Optimization of Circuits for IBM's five-qubit Quantum Computers

Gerhard W. Dueck[†], Anirban Pathak[*], Md Mazder Rahman[†], Abhishek Shukla[*], and Anindita Banerjee[*]
[†]Faculty of Computer Science, University of New Brunswick, Canada
[*]Department of Physics and Material Science Engineering, Jaypee Institute of Information Technology, Noida, INDIA

***Abstract*— IBM has made several quantum computers available to researchers around the world via cloud services. Two architectures with five qubits, one with 16, and one with 20 qubits are available to run experiments. The IBM architectures implement gates from the Clifford+T gate library. However, each architecture only implements a subset of the possible CNOT gates. In this paper, we show how Clifford+T circuits can efficiently be mapped into the two IBM quantum computers with 5 qubits. We further present an algorithm and a set of circuit identities that may be used to optimize the Clifford+T circuits in terms of gate count and number of levels. It is further shown that the optimized circuits can considerably reduce the gate count and number of levels and thus produce results with better fidelity.**

***Index Terms*—Reversible Logic, Logic Synthesis, IBM Quantum Computer, Quantum Circuit, and Circuit Optimization.**

I. Introduction

Interest in quantum computing has received a boost with the availability of IBM Quantum Computers (www.research.ibm.com/ibm-q) that enables users to run quantum experiments. Recently, various computational tasks (e.g., Bell state discrimination [1], teleportation using optimal quantum resources [2], quantum permutation algorithm [3], quantum cheque [4], testing Mermin inequalities [5], etc.) have been realized using IBM Quantum Computers. The IBM Quantum Computers work for a limited number of qubits (5, 16, or 20) and one of the problem with the current implementation is fidelity (i.e., noise in the output states) [6]. There is a direct (although not linear) correlation between the number of gates and the fidelity of the experimentally obtained output state and the desired output state which would have been obtained in the ideal scenario. In fact, in Ref. [1] and other recent works using IBM quantum computers it is clearly observed that the state fidelity reduces considerably with the increase in the gate count. Further, because of this fact, each IBM quantum computer imposes a bound on the maximum number of gates that can be used in a single experiment. It is therefore imperative to keep the number of gates in circuits a minimum. The objective of the present work is to design mapping algorithms for IBM quantum computers and to obtain optimized IBM circuits for some computational tasks of particular interest. We obtain several equivalent circuits for the same tasks and compared their gate count and number of levels.

The rest of the paper is structured as follows: Section II briefly describes the fundamentals of quantum computing and basic steps of synthesizing quantum circuits from Boolean functions. Section III illustrates mapping CNOT gates into IBM's QX2 and QX4 Architectures. Section IV presents an optimization method with examples. The significance of the proposed method is shown with experiments in Section V. Some additional ideas on how to further reduce the number of gates, are presented in Section VI. The paper concludes with some observations and directions for future research in Section VII.

II. Background

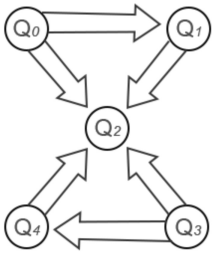
Quantum systems perform logic operations according to the principles of quantum mechanics that are entirely different from their classical counterpart [7]. The fundamental unit of information in quantum computation is the qubit. The state of a qubit is represented by a vector in a two-dimensional complex vector space [7] and is expressed as follows:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \quad (1)$$

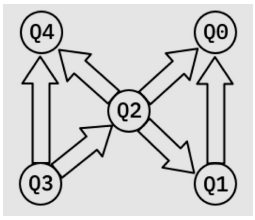
The coefficients α and β are complex numbers called probability amplitudes that satisfy the constraint $|\alpha|^2 + |\beta|^2 = 1$. The states $|0\rangle$ and $|1\rangle$ are known as computational basis states that are analogous to the states 0 and 1 respectively of a classical bit. The states of a qubit can be the superposition of the states $|0\rangle$ and $|1\rangle$. With the superposition of states, an infinite state space can be found in the quantum computation [7]. Time evolution of a quantum system (transition between states) is described by a unitary transformation. Quantum gates are unitary matrices and operations are inherently reversible—except for measurements and the effect of noise. IBM architectures implement quantum circuits with the Clifford+T gate library. However, it is very difficult to directly implement logic functions in quantum circuits with this library. Hence, synthesis of logic functions into IBM architectures can be generally viewed as a series of the following steps:

1) Transform the given Boolean function into a reversible one (Multiple controlled Toffoli gates are most frequently used here [8]);
2) Decompose the Toffoli circuit into gates from the Clifford+T gate library [9], [10].
3) Map the gates to one of the IBM architectures.

