 - y$, which gives $N_B[1] = 1$, $N_B[-1] = 0$. When $b_2 = 1$ we have y, which also gives $N_B[1] = 1$, $N_B[-1] = 0$. Either way $N_B[1] - N_B[-1] = 1$, so remembering the $\sqrt{2}$ from the Hadamard gate, we get the non-entangled output state $(1/\sqrt{2})(|00\rangle + |01\rangle)$. Our ultimate purpose beginning with these examples is to ask, what can the classical polynomial algebra tell us about the capacity of the circuits to produce—or undo—entanglements?

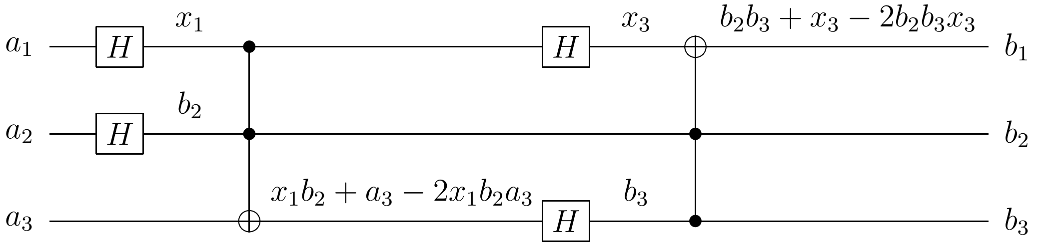
The simulation of CNOT by CZ and Hadamards shows another vicissitude of equivalence, this time with annotations ($a_1 \oplus a_2$ means $a_1 + a_2 - 2a_1a_2$) and different constants R.



$(1 - 2a_2y)(1 - 2a_1y)(1 - 2b_2y)$; $R \times 2$ vs. $1 - a_2 - b_2 + 2a_2b_2$ vs. 1

Even using the middle form with the CNOT polynomial that substitutes only on the first qubit line, the equivalence is not immediate to see, because the polynomial on the left is for a higher value of R. And with full substitution, when the circuit continues on the right, the CNOT annotation propagates $a_1 + a_2 - 2a_1a_2$ on the second qubit line, but the Hadamard plus CZ representation perforce propagates a single fresh variable.

Our last example is the circuit illustrated in [DHH$^+$04]. Our annotation is the same except for including the term with -2 to make the Toffoli substitution work over all rings, and renaming their "x_2" and "x_4" to "b_2" and "b_3." Also bear in mind the implicit I at the end of line 1. P'' and Q'' use $k = 2$.



$$\begin{array}{rcl} P' & = & (1-2a_1x_1)(1-2a_2b_2)(1-2x_1x_3)(1-2b_3(x_1b_2+a_3-2x_1b_2a_3)) \\ & & \cdot(1-b_1+(2b_1-1)(b_2b_3+x_3-2b_2b_3x_3)) \\ P'' & = & (1-a_1x_1(1+u^2))(1-a_2b_2(1+u^2))(1-x_1x_3(1+u^2))(1-b_3(x_1b_2+a_3)(1+u^2)) \\ & & \cdot(1-b_1-b_2b_3-x_3) \\ Q'' & = & a_1x_1+a_2b_2+x_1x_3+b_3x_1b_2+b_3a_3+w(b_1+b_2b_3+x_3) \end{array}$$

Note that Q'' is the most compact, but uses w. If to Q'' we adjoin the equation $b_1 = b_2b_3 + x_3$ (and restore x_2, x_4 with $b_2 = x_2$, $b_3 = x_4$) to avoid the w term, then we get the system in [DHH$^+$04]. Note that the w variables do not contribute to the denominator over which the numbers of solutions are placed. Here the denominator is 4 from the four Hadamard gates, and there are four variables x_1, x_2, x_3, x_4 Substituting $a_1 = a_2 = a_3 = 0$ to run on $\langle 000|$ gives

$$\begin{array}{rl} & b_2b_3x_1 + x_1x_3 + w(b_2b_3 + b_1 + x_3) \\ = & x_1(b_2b_3 + x_3) + w(b_1 + b_2b_3 + x_3). \end{array}$$

If $b_1 = 1$, then the values of $b_2b_3 + x_3$ that make the w-term vanish leave $x_1 \cdot 1$ in the other term. Since x_2 and x_4 are out of the picture, and since the arguments $x_1 = 0, 1$ give opposite parity, the algebra represents an interference effect that creates zero amplitude for $b_1 = 1$. With $b_1 = 0$, both terms vanish so the two assignments to x_1 create amplitude $2/4 = 1/2$ for each of the four basis combinations of b_2 and b_3. Thus the final state is a separable one, viz.

$$\frac{1}{2}(|000\rangle + |001\rangle + |010\rangle + |011\rangle = |0\rangle \otimes \frac{(|0\rangle + |1\rangle)^{\otimes 2}}{2}$$

Running on $a = |101\rangle$, however, gives

$$x_1(1 + b_2b_3 + x_3) + b_3 + w(b_1 + b_2b_3 + x_3).$$

This now interferes with $b_1 = 0$. When $b_1 = 1$, the only non-cancelling/non-cancelled term is b_3. This contributes to the sign of $\langle a|\ C\ |b\rangle$, but does not matter to the amplitude as there is no dependence on x_1 or x_3, so the final state is $|1\rangle \otimes$ the equal superposition of the other two qubits.

5 Applications

Two central theoretical problems are:

(1) Which subsets of quantum gates can be simulated efficiently with classical computation alone?

(2) What (classical) upper and lower bounds can be given for BQP?

