Quantum network optimization

Alexandre Blais*
Département de Physique and Centre de Recherche sur les Propriétés Électroniques de Matériaux Avancés, Université de Sherbrooke, Sherbrooke, Québec, Canada J1K 2R1

(Received 21 February 2001; published 13 July 2001)

In many candidate designs of solid-state quantum computers, interactions between qubits are limited to a small number of neighboring qubits. Taking into account this limitation we describe how quantum algorithms can be executed efficiently on these designs. We illustrate our results with the quantum Fourier transform for which linear depth networks are obtained. The concepts presented in this work can be applied to all quantum algorithms to reduce considerably the coherence time needed for their execution.

DOI: 10.1103/PhysRevA.64.022312 PACS number(s): 03.67.Lx

Since the presentation by Shor [1] of an efficient quantum algorithm for factorization there has been much interest in the development of quantum-information theory. In particular, a lot of attention has been focused on developing other efficient quantum algorithms [2–4] and quantum error correction codes [5–7].

In parallel, great effort is now invested in the design of physical implementations meeting the very stringent requirements needed for the coherent manipulation of quantum information [8]. The high level of expertise available in solid-state based technologies establishes this approach as a leading candidate for the realization of a useful (several thousand qubits) quantum computer. Several designs have already been proposed: Josephson junctions [9–12], quantum dots [13], and spin-resonance transistors [14]. Recent experimental successes [15,16] give good confidence that a practical quantum computer resting on those approaches will be built.

However, due to their large number of degrees of freedom, solid-state designs suffer from short coherence times. In order to take full advantage of their computational power, it will be important to optimize the algorithms (computation and error correction) for the specific quantum hardware in use.

Moreover, in most solid-state designs, it will be experimentally simpler to build arrays of qubits with qubit-qubit interactions limited to a small number of neighboring qubits. In this paper, we address the question of optimization of quantum algorithms for the most limiting but also experimentally more realistic case of qubits coupling: an unidimensional array of qubits with nearest-neighbor interactions. We will take advantage of the possibility of performing operations on different qubits simultaneously that is common to many designs. To illustrate our point, we present results for the important case of the quantum Fourier transform (QFT) for which, even under the strong constraint of only nearest-neighbor interactions, we obtain depth $O(n)$ networks.

Optimization of quantum networks by parallelization has already been investigated in references [17] and [18] but without taking into account the limited extent of qubit-qubit interactions. A study of the ''actual computational time cost'' of the quantum Fourier transform including such a limitation has been presented in Ref. [19], but many possible improvements, including parallelism, were omitted.

We begin by first reviewing the quantum Fourier transform. This transformation acts as follows on a n-qubit register indexed $|n-1,n-2,\ldots,1,0\rangle$

$$F_n:|x\rangle\mapsto\frac{1}{2^{n/2}}\sum_{y=0}^{2^n-1}e^{2\pi ixy/2^n}|y\rangle. \tag{1}$$

This evolution relies on two logical gates: a one-qubit gate A_j acting on qubit j

$$A_j=\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}, \tag{2}$$

and a two-qubit gate B_{jk} acting on qubits j and k

$$B_{jk}=\begin{pmatrix}1 & & & \\ & 1 & & \\ & & 1 & \\ & & & e^{i\theta_{jk}}\end{pmatrix} \tag{3}$$

with $\theta_{jk}=\pi/2^{k-j}$. To perform the Fourier transform of a five-qubit register F_5 the following sequence of A_j and B_{jk} gates is applied

$$A_4B_{34}B_{24}B_{14}B_{04}A_3B_{23}B_{13}B_{03}A_2B_{12}B_{02}A_1B_{01}A_0. \tag{4}$$

The corresponding network is shown in Fig. 1. F_5 is thus realized by 15 logical gates. More generally, n one-qubit and $n(n-1)/2$ two-qubit operations are necessary to implement F_n.

