Quantum advantage and noise reduction in distributed quantum computing

J. Avron ,* Ofer Casper ,† and Ilan Rozen
Department of Physics, Technion, 320000 Haifa, Israel

(Received 24 April 2021; revised 17 August 2021; accepted 8 October 2021; published 5 November 2021)

Distributed quantum computing can provide substantial noise reduction due to shallower circuits. An experiment illustrates the advantages in the case of a Grover search. This motivates study of the quantum advantage of the distributed version of the Simon and Deutsch-Jozsa algorithms. We show that the distributed Simon algorithm retains the exponential advantage, but the complexity deteriorates from $O(n)$ to $O(n^2)$, where $n = \log_2(N)$. The distributed Deutsch-Jozsa algorithm deteriorates to being probabilistic but retains a quantum advantage over classical random sampling.

DOI: 10.1103/PhysRevA.104.052404

I. INTRODUCTION

The IBM [1] quantum experience and Amazon Braket [2] offer the opportunity to implement quantum algorithms on many small and noisy quantum computers. More than 20 quantum computers, with at most 65 qubits, have been deployed by IBM. None can communicate quantumly. The question then arises what advantages and disadvantages distributed quantum computing with classical communication offers.

Replacing quantum with classical resources usually leads to a large overhead. For example, simulating n qubits requires $O(N = 2^n)$ classical bits. More generally, simulating a quantum circuit with $n + k$ qubits with a quantum circuit with n qubits requires[1] $O(2^{ck})$ uses of the quantum circuits [3].

How much quantum advantage survives in distributed computing depends on the algorithms. Cirac *et al.* [4,5] showed that the distributed 3SAT retains a quantum advantage. Bravyi *et al.* [3] estimated the overhead in classical computation for sparse quantum circuits, and Peng *et al.* [6] derived related results for tensor networks with limited connections between clusters.

Distributed quantum computing can offer, besides the obvious advantage of additional "virtual qubits," the advantage of significant noise reduction. This comes about because splitting an algorithm can result in a significant reduction in depth. Since the noise in the output scales exponentially with the depth of the circuit this can be a significant advantage. For example, if the depth of a circuit is large enough, the output of the quantum computer may be overwhelmed by noise, but a shallower distributed computation may give significant results. As far as we can tell, this simple, one may say trivial, point has not been studied before.

We describe an experiment involving a Grover search [7] that illustrates the advantage of distributed quantum computing over the undistributed computation. In addition, we study the quantum advantage of distributed algorithms for two basic textbook examples: Simon's [8,9] and Deutsch and Jozsa's [10]. Since the Grover and Deutsch-Jozsa algorithms involve the use of an Oracle we also consider the task of distributing an Oracle.

As we shall see:

i. The distributed Simon's algorithm retains the exponential speedup, albeit with a higher complexity (see Sec. V A).

ii. The quantum advantage of the distributed Deutsch-Jozsa algorithm deteriorates dramatically (see Sec. VI).

II. DISTRIBUTED COMPUTATIONS OF BOOLEAN FUNCTIONS

We restrict ourselves to a particular scheme of distributed computing which suffices to cover the problems we consider.

A. Distributed classical computation

Consider a (classical) circuit with n bits that computes the function

$$f : \{0, 1\}^n \mapsto \{0, 1\}^m, \quad m \leqslant n - 1. \tag{2.1}$$

The function can be split into its even and odd parts: $f_{\text{even/odd}}:\{0, 1\}^{n-1} \mapsto \{0, 1\}^m$,

$$\begin{aligned} f_{\text{even}}(y_1, \dots, y_m) &= f(y_1, \dots, y_m, 0), \\ f_{\text{odd}}(y_1, \dots, y_m) &= f(y_1, \dots, y_m, 1). \end{aligned} \tag{2.2}$$

More generally,

$$\begin{aligned} f_{\text{even}}(y_1, \dots, y_m) &= f(y_1, \dots, y_{j-1}, 0, y_{j+1}, \dots, y_{m+1}), \\ f_{\text{odd}}(y_1, \dots, y_m) &= f(y_1, \dots, y_{j-1}, 1, y_{j+1}, \dots, y_{m+1}). \end{aligned} \tag{2.3}$$

We assume that the even and odd parts can be computed by a circuit with $n - 1$ bits. We can then distribute computing f to two $n - 1$ bit processors.

B. Distributed quantum computation

Suppose Alice has a processor with n connected qubits and Bob has two devices with $n - 1$ connected qubits each. The

*avronj@technion.ac.il
†ofercasper@gmail.com

[1] c is a function of n.

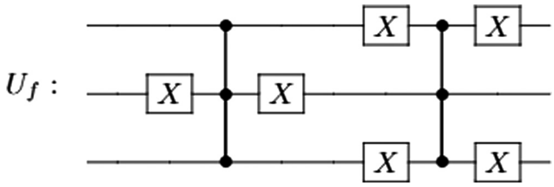


FIG. 1. The circuit corresponding to the DNF in Eq. (2.6). The columns of control gates show $C^{\otimes n}Z$ gates, in notation that manifests the symmetry of the gates.

general state of Alice's processor,

$$|\psi\rangle = \sum_{j=0}^{N-1} \psi_j |j\rangle, \quad N = 2^n, \tag{2.4}$$

is described by N amplitudes. The state of Bob's two processors is

$$|\phi_1\rangle \otimes |\phi_2\rangle, \quad |\phi_j\rangle = \sum_{k=0}^{N/2-1} \phi_{jk} |k\rangle, \tag{2.5}$$

and it, too, is described by $2N/2 = N$ amplitudes.