An important subclass for (1) is the collection of *stabilizer circuits*, formed from the single-qubit Pauli, Hadamard, and S gates, and the CNOT and/or CZ gate. (With the latter, the Pauli gates can be removed to make a minimal set.) The original $O(s^3)$-time algorithm for simulating stabilizer circuits with s gates has since been improved to $O(s^2)$, $O(s \log s)$, and Jozsa [Joz08] sketched an $O(s)$-time algorithm. Can we give an $O(s)$-time algorithm without needing overhead for data structures such as the graph-state representation? As itemized above, the Q'' polynomials for these gates mod $k = 4$ have especially simple forms and are invariant under adding 2 mod 4 to any argument. It is enough to treat the Hadamard, S, and CNOT and/or CZ gates:

1. Hadamard: $2yz$, with no substitution; and
2. S: y^2, substituting $z := y$; and
3. CZ: $2y_1y_2$, substituting $z_1 := y_1$, $z_2 := y_2$; *or*
4. CNOT: 0, substituting $z_1 := y_1$, $z_2 := y_1 + y_2$, with the latter being sound in place of the proper $z_2 := y_1 + y_2 - 2y_1y_2$ owing to the invariance under adding 2.

Is there an easy inductive argument here? One can complete squares so that (with the CZ option), every term is y^2 or $(y + z)^2$. By invariance we can replace the former by y, but can we massage the latter further?

Generally, the task here is to identify subsets of polynomials for which the associated solution-counting problems are solvable in classical polynomial time. Reductions by polynomial equivalence may also contribute to the algorithms.

For (2), an immediate problem is whether BQP is contained in the polynomial hierarchy. The reason this does not follow immediately from the approximate counting results of Stockmeyer [Sto83] is that approximations to $f(x)/2^m$ and $g(x)/2^m$ may not help with $(f(x)-g(x))/2^h$ when h is about $m/2$, as needed when a circuit has m Hadamard gates. If the denominator were 2^m, e.g. if the expression for the probability $\Pr(0)$ of measuring 0 on the first qubit line did not need m extra variables, then simulating BQP would only require distinguishing the cases of $f(x)$ being near 2^m versus being near 2^{m-1}, which a Δ_2^p algorithm can do (deterministically) without needing the fill Δ_3^p power of Stockmeyer's approximate counting [Sto83]. No such simple demonstration appears to be forthcoming. However, we can still hope to seize on particular properties of some of the representations and refine Stockmeyer's proof techniques for them.

We may also explore the algebraic barriers to doing classical simulations of sets of gates known to be universal. Josza and Miyake [JM08, Joz08] have shown some fine distinctions involving Valiant's *matchgates*:

General Matchgate

$$M = R \begin{bmatrix} p & 0 & 0 & q \\ 0 & a & b & 0 \\ 0 & c & d & 0 \\ r & 0 & 0 & s \end{bmatrix}, \qquad \begin{bmatrix} p & q \\ r & s \end{bmatrix}, \begin{bmatrix} a & b \\ c & d \end{bmatrix} \text{ unitary, } ps - rq = ad - cb$$

There are some nice properties here. For one, the entries are nonzero only when z_2 has the same parity as $y_1+y_2+z_1$, so we can substitute $z_2 := y_1+y_2+z_1-2y_1y_2-2y_1z_1-2y_2z_1+4y_1y_2z_1$ for P'_M over any ring, which becomes the simple linear substitution $z_2 := y_1+y_2+z_1$ over $\mathbf{Z}_2[u]$. The conditions on p, q, r, s and a, b, c, d may yield something similar for z_1. However, Josza's paper includes a case where the *nearest-neighbor* restriction yields a classical simulation, but the *next-nearest* case is universal! It remains to see how the nearest-neighbor restriction will show up in the algebra.

6 Algebra, Entanglement, and Complexity

Above, we recognized the entangled Bell state arising from the polynomial $1-b_1-b_2+2b_1b_2$ under multiplicative representation. The question is, can we give a general algebraic characterization of entanglement—more properly, of a quantum circuit's capacity to produce entanglements—via the "classical" polynomials? It is famously difficult even to choose among candidate entanglement measures for n-qubit states (see [PV05]), much less for circuits that might produce them. More generally, we suspect that certain algebraic invariants of polynomials and associated polynomial ideals should have physical significance in quantum circuits.

[More To Come]

7 Conclusions

Acknowledgments Most of this work was done while the first author was a sabbatical visitor to the Universitié de Montreal, partly supported by the UdeM Département d'informatique et de recherche opérationnelle, and by the University at Buffalo Computer Science Department. We thank especially Professors Pierre McKenzie and Alain Tapp for helpful discussions. More recently we thank John Sidles and Michael Nielsen for quick reactions, and the readers of *Gödel's Lost Letter and P=NP* for…

References

[DHH$^+$04] C. Dawson, H. Haselgrove, A. Hines, D. Mortimer, M. Nielsen, and T. Osborne. Quantum computing and polynomial equations over the finite field Z_2. *Quantum Information and Computation*, 5:102–112, 2004.

[GS06] V. Gerdt and V. Severyanov. A software package to construct polynomial sets over Z_2 for determining the output of quantum computations. *Nuclear Instruments and Methods in Physics Research A*, 59:260–264, 2006.

[HH08] B. Hiesmayr and M. Huber. Multipartite entanglement measure for all discrete systems. *Phys. Rev. A*, 78:012342 1–7, 2008.

[JM08] R. Jozsa and A. Miyake. Matchgates and classical simulation of quantum circuits, April 2008.

[Joz08] R. Jozsa. Embedding classical into quantum computation, Dec. 2008.

[PV05] M. Plenio and S. Virmani. An introduction to entanglement measures, April 2005. rev. 6/10/06.

[Sto83] L. Stockmeyer. The complexity of approximate counting. In *Proc. 15th Annual ACM Symposium on the Theory of Computing*, pages 118–126, Baltimore, USA, April 1983. ACM Press.