Quantum Mass Production Theorems

William Kretschmer
University of Texas at Austin, TX, USA

Abstract

We prove that for any n-qubit unitary transformation U and for any $r = 2^{o(n/\log n)}$, there exists a quantum circuit to implement $U^{\otimes r}$ with at most $O(4^n)$ gates. This asymptotically equals the number of gates needed to implement just a *single* copy of a worst-case U. We also establish analogous results for quantum states and diagonal unitary transformations. Our techniques are based on the work of Uhlig [Math. Notes 1974], who proved a similar mass production theorem for Boolean functions.

2012 ACM Subject Classification Theory of computation → Quantum complexity theory; Theory of computation → Circuit complexity

Keywords and phrases mass production, quantum circuit synthesis, quantum circuit complexity

Digital Object Identifier 10.4230/LIPIcs.TQC.2023.10

Related Version *Previous Version*: https://arxiv.org/abs/2212.14399

Funding Supported by an NDSEG fellowship.

Acknowledgements Part of this work was done while the author attended the 2022 Extended Reunion for the Quantum Wave in Computing at the Simons Institute for the Theory of Computing. We thank Alex Meiburg for helpful discussions.

1 Introduction

If a computational task requires c resources, then common sense dictates that repeating the same task r times should require roughly $c \cdot r$ resources. In many settings, including query complexity [11] and communication complexity [12, 4], this intuition can be made rigorous: such results are known as *direct sum theorems*. Closely related are *direct product theorems*, which show that, with a fixed computational budget, the probability of successfully performing r independent tasks decays in r. We recommend [7, Chapter 1] for a good overview of the topic.

Nevertheless, direct sum and direct product theorems are not universal. Some computational settings exhibit a "mass production" phenomenon, in which the cost of performing the same task many times in parallel does *not* scale linearly with the number of repetitions. A well-known example [13, 7] is based on the circuit complexity of matrix-vector multiplication. For a matrix $M \in \{0,1\}^{n\times n}$, define $f_M : \{0,1\}^n \to \{0,1\}^n$ by $f_M(v) = Mv$, where addition and multiplication are taken mod 2. Then a simple counting argument implies that for most M, the complexity of implementing f_M via a Boolean circuit is at least $\Omega(n^2/\log n)$, as measured by the number of 2-bit AND, OR, and NOT gates. Yet, by observing that f_M^n (i.e. f_M repeated n times) is simply a matrix-matrix multiplication, we find that the cost of implementing f_M^n is only $O(n^\omega)$, where $\omega < 2.38$ is the exponent of matrix multiplication [3, 8] – substantially less than the naive bound of $O(n^3)$.

One might be left with the impression that such mass production phenomena can only occur for extremely special functions, like matrix multiplication, that have a particular algebraic or combinatorial structure. Remarkably, this intuition fails dramatically in the setting of Boolean circuit complexity. A theorem of Uhlig [17, 18, 19] shows that for *any*

Boolean function $f : \{0,1\}^n \to \{0,1\}$ and for any $r = 2^{o(n/\log n)}$, there exists a Boolean circuit implementing f^r with at most $O\left(\frac{2^n}{n}\right)$ gates. Asymptotically, this equals the number of gates needed to evaluate a worst-case f on a *single* input, by the well-known counting argument of Shannon [15]. In fact, Uhlig even showed that the leading constant in the big-O does not increase with r, and hence arbitrary Boolean functions can be mass produced with essentially no overhead.

1.1 This Work

In this work, we consider the natural question of whether a similar mass production phenomenon holds for quantum circuit complexity. Our question is well-motivated by recent works demonstrating that for certain learning tasks, algorithms with access to many copies of a quantum state on a quantum memory can be exponentially more powerful than algorithms that have access only to single copies of the state at a time [6, 10, 5]. Indeed, these results suggest that optimizing the complexity of mass producing quantum states and processes could have valuable applications. We also view our question as interesting from a purely theoretical perspective, especially considering that Uhlig's theorem for classical functions has recently found complexity-theoretic applications in characterizing the minimum circuit size problem [14, 9].

For simplicity, we consider quantum circuit complexity in the setting of qubit quantum circuits, using the universal gate set of arbitrary single-qubit gates plus CNOT gates with all-to-all connectivity. We also allow ancilla qubits initialized to $|0\rangle$, so long as they are reset to $|0\rangle$ at the end of the computation. We measure circuit complexity in terms of the CNOT count. This measure is justified by the fact that multiple-qubit gates are more error-prone and expensive to implement than single-qubit gates, and also by the observation that the number of single-qubit gates is related to the CNOT count by at most a factor of 4 in any irredundant circuit.