Alice's quantum advantage comes from her ability to entangle her n qubits. Bob cannot entangle the two processors and can only entangle $n-1$ qubits on each processor. Bob has more qubits and shallower circuits. This is an advantage in noisy quantum computers. Bob also has the advantage that his measurements give $2(n-1)$ bits of information, while Alice's measurement gives her only n bits of information. We do not, however, make use of this advantage here.

C. Quantum circuits and the disjunctive normal form (DNF)

The Grover and Deutsch-Jozsa algorithms involve the use of Oracles. We therefore face the problem of distributing an Oracle without introducing bias. To do so we assume that the Oracle is an algorithm with a standard form[2] that allows us to split it into the even and odd arguments. The reader who is willing to accept this on good faith may want to skip to the next section.

The Oracle, by definition, computes a Boolean function. Any Boolean function can be represented by a DNF. (For an elementary introduction see Appendix A). For example, the DNF of the function that assigns 1 to 101 and 010 and 0 otherwise is

$$f(x_0, x_1, x_2) = x_0 \cdot \bar{x}_1 \cdot x_2 + \bar{x}_0 \cdot x_1 \cdot \bar{x}_2, \tag{2.6}$$

where $x_j \in \{0, 1\}$ are logical variables (equivalently, binaries) and $\bar{x}_j$ is the (logical) NOT, (equivalently, for binary $\bar{x}_j = x_j \oplus 1$). The $+$ is the (logical) OR, normally denoted $\vee$.

In the general case with n logical variables x_j, $j \in 1, \ldots, n$, the DNF can be a sum of a large number of terms; each term is a product over all logical variables, and each variable appears once either as x_j or as $\bar{x}_j$.

The quantum circuit that computes

$$U_f |x\rangle = (-1)^{f(x)} |x\rangle \tag{2.7}$$

can be read out from the DNF. For the DNF in Eq. (2.6) the circuit is shown in Fig. 1. In the general case, a pair of X gates decorates all the $\bar{x}$'s and the n-fold product is represented by $C^n Z$.

Remark 2.1. The DNF is, in general, not the most compact representation of the function f, and similarly, the corresponding quantum circuit need not be the optimal circuit. Optimization of quantum circuits is considered in [11].

The even and odd parts of f are easily constructed from the DNF. If we use x_0 as the bit that determines even or odd, then $f_{e/o}$ and f_o for the example in Eq. (2.6) are

$$f_e = x_1 \cdot \bar{x}_2 \quad \text{and} \quad f_o = \bar{x}_1 \cdot x_2. \tag{2.8}$$

In the general case, all the $\bar{x}_0$ terms make the even part (with $\bar{x}_0$ deleted) and all the x_0 terms make the odd part (with x_0 deleted). The DNF is then used to construct the $n-1$ qubit circuits for the even and odd parts.

In conclusion, given Alice's n-qubit circuit corresponding to the DNF, there is a simple procedure that constructs Bob's $n-1$ qubit circuits for the even and odd parts of the function.

Algorithm 2.1 is for generating even and odd circuits. The algorithm returns the requisite $U^o_{f_B}$ and $U^e_{f_B}$:

$$\begin{aligned} U_{f_A} |1\rangle \otimes |\varphi\rangle &\equiv |1\rangle \otimes U^o_{f_B} |\varphi\rangle, \\ U_{f_A} |0\rangle \otimes |\varphi\rangle &\equiv |0\rangle \otimes U^e_{f_B} |\varphi\rangle. \end{aligned} \tag{2.9}$$

In Appendix B we describe the converse procedure whereby, starting from Bob's distributed quantum circuits, Alice can generate her single quantum circuit.

III. DEPTH OF DISTRIBUTED COMPUTATION

Distributed algorithms have the advantage of a shallower depth. This is evident for circuits given by the DNF: The depth of the n-qubit circuit is distributed between the two $n-1$ circuits. [See Eqs. (2.6) and (2.8).]

In the experiment (Sec. IV) the depth of the distributed circuits was about a factor of 4 smaller than that of the undistributed circuit. This is a consequence of the fact that the depth of a $C^n Z$ circuit is considerably larger than the depth of a $C^{n-1}Z$. We illustrate this with an example.

The gain in depth depends on the choice of gates. We use the following rules [12]:

i. Any single-qubit gate is allowed.

ii. The only two-qubit gate allowed is CNOT.

iii. Gates that can be executed in parallel are grouped into columns.

iv Consecutive single-qubit gates are counted as a single-qubit gate.

Consider the circuit in Fig. 1 and its distributed cousins in Fig. 2. To compute depth(U_{f_A}) and depth(U_{f_B}) we first observe that

$$\text{depth}(C^2 Z) = 11. \tag{3.1}$$

This is shown in Fig. 3.