In this paper, we address the last point. It should also be noted that some quantum algorithms will not start with a Boolean specification. However, they will also need the last step in the above flow.



(a) QX2 [13].



(b) QX4 [14].

Figure 1: Architectures of the IBM's five-qubit computers.

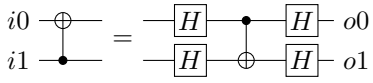
III. Mapping CNOT gates to IBM's QX2 and QX4 Architectures

IBM's 5-qubit quantum computers (QX2 and QX4) [11] support gates from the Clifford+T gate library [12]. Not all CNOT gates can be implemented directly, due to their architectures (shown in Figure 1a and 1b). The arrows shown in these figures indicate which CNOT gates are implemented. Specifically, a qubit shown at the tail of the arrow can only work as control qubit and the qubit shown at the head of the arrow can only work as target qubit. Thus, in QX2, a CNOT can be implemented between Q3 and Q4, with Q3 as control qubit and Q4 as the target qubit, but the opposite cannot be done directly. Further, no CNOT gate can be implemented directly between Q1 and Q3 as they are not connected. However, all CNOT gates can be implemented with the aid of some additional gates. The two architectures are considered in turn.

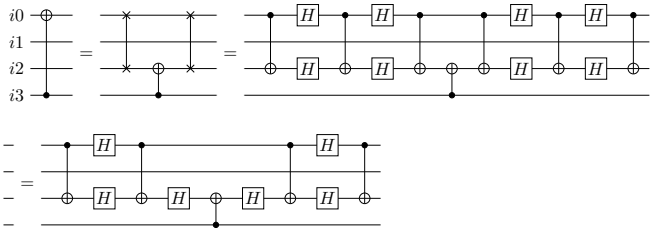
A. QX2 Architecture

In the **QX2** architecture, shown in Figure 1a, only six CNOTs (indicated by 6 arrows) out of $\binom{5}{2} = 20$ possible CNOT gates can be realized directly. The remaining 14 gates can be realized in two different ways.

Six can be realized by interchanging the target and control. This can be accomplished by adding four Hadamard gates (an example is shown below).

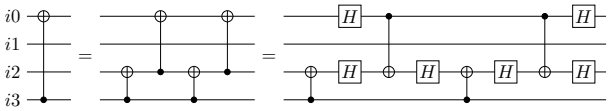


For the remaining eight CNOTs the target can be interchanged with qubit 2 and then changed back. This is illustrated with an example as shown below. A similar transformation is always possible in the other cases, too, since qubit 2 can be a target for any control.



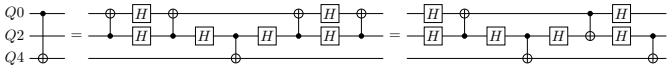
This adds six Hadamard and four CNOT gates. The number of levels increases by at most eight.

However, there is an alternative transformation—first proposed in [15]. As shown below, this will also add six Hadamard gates, but only three CNOT gates. Also, the number of levels increase by at most seven—one less than the previous method. An additional benefit may come from the fact that the two Hadamard gates at the end of the transformation may result in further reductions.

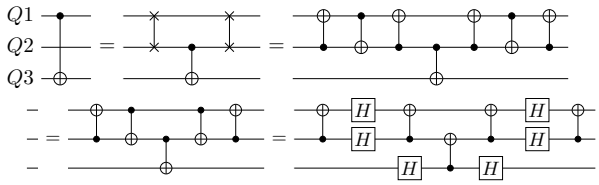


B. QX4 Architecture

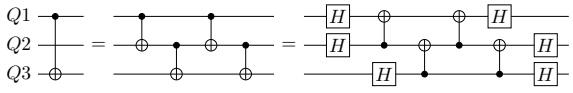
Now consider the **QX4** architecture as shown in Figure 1b. Again, six CNOTs can be realized directly and six can be realized by interchanging the target and control (as explained above.) For the following CNOTs (denoted as a pair of qubits, where the first qubit is the control and the second is the target) $\{(0,4),(1,4),(3,0),(3,1),(4,0),(4,1)\}$ we may exchange the control with qubit 2 at a cost of 10 additional gates using the swap gate approach, or with 9 additional gates using the template approach. Both methods are illustrated with an example below.