In analogy with Uhlig's theorem [17, 18, 19], our main result establishes mass production theorems for both quantum states and unitary transformations.

▶ **Theorem 1.** *Let $|\psi\rangle$ be an n-qubit quantum state, and let $r = 2^{o(n/\log n)}$. Then there exists a quantum circuit with at most $(1 + o(1))2^n$ CNOT gates to prepare $|\psi\rangle^{\otimes r}$.*

▶ **Theorem 2.** *Let U be an n-qubit unitary transformation, and let $r = 2^{o(n/\log n)}$. Then there exists a quantum circuit with at most $(5/2 + o(1))4^n$ CNOT gates to implement $U^{\otimes r}$.*

Note that the factor 2^n (respectively, 4^n), in Theorem 1 (respectively, Theorem 2) is optimal, because it asymptotically equals the number of CNOT gates needed to prepare a *single* copy of an arbitrary n-qubit state (respectively, to implement an arbitrary n-qubit unitary once), up to a small multiplicative constant [16]. Above, we made the leading constants explicit only to illustrate that they are not too large, and thus to demonstrate that these theorems have some hope of becoming practical. We leave a full optimization of these constants and the factors hidden in the $o(1)$ to future work.

1.2 Proof Overview

Our results build heavily on the simple proof of Uhlig's theorem given in [19], which we now briefly summarize. The proof proceeds by first showing that for an arbitrary $f : \{0,1\}^n \to \{0,1\}$, one can compute 2 copies of f using roughly $\frac{2^n}{n}$ gates – the same cost

as is needed to compute a single copy of a worst-case f. Then, Uhlig shows that we can generalize to a larger number of repetitions r by a straightforward recursive argument. So, we focus on the $r = 2$ case.

Fix a parameter k do be chosen later, and define for each $0 \leq i \leq 2^k - 1$ the function $f_i : \{0,1\}^{n-k} \to \{0,1\}$ to be the restriction of f obtained by fixing the first k bits to be the binary representation of i. So, for example,

$$f(\underbrace{0,0,\ldots,0}_{k \text{ times}}, x_{k+1}, \ldots, x_n) = f_0(x_{k+1}, \ldots, x_n).$$

Next, we define a set of functions $g_\ell : \{0,1\}^{n-k} \to \{0,1\}$ for each $0 \leq \ell \leq 2^k$ by:

- $g_0 = f_0$.
- $g_\ell = f_{\ell-1} \oplus f_\ell$ if $1 \leq \ell \leq 2^k - 1$.
- $g_{2^k} = f_{2^k-1}$.

Observe that

$$f_i = \bigoplus_{\ell=0}^{i} g_\ell = \bigoplus_{\ell=i+1}^{2^k} g_\ell. \tag{1}$$

Now, suppose that we have a pair of inputs $x, y \in \{0,1\}^n$ to f, and our goal is to evaluate $f(x)$ and $f(y)$ simultaneously. Let i and j denote the integers whose binary representations are the first k bits of x and y, respectively. Assume without loss of generality that $i \leq j$. Uhlig's idea is to evaluate $f(x)$ using the decomposition $f_i = \bigoplus_{\ell=0}^{i} g_\ell$ and $f(y)$ using $f_j = \bigoplus_{\ell=j+1}^{2^k} g_\ell$. The key observation is that in doing so, we only need to evaluate each g_ℓ at most once. The cost of computing $f(x)$ and $f(y)$ this way is dominated by computing the g_ℓs. So, the total size of the circuit is roughly

$$\left(2^k + 1\right)\left(\frac{2^{n-k}}{n-k}\right),$$

because there are $2^k + 1$ different g_ℓs, and each g_ℓ is a function on $n - k$ bits. For reasonable choices of k, this is asymptotically $(1 + o(1))\frac{2^n}{n}$, as desired.

Our main insight is that the same general approach generalizes straightforwardly from mass producing Boolean functions to mass producing diagonal unitary matrices, which we establish in Theorem 4. In one sense, the only conceptual change between our proof and Uhlig's is that we work with the group of complex units under multiplication, rather than the group $\{0,1\}$ under XOR. Nevertheless, our proof requires some care, as we do not deal with diagonal matrices directly. Rather, we mass produce the direct sum of a diagonal unitary with its inverse. In other words, for an n-qubit diagonal unitary U, we find it more convenient to work with the diagonal unitary on $n + 1$ qubits that applies U when the last qubit is $|0\rangle$, and $U^\dagger$ when the last qubit is $|1\rangle$. The intuitive reason why we require this change is that the XOR function is its own inverse, whereas multiplication by a complex unit is generally not.