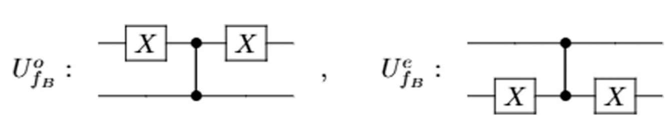


FIG. 2. The circuits corresponding to the DNF of Eq. (2.8).

[2]In computer science one normally does not worry about how the Oracle does what it does. But for the case at hand, we need to.

Algorithm 2.1: Splitting a circuit.

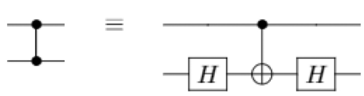
```
Result: two (n-1)-qubit circuits U^e_{f_B}, U^o_{f_B}
Divide (U_{f_A})
    parity ← 0;
    U^o_{f_B} ← 1;
    U^e_{f_B} ← 1;
    if U_{f_A} is not of DNF then
        abort ;
    foreach gate G in U_{f_A} in the order they are executed from top to bottom do
        if G is an X gate acting on the parity qubit ^a then
            parity ← NOT(parity) ;
        else if G is a C^{⊗(n−1)}Z then
            if parity == 1 then
                append C^{⊗(n−2)}Z to U^e_{f_B};
            else
                append C^{⊗(n−2)}Z to U^o_{f_B};
        else if G is a single qubit gate acting not on the parity qubit then
            append G to its respective qubit both in U^o_{f_B} and U^e_{f_B} ;
        else
            abort;
    return U^o_{f_B}, U^e_{f_B} ;
```

[a]The parity qubit is defined as the qubit that distinguishes between the even and odd subspaces.

It follows that

$$\text{depth}(U_{f_A}) = 3 + 2 \times \text{depth}(C^2Z) = 25. \quad (3.2)$$

To compute the depth of the corresponding distributed circuits in Fig. 2 we first recall that



(3.3)

For the even circuit one combines the X gates with the Hadamard gates. Hence both circuits have

$$\text{depth}\big(U_{f_B}^{e/o}\big) = 3.$$

The example shows that distributed circuits can lead to a substantial reduction in the depth of quantum circuits.

IV. DISTRIBUTED GROVER SEARCH: AN EXPERIMENT

In this section we describe an experiment carried out on the IBMQ5 (ibmq_santiago). The data for the machine (at the time of the experiments) are listed in Table I.

In the experiment a Grover search for a single target among 16 items ($N = 16$ and $M = 1$) was conducted on an undistributed circuit with $n = 4$ qubits and then on distributed circuits with $n = 3$ qubits each. The target state in both cases has been $|1111\rangle$.

We made use of the open-source Qiskit [13] library to generate the Grover search circuits, which were then run on real (i.e., not simulated) machines. The circuit data are reported in Table II.

Figure 5 shows the results for the undistributed Grover search, and Fig. 7 those for the distributed Grover search. Clearly, the undistributed search failed, while the distributed search qubits succeeded in finding $|1111\rangle$.

The results are in agreement with what one should expect for noisy machines that can handle a limited depth. For more details about the expected and observed noise in the circuits, see Appendix B.

A. Undistributed Grover search

The optimal number of Grover iterations with $n = 4$ (Fig. 4) is $r = (\pi/4)\sqrt{16} = \pi \approx 3$ and the (theoretical)

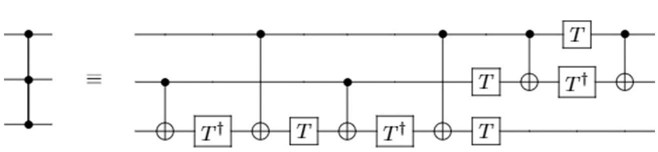


FIG. 3. C^2Z.

TABLE I. Machine data for IBM Santiago.

Single-qubit error	$O(2.2 \times 10^{-4})$
Two-qubit error	$\epsilon \approx 6.2 \times 10^{-3}$
Coherence	$T_c \approx 133\ \mu s$
Gate rate	$R \approx 408$ ns

TABLE II. Circuit data for the undistributed computation with $n = 4$ qubits and the two distributed circuits with $n = 3$ qubits. The values are for the transpiled circuits.

Qubits	Gates	CNOT	Time steps	Iterations	Repetitions	Success probability
4	519	291	396	3	8096	0.063
3 (odd)	140	55	93	2	8096	0.4
3 (even)	59	24	39	2	8096	NA

success probability is[3] 0.96. The circuit produced the histogram in Fig. 5 and manifestly failed to identify the target $|1111\rangle$.

It is useful to extract from the histogram the frequencies of bit-flip error in the target $|1111\rangle$. This is reported in Table III. The data agree with a binomial distribution up to $\pm\sigma$. The large error probability for bit-flip, $p_0 \approx 0.50$, is the reason for the failure of the search. p_0 is in reasonable agreement with the estimates based on the machine data in Table I (see Appendix B).

B. Distributed Grover search

The distributed Grover search runs with two Oracles, one for the 'odd' subspace and the other for the 'even' subspace, both with $n = 3$ qubits. The target $|1111\rangle$ is encoded in the odd Oracle and the even Oracle is "empty." The optimal number of iterations in the odd circuit is $r = (\pi/4)\sqrt{8} \approx 2$, with success probability $p \approx 0.95$. The circuits are shown in Fig. 6.