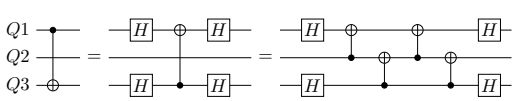
For the CNOT pairs $(0,3)$ and $(1,3)$ the target can be exchanged with qubit 2 and the direction of the CNOT is reversed. With some simplification (as shown below), this also adds 10 gates to the circuit.



A similar template based transformation as with QX2 can be applied here. Again, this leads to one fewer gate and one fewer level.



A careful analysis reveals that an even better transformation is possible (as shown below). In this case the number of Hadamard gates is further reduced by two.



Circuit identities described in this section facilitate the conversion of a given circuit into a circuit that can be realized using a particular architecture of IBM's 5 qubit quantum computers (QX2 or QX4). However, a circuit obtained in such

a way would not necessarily be optimal. Thus, to obtain better fidelity in a real experiment, we would require to develop methods for optimization of the obtained Clifford+T circuit remaining within the constrains imposed by the particular architecture. In the following section, we aim to propose such a method.

IV. Optimization

The problem addressed in this section can be stated as follows: given a quantum circuit with gates from the Clifford+T gate library, find the optimal implementation for an IBM five-qubit quantum computer. The only gates in such a circuit that are not supported directly, are some CNOT gates. It has been shown in the previous section, that there are mappings for each of those gates such that they can be realized with some added cost. The cost is measured by counting the number of gates and the number of levels. It is easy to see, that some permutation of qubits may result in differing final cost [16]. It is well known, that some gates in a quantum circuit may be interchanged. Specifically, if the matrices of two adjacent gates commute, then the gates can be interchanged. This property has been used extensively in the optimization of reversible circuits [17]. In what follows, we will use this property for the optimization of the quantum circuits designed for the implementation in IBM's 5-qubit quantum computers.

Two adjacent gates can be interchanged if one of the following conditions applies:

- They do not involve the same qubits;
- T, S, $S^\dagger$,$T^\dagger$, and Z will commute with each other even if they act on the same qubit;
- $CNOT(x, y)$ commutes with $CNOT(z, w)$ if $x \neq w$ and $y \neq z$.

The following reduction rules of adjacent gates can be applied:

- Any consecutive self inverse gates can be eliminated (such as H, X, Z, and $CNOT$);
- $TT = S$
- $SS = Z$
- $T^\dagger T^\dagger = S^\dagger$
- $S^\dagger S^\dagger = Z$
- $T^\dagger T = I$, $S^\dagger S = I$.

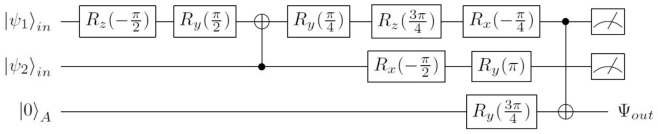
The reduction rules are applied by moving each gate as far as possible to the left in the circuit. At each position, apply a reduction rule if possible. This is only a heuristic, and some possible reductions may be missed. The main algorithm is outlined below. The function mapIBM(C_{in}) will map the necessary CNOT gates to the given IBM architecture. This function will differ according to the target architecture. Since the target architectures only have five qubits, it is feasible to consider all $5! = 120$ permutations. The function reduce() will attempt to minimized the number of gates according to the criteria given above.

Algorithm: Optimize Circuit

Data: C_{in}: circuit to be optimized (Cliffort+T gates)
Result: C_{best}: the best circuit found
C_{best} = mapIBM(C_{in});
reduce(C_{best});
foreach *line permutation* C_p *in* C_{in} **do**
  C_{tmp} = mapIBM(C_{in});
  reduce(C_{tmp});
  if *cost*(C_{tmp}) < *cost*(C_{best}) **then**
    C_{best} = C_{tmp}
  end
end

A. An Illustrative Example

Li *et al.* analyzed theoretical proposals for the implementation of approximate quantum adders [18]. These adders were optimized using a genetic algorithm. They show one example of realizing such a circuit with IBM's QX2 computer. Specifically, the following approximate quantum adder was reported in Figure 4 of [18].



This translates to a circuit with Clifford+T gates as shown in Figure 2. In [18] this is mapped to IBM QX2 quantum computer at a cost of 41 gates and 27 levels. With our proposed algorithm, the circuit shown in Figure 3 is obtained. This is accomplished by interchanging the first two qubits and applying some reductions. Interestingly, the cost is reduced to 28 gates and 18 levels. This is a significant reduction from the original proposal as it requires 32% fewer gates and 33% fewer levels. Such reduction has a big impact, given the poor fidelity of the QX2. Mapping the adder to the QX4, yields a similar result with a different qubit permutation (not shown here, due to the lack of space).