Finally, once we have established Theorem 4 for diagonal unitary transformations, we obtain the mass production theorems for quantum states and general unitary transformations by using well-known decompositions of states and unitaries into diagonal gates [16].

2 Preliminaries

2.1 Basic Notation

We denote by $\mathbb{1}\{p\}$ the function that evaluates to 1 if proposition p is true, and 0 otherwise. If α is a complex number, we let α^* denote its complex conjugate. We denote by $\mathbb{T} := \{a + bi : |a|^2 + |b|^2 = 1\}$ the set of complex units. For a function $f : \{0,1\}^n \to \mathbb{T}$, denote by $\bar{f} : \{0,1\}^{n+1} \to \mathbb{T}$ the function defined by $\bar{f}(x,c) = f(x)^{1-2c}$, so that $\bar{f}$ evaluates to f when $c = 0$ and evaluates to f^* when $c = 1$. We freely identify a function $f : \{0,1\}^n \to \mathbb{T}$ with the corresponding diagonal unitary transformation U that acts as $U|x\rangle = f(x)|x\rangle$ on basis states $x \in \{0,1\}^n$.

We use standard notation for quantum circuits, including CNOT, Toffoli, and Fredkin gates. We also borrow a large amount of notation and terminology from [16], as we detail further below. We define the x-, y-, and z-axis rotations by:

$$R_x(\theta) = \begin{pmatrix} \cos(\theta/2) & i\sin(\theta/2) \\ i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix},$$
$$R_y(\theta) = \begin{pmatrix} \cos(\theta/2) & \sin(\theta/2) \\ -\sin(\theta/2) & \cos(\theta/2) \end{pmatrix},$$
$$R_z(\theta) = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix}.$$

2.2 Multiplexors

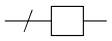
A *multiplexor* with s select qubits and d data qubits is a block-diagonal $(s+d)$-qubit unitary transformation that preserves every computational basis state $|x\rangle$ on the select qubits. For brevity, we call such a unitary an (s,d)-multiplexor. An $(s,1)$-multiplexor in which all of the diagonal blocks are R_z on the data qubit may alternatively be called a *multiplexed* R_z (analogously for R_x and R_y). Collectively, multiplexed R_x, R_y, and R_z are called *multiplexed rotations*. Observe that an $(s,1)$-multiplexed R_z is equivalent to a unitary implementing $\bar{f}$ for some $f : \{0,1\}^s \to \mathbb{T}$.

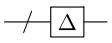
We require the following basic fact about implementing multiplexed rotations:

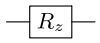
▶ **Proposition 3** ([16, Theorem 8]). *Let U be an $(n,1)$-multiplexed rotation. Then there exists a quantum circuit with at most 2^n CNOT gates to implement U.*

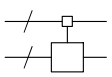
2.3 Generic Gates

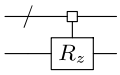
As in [16], we use circuit diagrams containing *generic gates*. An equivalence of two circuit diagrams containing generic gates means that for any assignment of parameters to the generic gates on one side, there exists an assignment of parameters to the gates on the other side that makes the two circuits compute the same operator. We use the following notation for generic gates:


A generic unitary gate.


A generic diagonal unitary gate.


An R_z gate for some unspecified θ. Conventions for R_x and R_y are analogous.


A generic multiplexor, with select qubits on the upper register and data qubits on the lower register


A multiplexed R_z. Conventions for R_x and R_y are analogous.

3 Diagonal Unitaries and Multiplexors

We begin by generalizing the proof of Uhlig's theorem [19] to diagonal unitary matrices (or, more precisely, multiplexed R_z gates).

▶ **Theorem 4.** *Let $f : \{0,1\}^n \to \mathbb{T}$ and let $r = 2^{o(n/\log n)}$. Then there exists a quantum circuit with at most $(1+o(1))2^n$ CNOT gates to implement $\bar{f}^{\otimes r}$.*

Proof. Without loss of generality, let $r = 2^t$ for some $t = o(n/\log n)$. Our proof proceeds by induction on t: for fixed k (chosen later) and for every $n > k \cdot t$, we construct for each $f : \{0,1\}^n \to \mathbb{T}$ a circuit $\mathcal{C}_{f,n,k,t}$ computing $\bar{f}^{\otimes 2^t}$. We proceed in order: first we construct $\mathcal{C}_{f,n,k,1}$ for every n and f, then $\mathcal{C}_{f,n,k,2}$ for every n and f, then $\mathcal{C}_{f,n,k,3}$ for every n and f, and so on. Ultimately, we show that there exists a universal constant d such that the number of CNOT gates in $\mathcal{C}_{f,n,k,t}$, denoted $s_{n,k,t}$, satisfies the bound:

$$s_{n,k,t} \leq \left(2^k + 1\right)^t \left(2^{n-tk} + 2^t dn\right). \tag{2}$$