The left histogram in Fig. 7 clearly identifies the correct answer $|111\rangle$ (corresponding to $|1111\rangle$). The even circuit has no marked element[4] and this is borne out by the histogram on the right.

It is instructive to look at the probability of bit-flip error. The histogram in Fig. 7 leads to Table IV. The agreement with a binomial distribution is only qualitative. The small bit-flip error, $p_0 \approx 0.30$, is why the search succeeded. p_0 can be estimated from the machine data in Table I (see Appendix B).

V. DISTRIBUTED SIMON'S ALGORITHM

Period finding and the $\mathbb{Z}_2^n$ Fourier transform

Consider $f : \{0, 1\}^n \mapsto \{0, 1\}$, a Boolean function periodic with period $s \neq 0$ under bitwise addition:

$$f(x \oplus s) = f(x) \quad \forall x \in \{0, 1\}^n. \tag{5.1}$$

[3] $p = \sin^2((2r + 1)\theta) \approx 0.96$, where $p = \sin\theta = 1/\sqrt{N}$.
[4] Although the formal optimum for $M = 0$ is $r = \infty$, the optimal number of Grover iterations is actually $r = 0$.

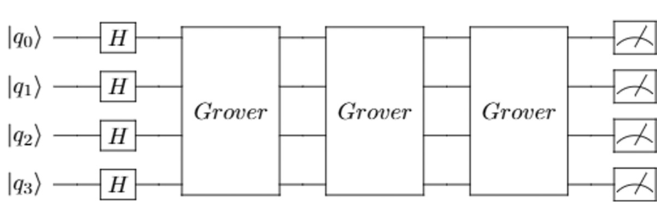


FIG. 4. The undistributed Grover search circuit with $n = 4$ qubits.

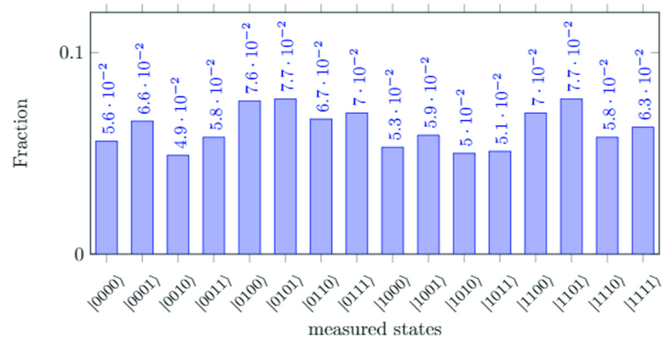


FIG. 5. Histogram for the undistributed Grover algorithm with $n = 4$ qubits searching for the target state $|1111\rangle$. The sampling error is ± 0.011. The search failed.

Task: Find s. This is a special case of Simon's problem [9].

Period finding is the business of Fourier transforms. For the case at hand, this is the $\mathbb{Z}_2^n$ Fourier transform. The circuit in Fig. 8 reduces period finding to solving a set of linear equations. Chasing the state $|0\rangle^{\otimes n}$ through the circuit in Fig. 8, using the identity

$$H^{\otimes n} |x\rangle = \frac{1}{\sqrt{N}} \sum_y (-1)^{x\cdot y} |y\rangle, \tag{5.2}$$

one finds

$$\begin{aligned} H^{\otimes n} |0\rangle &= \frac{1}{\sqrt{N}} \sum_x |x\rangle \\ &\xrightarrow{(-)^f} \frac{1}{\sqrt{N}} \sum_x (-1)^{f(x)} |x\rangle \\ &= \frac{1}{2\sqrt{N}} \sum_x (-1)^{f(x)} (|x\rangle + |x \oplus s\rangle) \\ &\xrightarrow{H^{\otimes n}} \frac{1}{2N} \sum_y \underbrace{\left(\sum_x (-1)^{f(x)+x\cdot y} \right)}_{=g(y)} (1 + (-1)^{s\cdot y}) |y\rangle. \end{aligned} \tag{5.3}$$

Here $g(y)$ is the $\mathbb{Z}_2^n$ Fourier transform of $(-1)^f$. As such it is localized on arguments related to the period as can be seen from

$$\begin{aligned} 2g(y) &= \sum_x (-1)^{f(x)+x\cdot y} + \sum_x (-1)^{f(x\oplus s)+(x\oplus s)\cdot y} \\ &= g(y)(1 + (-1)^{s\cdot y}). \end{aligned} \tag{5.4}$$

TABLE III. The bit-flip error frequency in the search for the target $|1111\rangle$ is well approximated by the binomial distribution.

	No. of errors				
	0	1	2	3	4
Frequency	0.06	0.26	0.38	0.25	0.06
Binomial ($n = 4$, $p_0 \approx 0.50$)	0.06	0.25	0.38	0.25	0.06

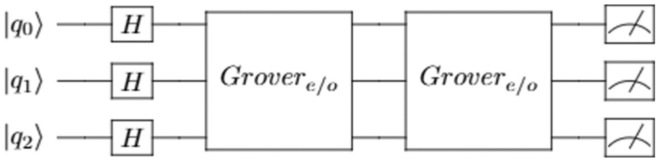


FIG. 6. The distributed circuits with $n = 3$ for the even and odd arguments.