To proceed with optimization of this algorithm, an explicit prescription for the implementation of A and B in terms of a universal set of elementary gates is needed. Of course this restricts the optimization to implementations having this particular set of elementary gates in their repertory, but application to other implementations is straightforward. Here we will use the universal set

$$X_j(\theta)=e^{-i\sigma_x^j\theta/2},\quad Z_j(\phi)=e^{-i\sigma_z^j\phi/2}, \tag{5}$$

*Electronic address: ablais@physique.usherb.ca

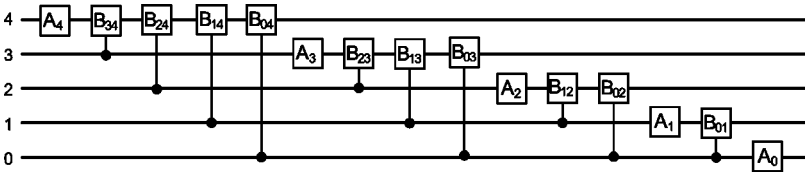

FIG. 1. Network realizing the quantum Fourier algorithm on a five-qubit register. The output is the bit reversal of the Fourier transform Eq. (1).

$$P_{jk}(\zeta)=\exp(-i\sigma_z^j\otimes\sigma_z^k\zeta/2). \tag{6}$$

acting on qubits j and k. This set is useful as it corresponds to the elementary set of gates of many solid-state designs [10,11,14] (and can be implemented by nuclear magnetic resonance [20,21]).

These elementary operations can be used to implement the logical gates A_j and B_{jk} (on two adjacent qubits) in the following fashion

$$A_j=iZ_j(\pi/2)X_j(\pi/2)Z_j(\pi/2), \tag{7}$$

$$B_{jk}=e^{i\theta_{jk+2}}Z_j(\theta_{jk+1})Z_k(\theta_{jk+1})P_{jk}(\theta_{jk+1}). \tag{8}$$

The phase $e^{i\theta_{jk+2}}$ depends on the qubits on which B_{jk} is applied but is independent of their state. It is thus an unimportant global phase factor and can be ignored.

To realize the network of Fig. 1 it will be necessary to apply B_{jk} gates on nonadjacent qubits. In a unidimensional array of qubits limited to nearest-neighbor couplings this will entail swapping recursively the state of adjacent qubits in order to juxtapose the state of qubits to couple. For quantum bits initially at locations j and k in a quantum register, $|j-k|-1$ swap operations are required to bring the qubits together and another $|j-k|-1$ to bring them back to their original locations, Fig. 2(a).

A swap between the state of qubits at positions r and s, respectively, can be implemented by a sequence of controlled-NOT (C) gates

$$S_{rs}=C_{rs}C_{sr}C_{rs}. \tag{9}$$

Using Eqs. (5) and (6), the C gate is itself implemented by a sequence of seven elementary gates

$$C_{rs}=e^{-i3\pi/4}X_s(3\pi/2)P_{rs}(-\pi/2)Z_s(\pi/2)X_s(\pi/2)Z_s(\pi/2)\times Z_r(\pi/2)P_{rs}(-\pi/2). \tag{10}$$

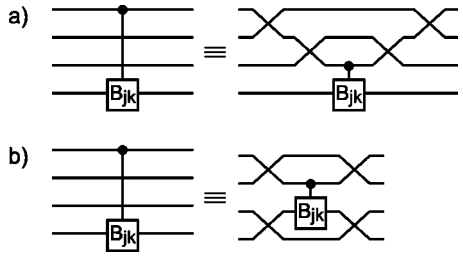

FIG. 2. (a) Repetitive use of swap operations to perform a controlled interaction between initially nonadjacent qubits. (b) Equivalent network using simultaneous operations. Strategy (a) was used in Ref. [19] to estimate the time cost of F_n.

As a result, a single swap operation requires 21 elementary operations (time steps) and $42\times(|j-k|-1)$ elementary operations are required to realize the network of Fig. 2(a).

It is however possible to reduce this number significantly. First, by taking advantage of the commutation relations between the elementary gates (5) and (6) and the symmetry of the controlled-NOT (10) it is possible to ''compile'' the sequence (9) by removing redundant gates. Doing so, we obtain the following swap sequence implemented by 15 elementary operations

$$\begin{aligned}S_{rs}=e^{-i\pi/4}&X_s(\pi/2)P_{rs}(-\pi/2)Z_s(\pi/2)X_s(\pi/2)\\&\times[Z_s(\pi/2)X_r(\pi/2)]P_{rs}(-\pi/2)Z_r(\pi/2)\\&\times[X_r(\pi/2)X_s(\pi/2)]P_{rs}(-\pi/2)Z_s(\pi/2)X_s(\pi/2)\\&\times[Z_s(\pi/2)Z_r(\pi/2)].\end{aligned} \tag{11}$$

Moreover, taking advantage of the ability to perform operations on different qubits simultaneously, it is possible to further reduce the number of time steps used for moving states through the register. Indeed, since operations on different qubits commute, the gates in square brackets in Eq. (11) can be performed simultaneously, reducing the number of time steps to 12. In addition, the simultaneous swaps in Fig. 2(b) requires only $2\lceil|j-k|/2\rceil$ time steps, where $\lceil x\rceil$ is the smallest integer larger than x, and is thus much more efficient than the straightforward implementation of Fig. 2(a). A final simplification is possible when we notice that there is no need to move the qubits back to their original locations. Once the states of a pair of qubits have been brought together and have interacted, the next reorganization should be done in a way optimizing the realization of the following logical operations.