We begin by describing the construction of $\mathcal{C}_{f,n,k,1}$. For each $0 \leq i \leq 2^k - 1$, let $f_i : \{0,1\}^{n-k} \to \mathbb{T}$ denote the restriction of f obtained by fixing the first k bits to the binary representation of i. For each $0 \leq i \leq 2^k$, define $g_i : \{0,1\}^{n-k} \to \mathbb{T}$ by:

- $g_0 = f_0$.
- $g_\ell = f^*_{\ell-1} f_\ell$ if $1 \leq \ell \leq 2^k - 1$.
- $g_{2^k} = f^*_{2^k-1}$.

Observe that

$$f_i = \prod_{\ell=0}^{i} g_\ell = \prod_{\ell=i+1}^{2^k} g^*_\ell. \tag{3}$$

The key idea in the remainder of the proof is to evaluate $\bar{f}$ on a pair of inputs (x, y) using the two decompositions in (3), one each for x and y. Indeed, the following algorithm accomplishes this.

Algorithm 1 Evaluate $\bar{f}^{\otimes 2}$.

Input: $x, y \in \{0,1\}^n$, $c_x, c_y \in \{0,1\}$
Output: $\bar{f}(x, c_x) \cdot \bar{f}(y, c_y)$

$\alpha \coloneqq 1$
if $x \leq y$ **then** /* viewing x, y as integers w/ highest order bits x_1, y_1 */
  $m \coloneqq x$; $c_m \coloneqq c_x$ /* set $m = \min\{x, y\}$, $M = \max\{x, y\}$ */
  $M \coloneqq y$; $c_M \coloneqq c_y$
else
  $m \coloneqq y$; $c_m \coloneqq c_y$
  $M \coloneqq x$; $c_M \coloneqq c_x$
for $0 \leq \ell \leq 2^k$ **do**
  if $\ell \leq m_{[1:k]}$ **then** /* $x_{[i:j]}$ denotes bits i through j of x */
    Multiply α by $\bar{g}_\ell(m_{[k+1:n]}, c_m)$
  else if $\ell > M_{[1:k]}$ **then**
    Multiply α by $\bar{g}_\ell(M_{[k+1:n]}, 1 - c_M)$ /* note negation on c_M */
  else
    Multiply α by 1
return α

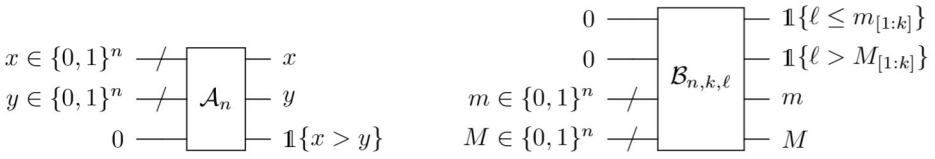
Here, the $\ell \leq m_{[1:k]}$ clause corresponds to the multiplication $\prod_{\ell=0}^{m_{[1:k]}} g_\ell$, while the $\ell > M_{[1:k]}$ clause corresponds to $\prod_{\ell=M_{[1:k]}}^{2^n} g_\ell^*$. An equivalent reformulation of Algorithm 1 is given below.

Algorithm 2 Evaluate $\bar{f}^{\otimes 2}$

Input: $x, y \in \{0,1\}^n$, $c_x, c_y \in \{0,1\}$
Output: $\bar{f}(x, c_x) \cdot \bar{f}(y, c_y)$

$\alpha \coloneqq 1$
if $x \leq y$ **then**
  $m \coloneqq x$; $c_m \coloneqq c_x$
  $M \coloneqq y$; $c_M \coloneqq c_y$
else
  $m \coloneqq y$; $c_m \coloneqq c_y$
  $M \coloneqq x$; $c_M \coloneqq c_x$
for $0 \leq \ell \leq 2^k$ **do**
  $a \coloneqq \mathbb{1}\{\ell \leq m_{[1:k]}\}$ /* at most one of a, b is nonzero */
  $b \coloneqq \mathbb{1}\{\ell > M_{[1:k]}\}$
  $z \coloneqq a \cdot m_{[k+1:n]} \oplus b \cdot M_{[k+1:n]}$
  $c \coloneqq a \cdot c_m \oplus b \cdot (1 - c_M)$
  Multiply α by $\bar{g}_\ell(z, c)$
  Multiply α by $g_\ell^*(0^{n-k})^{(1-a)\cdot(1-b)}$ /* undo added phase in case $a = b = 0$ */
return α

Algorithm 2 readily extends to a quantum circuit implementation. Define a pair of classical reversible circuits $\mathcal{A}_n$ and $\mathcal{B}_{n,k,\ell}$ whose input and output behavior are given in Figure 1. Using $\mathcal{A}_n$ and $\mathcal{B}_{n,k,\ell}$, via the same strategy as Algorithm 2, we obtain the quantum circuit $\mathcal{C}_{f,n,k,1}$ defined in Figure 2 that implements $\bar{f}^{\otimes 2}$.