It follows that

$$g(y) = w_y \delta(s \cdot y), \tag{5.5}$$

where $w_y \in \mathbb{Z}$ is the "weight" of the delta function. Inserting this into Eq. (5.3) gives for the state exiting the circuit

$$\frac{1}{N} \sum_{y_j \cdot s = 0} w_{y_j} |y_j\rangle . \tag{5.6}$$

The outgoing state is therefore a linear combination of solutions of $s \cdot y = 0 \mod 2$. The period is determined as the solutions of the linear system

$$y_j \cdot s = 0 \quad \mod 2, \quad w_{y_j} \neq 0. \tag{5.7}$$

The random outcomes of the measurement of the quantum circuits give y_j.

In the case where there are n independent vectors y_j, Eq. (5.7) has a unique solution, $s = 0$. In the case where $n - 1$ vectors are independent there is a unique nontrivial period s, etc. In the case where the sum in Eq. (5.6) has a single term w_0 for $y = 0$, Eq. (5.7) trivializes and any s is a solution: f is a constant.

Remark 5.1. The weights $\{w_y\}$ are (at most) N integers that satisfy Pythagoras,

$$\sum_{y \cdot s = 0} w_y^2 = N^2. \tag{5.8}$$

This imposes a (Diophantine) constraint on the allowed $\{w_y\}$ which is independent of the function f.

Example 5.1. With $N = 4$, the solutions $\{w_y\}$ of Eq. (5.8) are[5]

$$\{|4|\}, \quad \{|2|, |2|, |2|, |2|\}. \tag{5.9}$$

Not all of the solutions are realized as Fourier transforms of (the phases of) Boolean functions. In fact, there are 16 Boolean functions of 2 bits:

(i) 2 constant functions, $f(x_1, x_2) = 0$ and $f(x_1, x_2) = 1$, where s is arbitrary;

(ii) 6 balanced functions where 1 has two preimages—these have a single nontrivial period; and

(iii) the 4 functions where 1 has a single preimage and 4 where 1 has three preimages, which are aperiodic with $s = 0$.

The Fourier transforms of the phase functions $(-1)^f$ are

(i) $\pm 4\delta(y)$ for the constant functions;

(ii) $\pm 4\delta(y - j)$, $j \in 1, 2, 3$, for the balanced functions; and

(iii) $\pm 2(1 - 2\delta(y - j))$, $j \in 0, 1, 2, 3$, for the aperiodic.

Consider the distributed algorithm in the case of a unique $s \neq 0$. Suppose first that s is even (e.g., $s = 10$). The distributed Oracle reduces to the problem for $n - 1$ qubits for the even (odd) Oracles. This allows us to determine s after $O(n)$ queries.

In the case where s is odd, x and $x \oplus s$ have different parities. The $n - 2$ queries of the distributed algorithm will give the trivial result $2s = 0$. One then needs to try again with a different notion of even-odd, per Eq. (2.3). If the new notion of even-odd gives s even, the next $n - 2$ queries will determine s after a total of $3(n - 2)$ queries. If s is odd, we need to repeat the process. The complexity of an algorithm is determined by the worst case corresponding to n repetitions. This gives

$$O(n^2). \tag{5.10}$$

Remark 5.2. Similar arguments apply for the standard Simon algorithm for $f : \{0, 1\}^n \mapsto \{0, 1\}^n$ (which is represented by a quantum circuit acting on $2n$ qubits). Consider two quantum circuits each corresponding to the odd and even subspaces, defining functions $f_e, f_o : \{0, 1\}^{(n-1)} \mapsto \{0, 1\}^n$ (each represented by a quantum circuit acting on $2n - 1$ qubits). Following steps similar to those in the paragraph above, we find that the complexity in this case is also $O(n^2)$.

In summary, the complexity of the period finding of a phase Oracle is

(i) $O(N)$ classically, the cost of the $\mathbb{Z}_2^n$ Fourier transform;

(ii) $O(n)$ for the n-qubit quantum circuit; and

(iii) $O(n^2)$ for the distributed quantum circuit.

VI. DISTRIBUTED DEUTSCH-JOZSA ALGORITHM

A Boolean function f is called balanced if

$$\sum_x (-)^{f(x)} = \sum_x (1 - 2f(x)) = 0. \tag{6.1}$$

There are

$$\binom{N}{N/2}, \quad N = 2^n \tag{6.2}$$

balanced functions, a number which is superexponentially large. There are, of course, only two constant Boolean functions: $f \equiv 0$ and $f \equiv 1$.

The Deutsch-Jozsa task is, given the promise that f is either constant or balanced, determine which it is.

If no error is tolerated, one needs $N/2 + 1$ classical queries of f. If one is satisfied with a correct answer with a high probability, then a few queries suffice. Indeed, the probability that k random queries of a balanced function have the same

[5]In the sense that, for example, for the first set $w_y = |4|$, meaning either $w_y = +4$ or $w_y = -4$.

TABLE IV. Bit-flip error frequencies in the distributed search.