With these results, the number of time steps is reduced from $42(|j-k|-1)$ to $12\lceil|j-k|/2\rceil$ for a single swap sequence. In the case of networks involving multiple swaps, further improvements are possible by reorganizing the qubits properly during computation.

While the above optimization for swap gates is relevant to the implementation of all quantum algorithms in one-dimensional (1D) arrays of qubits, a key observation specific to the quantum Fourier transform is

$$[A_l,B_{jk}]=0 \quad \text{if} \quad l\neq j,k,$$

$$[B_{jk},B_{rs}]=0 \quad \forall \quad j,k,r,s. \tag{12}$$

The first of these relations express the fact that gates acting on distinct qubits commute while the second relation holds because B_{jk} is diagonal in the computational basis. While

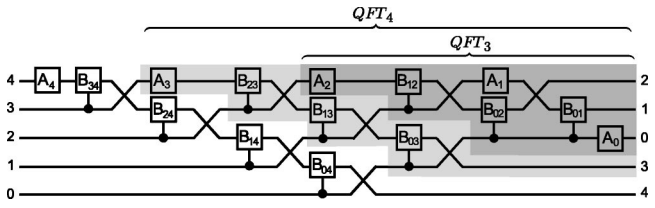

FIG. 3. Network realizing the quantum Fourier algorithm on a five-qubit register using simultaneous operations and efficient use of the swap operations. The numbers at both end of the circuit are used to keep track of the position of the logical states. The operations in the dark gray area implement F_3.

respecting these commutation relations, it is now possible to permute logical operations and apply them when it is most convenient.

Let us now apply these concepts. For this purpose, we developed an algorithm adding swaps, as suggested in Fig. 2(b), to the original QFT sequence. To minimize the length of the networks, the algorithm then maximizes parallelism by grouping operations that commute while not acting on the same qubits. The result, for five qubits, is given by the network of Fig. 3. We compute the depth of our networks in terms of the elementary gates implementing each logical gates. Recognizing that operations B_{jk} are realized in two time steps [all elementary gates realizing this operation, Eq. (8), commute and it is possible to apply the first two simultaneously] while A_j in 3 we obtain that the network of Fig. 3, is implemented in 95 time steps [23]. Without optimization, F_5 would necessitate 20 swaps for a total of 275 time steps.

Figure 4 compares, on a logarithmic scale, the time cost of the improved networks obtained numerically (black circles) with nonimproved networks (open squares) [24] for up to 300 qubits. The time cost of the later networks is easily determined to be $10n-11n^2+4n^3\approx O(n^3)$.

To go beyond numerical constructions, a useful observation is that the dark gray region at the rightmost end of Fig. 3 is an optimized quantum Fourier transform on three qubits. Moreover, adding to this transformation the four logical gates

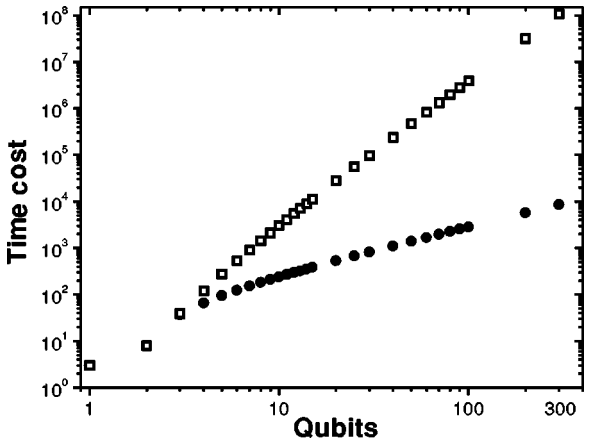

FIG. 4. Number of time steps for QFT networks as a function of the number of qubits on a logarithmic scale. Black circles are for the improved networks obtained numerically while open squares correspond to nonimproved networks.

and three swaps which are in the light shaded area of the same figure, we now obtain an optimized QFT on four qubits. These extra gates add 29 time steps to F_3 [25]. The same observation also holds in going from F_4 to F_5 by adding the five logical operations and four swaps that are on the left of the light shaded area, adding again 29 time steps. By using this construction recursively, we can now obtain an optimized network for F_n once the network for F_{n-1} is given. As seen from the above constructions, this is easily done by adding n logical operations and $n-1$ swaps to F_{n-1} (for $n>2$). These $O(n)$ extra gates add 29 time steps to F_{n-1} and the number of time steps required to perform F_n in this fashion can then be evaluated to be $8+29(n-2)$ for $n>2$. Hence, we have obtained linear depth networks for F_n.