(a) The circuit $\mathcal{A}_n$. **(b)** The circuit $\mathcal{B}_{n,k,\ell}$.

Figure 1 Inputs and outputs of reversible circuits $\mathcal{A}_n$ and $\mathcal{B}_{n,k,\ell}$.

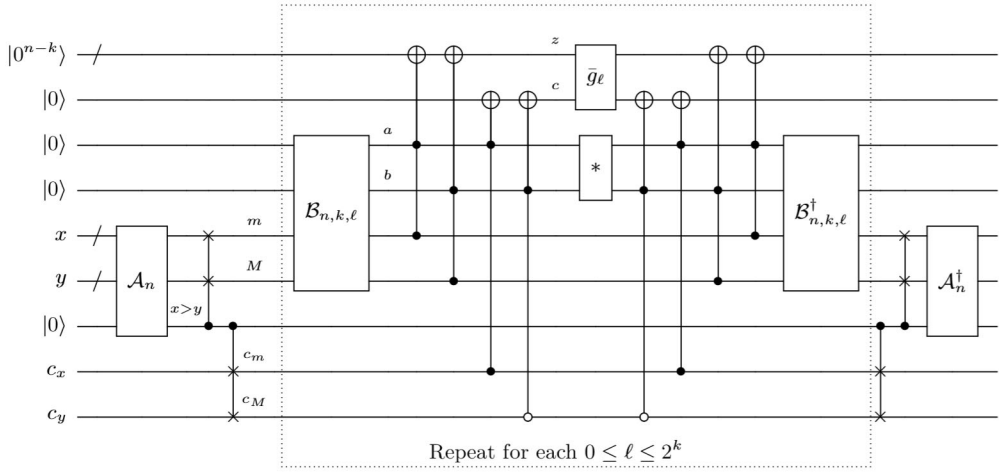


Figure 2 Circuit diagram of $\mathcal{C}_{f,n,k,1}$. The Toffoli gates with controls acting on the x and y registers are understood to be arrays of $n-k$ Toffoli gates between the corresponding qubits of the control and target registers. The gate marked $*$ adds a phase of $g^*_\ell(0^{n-k})$ if both qubits are $|0\rangle$ and otherwise does nothing. For convenience, several of the wires are labeled with the values they take on corresponding to variables in Algorithm 2.

By Proposition 3, for every ℓ, $\bar{g}_\ell$ can be implemented using at most 2^{n-k} CNOT gates, because $\bar{g}_\ell$ is equivalent to an $(n-k,1)$-multiplexed R_z. Moreover, it is easy to see that $\mathcal{A}_n$ and $\mathcal{B}_{n,k,\ell}$ can be implemented using at most $O(n)$ CNOT gates each, because comparison of two n-bit integers can be performed by a classical circuit of at most $O(n)$ gates. As a consequence, we conclude that there exists a constant d such that:

$$s_{n,k,1} \leq \left(2^k + 1\right)\left(2^{n-k} + dn\right). \tag{4}$$

This is certainly less than the bound in (2), so this establishes the base case of the induction proof.

Now we proceed to the induction step on t. Suppose that for every $n > k \cdot (t-1)$, we have a circuit $\mathcal{C}_{f,n,k,t-1}$ computing $\bar{f}^{\otimes 2^{t-1}}$ with CNOT count bounded by

$$s_{n,k,t-1} \leq \left(2^k + 1\right)^{t-1}\left(2^{n-(t-1)k} + 2^{t-1}dn\right). \tag{5}$$