	No. of errors			
	0	1	2	3
Frequency	0.40	0.36	0.19	0.05
Binomial ($n = 3$, $p_0 \approx 0.30$)	0.35	0.44	0.19	0.03

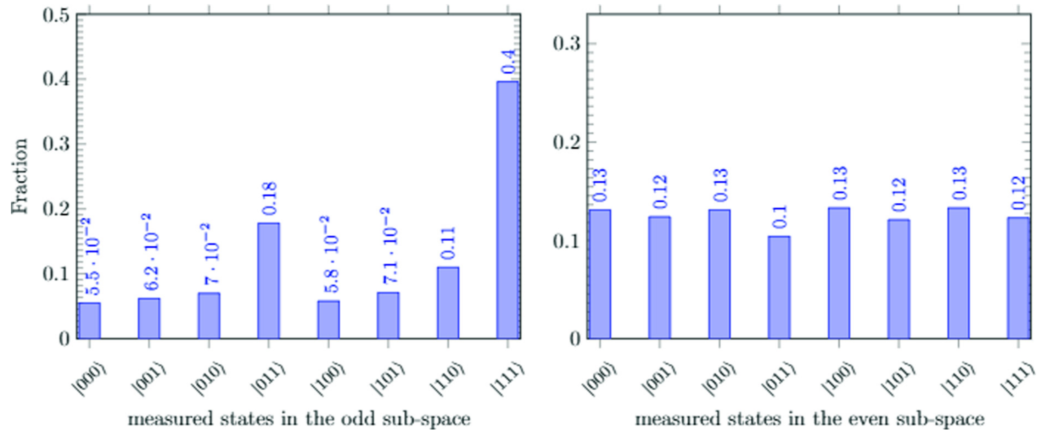


FIG. 7. The two histograms for the distributed search with $n = 3$ qubits. The sampling error is ± 0.011. The left histogram correctly identified $|111\rangle$. The right histogram did not single out any state, as it should.

image under f is[6]

$$p = \left(1 - \frac{N}{2N-1}\right)\dots\left(1 - \frac{N}{2N-k+1}\right). \tag{6.3}$$

When $k \ll N$ one has

$$p \approx \left(\frac{1}{2}\right)^k. \tag{6.4}$$

If one tolerates ϵ error probability,

$$k = O(\log 1/\epsilon) \tag{6.5}$$

queries suffice.

The Deutsch-Jozsa circuit in Fig. 8 outputs

$$\text{Prob}(y=0) = \left|\frac{1}{N}\sum_x (-1)^{f(x)}\right|^2 = \begin{cases} 1, & f \in \text{const}, \\ 0, & f \in \text{balanced} \end{cases} \tag{6.6}$$

and determines f, with no error, with a single query (assuming that the quantum gates are error-free and the f is indeed either balanced or constant).

Now consider the corresponding distributed Deutsch-Jozsa algorithm. The even and odd parts of a constant function are still constant functions. But the even and odd parts of a balanced function need not be (two) balanced functions. The distribution of 1 in the even function is the same as randomly drawing $N/2$ stones from two urns with $N/2$ stones each, all 1 or all 0. The probability distribution for finding k 1's in the even sequence is

$$\text{Prob}(k) = \binom{N/2}{k}\binom{N/2}{N/2-k}\Big/\binom{N}{N/2}. \tag{6.7}$$

We are interested in the probability that the distributed circuit will (mis)identify a balanced function as constant. By Eq. (6.6) this is related to the expectation of

$$\left|\frac{1}{N}\sum_x (-1)^{f_e(x)}\right|^2 = \left(1 - \frac{2}{N}\sum_x f_e(x)\right)^2$$
$$= 1 - \frac{4}{N}\sum_x f_e(x) + 4\left(\frac{1}{N}\sum f_e(x)\right)^2. \tag{6.8}$$

Evidently

$$\mathbb{E}\left(\sum_x f_e(x)\right) = \sum_k k\,\text{Prob}(k) = \frac{N}{4}. \tag{6.9}$$

As $f_e(x)$ and $f_e(y)$ are independent for $x \neq y$ and $f_e^2(x) = f_e(x)$ we also have

$$\mathbb{E}\left(\sum_x f_e(x)\right)^2 = \mathbb{E}\left(\sum_{x,y} f_e(x) f_e(y)\right) = \mathbb{E}\left(\sum_x f_e(x)\right) = \frac{N}{4}. \tag{6.10}$$

It follows that the expectation values that a single query of the distributed circuit will make the mistake of identifying a balanced f as constant is

$$\mathbb{E}(y=0|f=\text{balanced}) = \frac{1}{N}. \tag{6.11}$$

Comparing with Eq. (6.5) we see that one needs $O(n)$ classical queries to get the same margin of error as a single quantum query.

In summary, the complexity of the Deutsch-Jozsa problem is

(i) $1 + N/2$ classical queries for a deterministic result,
(ii) a single quantum query (for an error-free circuit),

[6]The formula follows from repeated application of the fact that in an urn with W white stones and B black stones, the probability of picking a black stone is $B/(B+W)$.

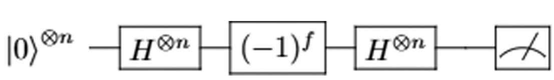


FIG. 8. The Deutsch-Jozsa and the period-finding circuits.