V. Experimental Results

The algorithms presented in this paper have been integrated into RevKit [19]. Runtimes for the experiments are negligible, and therefore not reported. Some experimental results are shown in Table I. For every benchmark circuit, there are two rows in the table. The first row shows the circuit without permuting the qubits. There are four ways that this can be accomplished:

1) For QX2 using the swap gate principle;
2) For QX4 using the swap gate principle;
3) For QX2 using the template principle;
4) For QX4 using the template principle.

The second row shows the best result from all permutations of the qubits. For each method the number of gates as well as the number of levels are shown in the table. The best results are shown in red.

As can be seen from Table I, significant reductions can be obtained by simply permuting the qubits. For example,

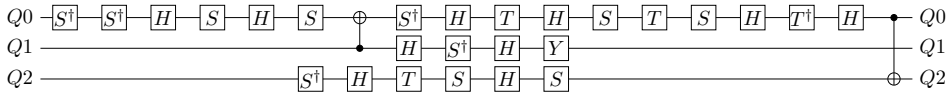


Figure 2: The circuit with gates from the Clifford+T gate library.

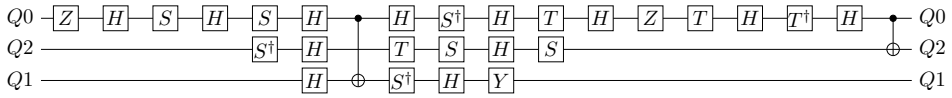


Figure 3: An adder optimized for IBM QX2.

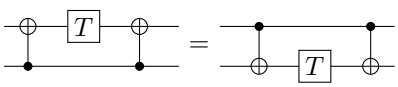
Table I: Experimental Results.

Name	lines	Swap transformation				Template transformation			
		QX2	Dep.	QX4	Dep.	GX2	Dep.	QX4	Dep.
17 [20]	4	141	92	125	72	131	91	105	69
		101	68	111	76	116	79	**87**	**56**
12 [20]	5	168	111	154	101	137	91	111	74
		104	60	98	54	85	47	**83**	**45**
a2x_c [9]	4	85	58	75	46	76	54	60	36
		59	41	59	38	55	40	**40**	**28**
a3x_c [9]	5	176	127	140	101	160	117	104	77
		86	56	56	43	86	56	**56**	**43**
7 [20]	5	184	114	182	108	174	108	160	104
		122	73	130	73	125	76	**111**	**64**
Full_Adder_c [9]	4	60	48	60	40	60	50	50	35
		28	22	32	26	27	**22**	**26**	23
Toffoli_c [9]	3	17	14	31	23	17	14	31	23
		17	**14**	**17**	**14**	**17**	**14**	**17**	**14**
4mod5-v0_18 [16]	5	209	134	211	130	192	118	184	111
		167	104	157	98	152	98	**132**	**80**
3_17_e	3	47	26	49	26	47	26	49	26
		41	**23**	**41**	**23**	**41**	**23**	**41**	**23**
01 [20]	5	149	89	157	92	141	83	135	84
		77	**38**	77	40	**77**	**38**	77	40

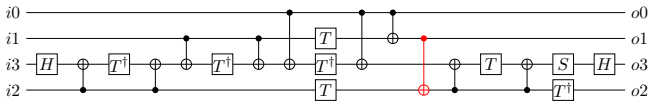
the circuit `4mod5-v0_18`, designed for QX4, was reduced from 211 (with swap transformations) to 157 gates (26%). By applying template transformation the gate count can be further reduced to 132, for a total reduction of 37%. The number of levels went from 130 to 80 (a 38% reduction). As expected, "template" transformations yield better results in most cases.

VI. Additional Optimization Ideas

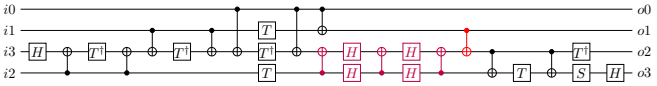
There are some circuit identities that can be particularly useful in optimizing circuits to be implemented in IBM's 5-qubit quantum computers. To illustrate this point, in what follows, we would develop some rewriting rules (circuit identities) that can facilitate gate reductions in Clifford+T circuits. The first such useful circuit identity is obtained as (where both T gates may be replaced with $T^\dagger$ gates):