To construct $\mathcal{C}_{f,n,k,t}$, we start by first taking 2^{t-1} copies of $\mathcal{C}_{f,n,k,1}$. Then, for each $0 \leq \ell \leq 2^k$, we replace each of the 2^{t-1} sub-circuits that compute $\bar{g}_\ell$ with $\mathcal{C}_{g_\ell,n-k,k,t-1}$. Then, the number

of gates in $\mathcal{C}_{f,n,k,t}$ is bounded by:

$$\begin{aligned}
s_{n,k,t} &\leq \left(2^k+1\right)\left(s_{n-k,k,t-1}+2^{t-1}dn\right)\\
&\leq \left(2^k+1\right)\left(\left(2^k+1\right)^{t-1}\left(2^{n-k-(t-1)k}+2^{t-1}d(n-k)\right)+2^{t-1}dn\right)\\
&\leq \left(2^k+1\right)^t\left(2^{n-tk}+2^{t-1}dn\right)+\left(2^k+1\right)2^{t-1}dn\\
&\leq \left(2^k+1\right)^t\left(2^{n-tk}+2^t dn\right),
\end{aligned}$$

where the first line substitutes (5) for the cost of the $\bar{g}_\ell$'s and otherwise uses the same bound as (4) for the non-$\bar{g}_\ell$ gates, and the second line applies the induction hypothesis (5). This establishes the induction step, and thus (2) holds for every $n > k \cdot t$.

Choose $k = \lceil \log n \rceil$. Then:

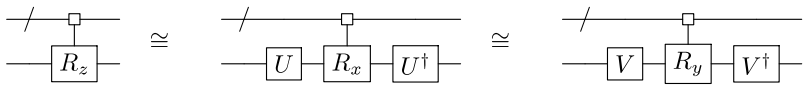
$$\begin{aligned}
s_{n,k,t} &\leq \left(2^k+1\right)^t\left(2^{n-tk}+2^t dn\right)\\
&= 2^{kt}\left(1+\frac{1}{2^k}\right)^t\left(2^{n-tk}+2^t dn\right)\\
&\leq 2^{kt}e^{t/2^k}\left(2^{n-tk}+2^t dn\right)\\
&\leq 2^{kt}(1+o(1))\left(2^{n-tk}+2^t dn\right)\\
&\leq 2^{kt}(1+o(1))\left(2^{n-tk}+o\left(2^{n-tk}\right)\right)\\
&\leq (1+o(1))2^n,
\end{aligned}$$

where we applied the exponential inequality in the third line, and used the assumption $t \leq o(n/\log n)$ in the fourth and fifth lines. This proves the theorem. ◀

Theorem 4 straightforwardly generalizes to arbitrary multiplexed rotations and multiplexors with a single data qubit, as below.

▶ **Corollary 5.** *Let U be an $(n,1)$-multiplexed rotation, and let $r = 2^{o(n/\log n)}$. Then there exists a quantum circuit with at most $(1+o(1))2^n$ CNOT gates to implement $U^{\otimes r}$.*

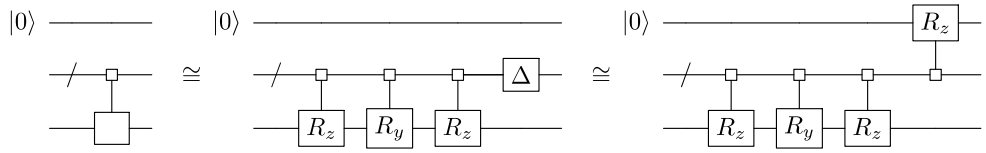
Proof. The R_z case follows by observing that $\bar{f}$ is exactly an $(n,1)$-multiplexed R_z in Theorem 4. This also extends to multiplexed R_x and R_y, because multiplexed R_x, R_y, and R_z are equivalent up to conjugation by single-qubit unitaries on the data qubit. That is, there exist single-qubit unitaries U and V such that:



Hence, the CNOT count is identical for multiplexed R_x and R_y as well. ◀

▶ **Corollary 6.** *Let U be an $(n,1)$-multiplexor, and let $r = 2^{o(n/\log n)}$. Then there exists a quantum circuit with at most $(4+o(1))2^n$ CNOT gates to implement $U^{\otimes r}$.*

Proof. By [16, Theorem 6], an arbitrary $(n,1)$-multiplexor may be implemented via a product of 4 $(n,1)$-multiplexed rotations, as below.



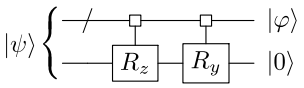
Applying Corollary 5 to each of the multiplexed rotations on the right side above completes the proof. ◀

4 States and General Unitaries

We now prove the main results of this work that generalize the mass production theorems above to state preparation and unitary compilation. The proofs proceed via the techniques of [16], by decomposing operators into multiplexors.

▶ **Theorem 1.** *Let $|\psi\rangle$ be an n-qubit quantum state, and let $r = 2^{o(n/\log n)}$. Then there exists a quantum circuit with at most $(1+o(1))2^n$ CNOT gates to prepare $|\psi\rangle^{\otimes r}$.*

Proof. By [16, Theorem 9], for any n-qubit quantum state $|\psi\rangle$, there exists an $(n-1)$-qubit state $|\varphi\rangle$ such that $|\psi\rangle$ has the following decomposition.