(iii) a single query of a distributed quantum circuit for $O(1/N)$ error, and

(iv) $O(n)$ classical queries for $O(1/N)$ error.

The first two entries imply an $O(N)$ quantum advantage and the last two entries an $O(n)$ advantage of distributed quantum computing.

VII. CONCLUSION

Distributed quantum computing with classical communication on ideal devices is, in general, inferior to undistributed quantum computing. However, in the context of the currently available noisy small computers that offer no error corrections, distributed computing offers two advantages: amalgamated qubit resources and shallow circuits with significant noise reduction.

ACKNOWLEDGMENTS

We thank Eyal Bairey, Shay HaCohen-Gourgy, Oded Kenneth, and Netanel Lindner for helpful discussion and E.B. for pointing out Ref. [4]. We thank the referees for pointing out Refs. [3] and [6].

APPENDIX A: THE DNF OF QUANTUM CIRCUITS

For the sake of simplicity and concreteness we illustrate an algorithm that works for any Boolean function $f:\{0,1\}^n \to \{0,1\}^m$ by considering the special case where $n=2$ and $m=1$.

1. From the truth table to the DNF [14]

The disjunctive normal form [15] of a Boolean function can be calculated directly from the truth table of the function. Consider the following.

x_0	x_1	$f(x)$
0	0	1
0	1	0
1	0	1
1	1	0

The DNF form of f is given by

$$f_{\rm DNF}(x) = \bar{x}_0 \cdot \bar{x}_1 + x_0 \cdot \bar{x}_1, \tag{A1}$$

where $\bar{x}$ denotes NOT(x).

We only need to consider the rows where $f(x)=1$. For every such row we build out of the arguments x_0 and x_1 a Boolean statement made of only NOT and AND operations (conjunctive) so that an argument that takes the value 1 is written as is, while an argument with value 0 is negated. For the example,

(i) the first row is described by $\bar{x}_0 \cdot \bar{x}_1$ and

(ii) the third row is described by $x_0 \cdot \bar{x}_1$.

Generalization: In the case of general n do the same with n variables. For a different m repeat the process for each $f_i(x)$.

The DNF extracted from the truth table is not, in general, the simplest DNF formula of the function. For example, f in Eq. (A1) can be written more simply as

$$f_{\rm DNF} = \bar{x}_1.$$

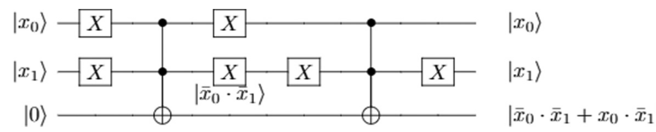


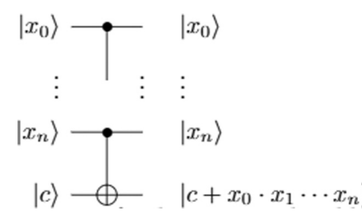
FIG. 9. The DNF circuit for the function gate of Eq. (A1).

The optimal DNF formula can be calculated using Karnaugh maps [16].

2. From the DNF to the quantum circuit [17]

The quantum circuit for any $f:\{0,1\}^n \to \{0,1\}^m$ can be built with X gates and $C^{\otimes n}X$ gates. To see this recall that the Toffoli gate [18] gives the conjunction of its arguments $f_T(x_0,x_1) = x_0 \cdot x_1$. The quantum circuit for calculating the function in Eq. (A1) is shown in Fig. 9. (It can be simplified using $XX \equiv \mathbb{1}$.)

Generalization: For the case of n conjuctions, a similar construction works with $C^{\otimes n}X$ replacing the Toffoli.



For Boolean functions whose output is a string of m bits one simply adds additional target qubits.

3. Phase Oracles

The same method can be used to construct the phase Oracle for $(-1)^{f(x)}$. This is done by replacing $C^{\otimes n}X$ with $C^{\otimes(n-1)}Z$ gates; for example, the circuit for the phase gate of the function in Eq. (A1) is shown in Fig. 10.

APPENDIX B: THE EXPERIMENT: ADDITIONAL DETAILS

1. Splitting the Grover Oracle

Alice and Bob each get the same n-qubit Oracle. Alice uses the Oracle as is. Bob uses the following algorithm to split the Oracle to the two computers B_1 and B_2.

Algorithm B.1: Distributed Grover algorithm.

Result: The binary index of a desired element from the whole set.

```
BobGrover (U_f)
    U_odd, U_even ← Divide(U_f) ;
    res_1 ← Grover(B_1, U_even) ;
    res_2 ← Grover(B_2, U_odd) ;
    if U_even(res_1) == 1 then
        return res_1 + '0' ;
    return res_2 + '1' ;
```

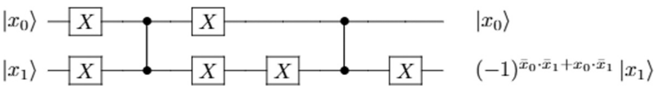


FIG. 10. Phase gate for the function in Eq. (A1).