Thus, while taking into account the limited range of nonlocal interactions between qubits we have obtained a speedup quadratic in n over nonoptimized networks. Efficient use of swaps and massive (classical) parallelism are responsible for this speedup. Indeed, speedup by a factor of $O(n)$ can be seen to come from the fact that $O(n^3)$ swaps are necessary in nonoptimized circuits while only $O(n^2)$ are requested in the optimized case. The other factor of $O(n)$ comes from the fact that up to n simultaneous operations can be realized on n qubits. As in the case of classical parallel computers, this provides a speedup of order $O(n)$.

Interestingly, linear depth networks were also obtained in Ref. [17] in the simpler case where no limitations in the range of coupling between qubits was considered [26]. In fact, depth $O(n)$ networks is the best one can achieve for the exact F_n network given by sequence (4) since n operations have to be applied on the nth [top most in Fig. (1)] qubit and no operations acting on similar qubits can be parallelized. As a result, the limitations of a linear design restricted to nearest-neighbor interactions, at least for the exact Fourier transform, does not seem to be an important one as long as parallelism is possible. A circuit similar to Fig. 3 was published in a different context in [22].

It is possible to shorten further, by a constant factor, the length of QFT networks by starting the optimization procedure with a permutation [respecting relations (12)] of the original sequence (4). We then seek the permutation minimizing the depth. This is a constrained optimization problem and has many similarities with the problem of placement occurring in very large scale integrated circuit related technologies for which heuristics, like simulated annealing or tabu search, are known to give good results. In placement, one seeks to arrange the components of a (classical) circuit such as to minimize the length of interconnecting wires and the circuit area [27].

The problem at hand is very similar but with the additional complication that reordering two logical operations at a given location in a circuit will change the sequence and possibly the number of swaps needed at all further points in this circuit. Based on this analogy, we developed a simulated annealing (SA) algorithm [28] to obtain improved networks. As SA is a heuristic algorithm, it will not necessarily provide optimal solutions, but solutions that are close to the optimal ones. This is not a problem here as we will be satisfied with any improvements over straightforward networks.

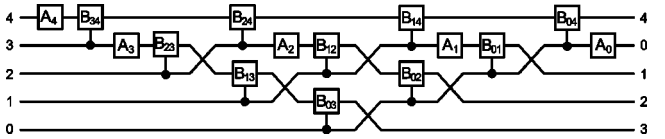

FIG. 5. Improved quantum Fourier network on five qubits.

Networks obtained with this approach are only a few percent shorter than the best networks (black dots) of Fig. 4. Moreover, SA is computationally demanding and becomes impractical for a few tens of qubits. This further speedup will nevertheless be welcomed for prototypical quantum computers. The network of Fig. 5 was obtained by improving upon the circuit of Fig. 3 in this way. It requires only six swaps and 83 time steps.

Finally, we note that for very large networks, it is experimentally challenging to implement the operations B_{jk} for large $|j-k|$ as this corresponds to very small phases and will require the application of elementary gates for short periods of times. It is however possible to omit the B_{jk} such that $\pi/2^{k-j}<\pi/2^m$ for a chosen m and obtain a result that differs only by a multiplicative factor $e^{i\varepsilon}$, with $|\varepsilon|\leqslant 2\pi n/2^m$, from the result of the original F_n network [29]. Combined with the optimization procedure presented here, this will yield networks that are still more efficient, but at the expense of losing some accuracy. This was explored recently by Cleve and Watrous [18] in the case of arbitrarily long qubit-qubit interactions for which they obtained depth $O(\log_2 n)$ networks. We note that for the unidimentional architecture studied in this paper, the limitation of nearest-neighbor coupling would not allow for the implementation of the approximate QFT in logarithmic depths.

In summary, we considered several improvements that should be applied in the implementation of quantum algorithms. To illustrate our point, we presented results for the important case of the quantum Fourier transform. These improvements were obtained by first expressing the necessary gates in terms of an elementary set of gates. Knowing the number of time steps required to realize each member of this set, we then minimized the length of quantum Fourier networks by taking advantage of (classical) parallelism and of the commutation relations between logical operations. While limiting the spatial extent of coupling between qubits, we obtained circuits that can be implemented in $O(n)$ time steps corresponding to a quadratic speedup for time resources.