Applying this decomposition recursively, we conclude that $|\psi\rangle$ can be prepared by a circuit consisting of a pair of $(\ell,1)$-multiplexed rotations for each $1 \leq \ell \leq n-1$, and a pair of single-qubit gates.

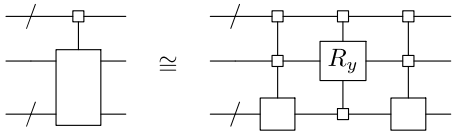
Apply Corollary 5 to the $(\ell,1)$-multiplexed rotations for each $\lceil n/2 \rceil \leq \ell \leq n-1$, and otherwise apply Proposition 3 r times for each $1 \leq \ell \leq \lceil n/2 \rceil - 1$. Then the total number of CNOT gates to prepare $|\psi\rangle^{\otimes r}$ is upper bounded by

$$\begin{aligned} r \cdot \sum_{\ell=1}^{\lceil n/2 \rceil - 1} 2^\ell + \sum_{\ell=\lceil n/2 \rceil}^{n-1} (1+o(1))2^\ell &\leq r2^{\lceil n/2 \rceil} + (1+o(1))2^n \\ &\leq 2^{\lceil n/2 \rceil + o(n/\log n)} + (1+o(1))2^n \\ &\leq (1+o(1))2^n \end{aligned}$$

◀

▶ **Theorem 2.** *Let U be an n-qubit unitary transformation, and let $r = 2^{o(n/\log n)}$. Then there exists a quantum circuit with at most $(5/2+o(1))4^n$ CNOT gates to implement $U^{\otimes r}$.*

Proof. By [16, Theorem 11], an arbitrary multiplexor can be expressed as below.



This decomposition is also valid when the multiplexor on the left side of the equivalence has 0 select bits. A recursive application of this decomposition implies that an arbitrary n-qubit unitary may be expressed as a product of $2^n - 1$ different $(n-1,1)$-multiplexors, of which $2^{n-1}-1$ are multiplexed R_y gates, and the remaining 2^{n-1} are arbitrary multiplexors. Applying Corollary 5 and Corollary 6 to these multiplexors gives the desired bound. ◀

5 Conclusion and Outlook

We have demonstrated that mass production phenomena are not unique to classical computation, and that they extend to quantum circuit complexity as well. As the message of this work is primarily conceptual in nature, we have not attempted to optimize every aspect of our results. Indeed, our mass production theorems could be extended further in a variety of ways; we outline a few such possibilities below.

If our results have any hope of being used in practice, then still more work needs to be done to optimize various constants. We suspect that the leading constant in Theorem 2 could be brought down from 5/2 to 1 with a more clever decomposition into multiplexors. The factors hidden in the $o(1)$ could probably be optimized further as well, especially those related to the constant factor d that appears in Theorem 4. Indeed, we believe that much of the redundancy in computing and uncomputing $\mathcal{B}_{n,k,\ell}$ for each $0 \leq \ell \leq 2^k$ could be reduced by more careful accounting.

It is also worth attempting to optimize other parameters of practical relevance, such as constraints on the gate set, locality, depth, and ancilla qubit count. In principle, our proof should allow for some tradeoff between depth and ancilla count, because the $\bar{g}_\ell$s in Figure 2 can either be evaluated sequentially or in parallel. Another particularly interesting question is whether ancilla qubits are necessary at all to achieve quantum mass production.

We leave open the circuit complexity of quantum mass production in other parameter regimes. As Theorem 1 and Theorem 2 only apply when $r = 2^{o(n/\log n)}$, it is natural to ask what happens when r is much larger. For Boolean functions, it is known that for any n-bit f, the "asymptotic complexity" of mass production $\lim_{r\to\infty} \frac{C(f^r)}{r}$ is bounded by $\mathrm{poly}(n)$ [13, 2], where $C(f^r)$ denotes the Boolean circuit complexity of implementing r copies of f. However, it is unclear whether the same approach would generalize to quantum circuits.