2. A depolarizing channel model for the errors

A simple model of the noise in circuits is in terms of depolarizing channels:[7]

$$\rho \mapsto \lambda\rho + \frac{1-\lambda}{2}\mathbb{1}, \quad -1/3 \leqslant \lambda \leqslant 1. \tag{B1}$$

The probabilities that a qubit $|\psi\rangle$ gives the correct (1) or incorrect (0) answer are

$$p_0 = \frac{1-\lambda}{2}, \quad p_1 = \frac{1+\lambda}{2}. \tag{B2}$$

Assuming independence, the probability of m bit-flip errors in an n-qubit register is

$$\binom{n}{m} p_0^m p_1^{n-m}. \tag{B3}$$

The expected number of errors in the register is

$$np_0 \tag{B4}$$

and the probability of the correct answer in all n qubits is

$$\left(\frac{1+\lambda}{2}\right)^n. \tag{B5}$$

3. Decoherence and gate errors

The decoherence associated with each time step is represented by a polarizing channel with $\lambda \mapsto \mu_c$, and the gate errors by a polarizing channel with $\lambda \mapsto \mu_g$. The composition of polarizing channels is ordinary multiplication. It follows that, for a circuit with T time steps and N_g noisy gates, there is a channel where

$$\lambda \approx \mu_c^T \mu_g^{N_g}. \tag{B6}$$

Since the dominant error is in the two-qubit gates, N_g is the number of two-qubit gates.

From Table I

$$\log \mu_c = -\frac{R}{T_c} \approx -3\times 10^{-3} \Longrightarrow \mu_c \approx 0.997,$$
$$\mu_g = 1-\epsilon \approx 0.994. \tag{B7}$$

An estimate for λ is then[8]

$$\lambda \approx (0.997)^T \cdot (0.994)^{N_g}. \tag{B8}$$

This allows us to compare the experimentally observed probability of bit-flip error p_0 computed via Eq. (B4) and the data in Tables III and IV with the expected bit-flip error $\bar{p}_0$ computed via Eqs. (B2) and (B8) using the IBM machine data in Table I (see Table V). The qualitative agreement between p_0 and $\bar{p}_0$ gives some support to the noise model as the depolarizing channel.

TABLE V. The bit-flip errors p_0 computed from the histogram and $\bar{p}_0$ computed from the IBM machine data.

Qubits	p_0	$\bar{p}_0$	λ
4	0.50	0.47	0.05
3 (odd)	0.30	0.23	0.54

[7]The condition on λ guarantees complete positivity [19].

[8]Since $T, N_g = O(1000)$ we need to keep three significant figures.

[1] See https://quantum-computing.ibm.com/.
[2] See https://aws.amazon.com/braket/.
[3] S. Bravyi, G. Smith, and J. A. Smolin, Trading Classical and Quantum Computational Resources, Phys. Rev. X **6**, 021043 (2016).
[4] V. Dunjko, Y. Ge, and J. I. Cirac, Computational Speedups Using Small Quantum Devices, Phys. Rev. Lett. **121**, 250501 (2018).
[5] J. I. Cirac, A. K. Ekert, S. F. Huelga, and C. Macchiavello, Distributed quantum computation over noisy channels, Phys. Rev. A **59**, 4249 (1999).
[6] T. Peng, A. W. Harrow, M. Ozols, and X. Wu, Simulating Large Quantum Circuits on a Small Quantum Computer, Phys. Rev. Lett. **125**, 150504 (2020).
[7] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC), May 1996* (Association for Computing Machinery, New York, 1996), pp. 212–219.
[8] P. W. Shor, Introduction to quantum algorithms, in *Proceedings of Symposia in Applied Mathematics* (American Mathematical Society, Providence, RI, 2002), Vol. 58, pp. 143–160.
[9] D. R. Simon, On the power of quantum computation, SIAM J. Comput. **26**, 1474 (1997).
[10] D. Deutsch and R. Jozsa, Rapid solution of problems by quantum computation, Proc. R. Soc. London Ser. A **439**, 553 (1992).
[11] J.-H. Bae, P. M. Alsing, D. Ahn, and W. A. Miller, Quantum circuit optimization using quantum Karnaugh map, Sci. Rep. **10**, 15651 (2020).
[12] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A **52**, 3457 (1995).
[13] See https://qiskit.com/.
[14] D. Hilbert and W. Ackermann, *Principles of Mathematical Logic* (American Mathematical Society, Providence, RI, 1999).

[15] B. A. Davey and H. A. Priestley, *Introduction to Lattices and Order* (Cambridge University Press, Cambridge, UK, 1990), p. 153.

[16] M. Karnaugh, The map method for synthesis of combinational logic circuits, Trans. Am. Inst. Electr. Eng., Part 1 **72**, 593 (1953).

[17] Yu. I. Bogdanov, N. A. Bogdanova, D. V. Fastovets, and V. F. Lukichev, Representation of Boolean functions in terms of quantum computation, in *Proceedings Vol. 11022. SPIE International Conference on Micro- and Nano-Electronics 2018* (SPIE, Bellingham, WA, 2019).

[18] T. Toffoli, Reversible computing, in *Automata, Languages and Programming*, edited by J. de Bakker and J. van Leeuwen (Springer, Berlin, 1980), pp. 632–644.

[19] https://en.wikipedia.org/wiki/Quantum_depolarizing_channel.