Two main conclusions can be drawn from this work: (i) Quantum algorithms can be optimized by prior classical computation to yield networks necessitating much smaller coherence times. (ii) As long as they allow for simultaneous operations of distinct qubits, solid state quantum computers can be designed with very simple qubit-qubit interaction schemes without degrading their performance, at least for some computational tasks. From the results obtained in this work, parallelism seems like a very desirable feature for candidate quantum computer implementations. These conclusions are significant for the design and use of prototypical solid-state quantum computers.

We stress that the concepts presented here are applicable to all quantum algorithms and can be generalized to other quantum computer architectures and geometries (2D arrays of qubits, quasi-1D arrays, etc.) and to the case where interactions are not restricted to neighboring qubits but nevertheless have limited spacial extents [9,13]. Moreover, the results obtained here can be improved by working directly with the elementary gates implementing the logical operations rather than with the logical operations themselves. As seen from Eqs. (9) and (11), this certainly can yield further improvements.

Helpful discussions with S. Lacelle, A.-M. Tremblay, J.-F. Veillette, A.M. Zagoskin, and particularly with M. Beaudry on circuit complexity are acknowledged. This work was supported in part by NSERC, D-Wave Systems Inc. and FCAR.

[1] P.W. Shor, SIAM J. Comput. **26**, 1484 (1997).
[2] L.K. Grover, Phys. Rev. Lett. **79**, 325 (1997).
[3] D.S. Abrams and S. Lloyd, Phys. Rev. Lett. **83**, 5162 (1999).
[4] B.M. Terhal and D.P. DiVincenzo, Phys. Rev. A **61**, 062312 (2000).
[5] P.W. Shor, Phys. Rev. A **52**, R2493 (1995).
[6] R. Laflamme *et al.*, Phys. Rev. Lett. **77**, 198 (1996).
[7] A.M. Steane, Nature (London) **399**, 124 (1999).
[8] D.P. DiVincenzo, Fortschr. Phys. **48**, 771 (2000).
[9] A. Shnirman *et al.*, Phys. Rev. Lett. **79**, 2371 (1997).
[10] L.B. Ioffe *et al.*, Nature (London) **398**, 679 (1999).
[11] A. Blais and A.M. Zagoskin, Phys. Rev. A **61**, 042308 (2000).
[12] J.E. Mooij *et al.*, Science **285**, 1036 (1999).
[13] D. Loss and D.P. DiVincenzo, Phys. Rev. A **57**, 120 (1998).
[14] B.E. Kane, Nature (London) **393**, 133 (1998).
[15] Y. Nakamura *et al.*, Nature (London) **398**, 786 (1999).
[16] J.R. Friedman *et al.*, Nature (London) **406**, 43 (2000).
[17] C. Moore and M. Nilsson, SIAM J. Comput. (to be published); (e-print quant-ph/9808027).
[18] R. Cleve and J. Watrous, e-print quant-ph/0006004.
[19] A. Saito *et al.*, e-print quant-ph/0001113.
[20] D.G. Cory *et al.*, Proc. Natl. Acad. Sci. U.S.A. **94**, 1634 (1997).
[21] N.A. Gershenfeld and I.L. Chuang, Science **275**, 351 (1997).
[22] R.B. Griffiths and C. Niu, Phys. Rev. Lett. **76**, 3228 (1996).
[23] In evaluating the time cost of a given network, when simultaneous operations are performed, the number of steps of the most time consuming operation is used.
[24] Nonimproved networks do not take advantage of parallelism and use the strategy of Fig. 2(a) for swapping.
[25] The logical gates A_3, B_{23} and the first two swaps of the light shaded area of Fig. 3 are not done in parallel with any gates of F_3. These extra operations take 29 time steps to be executed.
[26] In terms of the logical operations (and not of the number of time steps it takes to implement them), the networks obtained here have depths $4(n-3)+7$ (for $n>2$) and, those of Ref. [17], depths $2n-1$. The depths obtained here are then only asymtotically twice those obtained in the less physically

relevant case of arbitrarily long range interactions *and* parallelism.

[27] S.M. Sait and H. Youssef, *VLSI Physical Design Automation* (IEEE Press, New York, 1995).

[28] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines* (Wiley, New York, 1989).

[29] D. Coppersmith, IBM Research Report No. RC19642, 1994 (unpublished).