Lastly, we ask: are there any restricted examples of quantum circuits that exhibit a mass production phenomenon? What about Clifford circuits? We observe if one allows implementation by non-Clifford gates, then n copies of an arbitrary Clifford operation can be implemented by a circuit with at most $O(n^\omega)$ gates, where ω is the exponent of matrix multiplication. By the "canonical form theorem" of Aaronson and Gottesman [1], every Clifford circuit can be expressed in the form H-C-P-C-P-C-H-P-C-P-C, where each letter corresponds to a layer of Hadamard, CNOT, or phase gates. The Hadamard and phase layers contain at most $O(n)$ gates total, so it suffices to show how to implement n copies of a CNOT circuit using $O(n^\omega)$ gates. For any $M \in \mathbb{F}_2^{n\times n}$, define U_M as the unitary transformation that acts as $U_M |x\rangle |y\rangle = |x\rangle |y \oplus Mx\rangle$ on computational basis states. As every CNOT circuit implements an invertible linear transformation $|x\rangle \to |Mx\rangle$ for some $M \in \mathbb{F}_2^{n\times n}$, a CNOT circuit can be implemented using U_M and $U_{M^{-1}}$ and $O(n)$ additional gates via:

$$|x\rangle |0^n\rangle \xrightarrow{U_M} |x\rangle |Mx\rangle \xrightarrow{U_{M^{-1}}} |0^n\rangle |Mx\rangle \xrightarrow{\text{SWAP}} |Mx\rangle |0^n\rangle .$$

Then, as in Section 1, we can mass produce U_M and $U_{M^{-1}}$ using fast matrix multiplication.

References

1 Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70:052328, November 2004. doi:10.1103/PhysRevA.70.052328.

2 Andreas Albrecht. On simultaneous realizations of Boolean functions, with applications. In Gottfried Wolf, Tamáas Legendi, and Udo Schendel, editors, *Parcella '88*, pages 51–56, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg. doi:10.1007/3-540-50647-0_102.

3 Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539, 2021. doi:10.1137/1.9781611976465.32.

4 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, pages 67–76, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1806689.1806701.

5 Matthias C. Caro. Learning quantum processes and Hamiltonians via the Pauli transfer matrix, 2022. arXiv:2212.04471.

6 Sitan Chen, Jordan Cotler, Hsin-Yuan Huang, and Jerry Li. Exponential separations between learning with and without quantum memory. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 574–585, 2022. doi:10.1109/FOCS52979.2021.00063.

7 Andrew Donald Drucker. *The complexity of joint computation*. PhD thesis, Massachusetts Institute of Technology, 2012. URL: http://dspace.mit.edu/handle/1721.1/7582.

8 Ran Duan, Hongxun Wu, and Renfei Zhou. Faster matrix multiplication via asymmetric hashing, 2022. arXiv:2210.10173.

9 Shuichi Hirahara. NP-hardness of learning programs and partial MCSP. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 968–979. IEEE, 2022. doi:10.1109/FOCS54457.2022.00095.

10 Hsin-Yuan Huang, Michael Broughton, Jordan Cotler, Sitan Chen, Jerry Li, Masoud Mohseni, Hartmut Neven, Ryan Babbush, Richard Kueng, John Preskill, and Jarrod R. McClean. Quantum advantage in learning from experiments. *Science*, 376(6598):1182–1186, 2022. doi:10.1126/science.abn7293.

11 Rahul Jain, Hartmut Klauck, and Miklos Santha. Optimal direct sum results for deterministic and randomized decision tree complexity. *Information Processing Letters*, 110(20):893–897, 2010. doi:10.1016/j.ipl.2010.07.020.

12 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A direct sum theorem in communication complexity via message compression. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *Automata, Languages and Programming*, pages 300–315, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. doi:10.1007/3-540-45061-0_26.

13 Wolfgang J. Paul. Realizing Boolean functions on disjoint sets of variables. *Theoretical Computer Science*, 2(3):383–396, 1976. doi:10.1016/0304-3975(76)90089-X.

14 Hanlin Ren and Rahul Santhanam. Hardness of KT Characterizes Parallel Cryptography. In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:58, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2021.35.

15 Claude. E. Shannon. The synthesis of two-terminal switching circuits. *The Bell System Technical Journal*, 28(1):59–98, 1949. doi:10.1002/j.1538-7305.1949.tb03624.x.

16 Vivek V. Shende, Stephen S. Bullock, and Igor L. Markov. Synthesis of quantum-logic circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(6):1000–1010, 2006. doi:10.1109/TCAD.2005.855930.

17 Dietmar Uhlig. On the synthesis of self-correcting schemes from functional elements with a small number of reliable elements. *Matematicheskie Zametki*, 15(6):937–944, 1974. In Russian. URL: http://mi.mathnet.ru/mz7425.

18 Dietmar Uhlig. On the synthesis of self-correcting schemes from functional elements with a small number of reliable elements. *Mathematical notes of the Academy of Sciences of the USSR*, 15(6):558–562, 1974. Translated from Russian. doi:10.1007/BF01152835.

19 Dietmar Uhlig. *Networks Computing Boolean Functions for Multiple Input Values*, pages 165–173. London Mathematical Society Lecture Note Series. Cambridge University Press, 1992. doi:10.1017/CBO9780511526633.013.