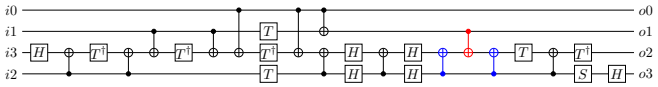
We will now illustrate the effectiveness of this circuit identity with an example. Consider the full adder implementation from [9] (with the last two lines interchanged):



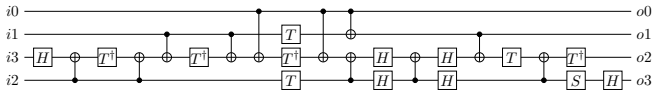
All gates, except the CNOT coloured in red are directly supported by the IBM QX2 architecture. The CNOT in question can be transformed at a cost of 10 additional gates using the "swap" transformation or 9 additional gates using the "template" transformation. However, there is a better solution as shown below. First, interchange lines 2 and 3 right after gate 13. This has the effect that the output qubits are permuted—as indicated by the output labelling. This results in the following:



Now the last two CNOTs need to be flipped. However, as explained above this can be done at no cost. This results in the following circuit:



The circuit can be further reduced by eliminating the two blue CNOT gates. Finally, we have the following circuit with an additional 4 gates—much better than the 8 in the first approach (which started with 10 additional gates, but a pair of CNOT gates could be removed.)



VII. Conclusion

Optimization of circuits is always an important step in the design of computing devices. With integrated semiconductor circuits, optimizations may have different objectives: chip area, power consumption, and speed are the most common parameters that require to be optimized. With quantum computing, an important parameter that can quantitatively describe the quality of the corresponding quantum circuit is the fidelity of the obtained output state and the desired output state. In this context, it would be apt to note that the authors of [2] have shown that a circuit with a larger number of gates has lower fidelity than an equivalent one with fewer gates. This is due to the decoherence of quantum states and the noise in the system (channel noise and lower than unity gate fidelity). Experimental results support this claim.

In this paper, we have presented tools that can help the circuit designer to find circuits with potentially fewer gates, and thus help increase the fidelity of of the results obtained in the experiments. Even if a Clifford+T circuit—of the function to be implemented—is available, it may not be possible to implement it directly on one of the IBM quantum computers. This is due to the fact, that not all CNOT gates are available. Here, we have provided a clear prescription for obtaining Clifford+T circuit which is implementable in the target IBM quantum computer.

Finally, we show some ways in which additional optimizations are possible. The next step will be to scale the tools presented here to quantum computers with 16 and 20 qubits. The translation of CNOT gates and the application of reduction rules will be pretty straightforward. However, it is not feasible to check all 16! or 20! permutations that would be required for the larger quantum computers. Consequently, smart ways of reducing the search space must be developed. We may further note that the method adapted in this work is quite general can easily be extended to improve the performance of the IBM quantum computer based experiments reported in Refs. [1], [2], [3], [4], [5] and references therein by optimizing the corresponding circuits and thus improving the fidelity. Finally, we conclude the paper with an expectation that this work will influence the performance (actually improve the fidelity) of many future experiments involving IBM quantum computers.

References

[1] M. Sisodia, A. Shukla, and A. Pathak, "Experimental realization of nondestructive discrimination of bell states using a five-qubit quantum computer," *Physics Letters A*, vol. 381, no. 46, pp. 3860–3874, 2017.

[2] M. Sisodia, A. Shukla, K. Thapliyal, and A. Pathak, "Design and experimental realization of an optimal scheme for teleportation of an n-qubit quantum state," *Quantum Information Processing*, vol. 16, no. 12, p. 292, 2017.

[3] İ. Yalçınkaya and Z. Gedik, "Optimization and experimental realization of the quantum permutation algorithm," *Physical Review A*, vol. 96, no. 6, p. 062339, 2017.

[4] B. K. Behera, A. Banerjee, and P. K. Panigrahi, "Experimental realization of quantum cheque using a five-qubit quantum computer," *Quantum Information Processing*, vol. 16, no. 12, p. 312, 2017.

[5] D. Alsina and J. I. Latorre, "Experimental test of mermin inequalities on a five-qubit quantum computer," *Physical Review A*, vol. 94, no. 1, p. 012314, 2016.

[6] N. M. Linke, D. Maslov, M. Rötteler, S. Debnath, C. Figgatt, K. A. Landsman, K. Wright, and C. R. Monroe, "Experimental comparison of two quantum computing architectures," *CoRR*, vol. abs/1702.01852, 2017. [Online]. Available: http://arxiv.org/abs/1702.01852

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

[8] E. Fredkin and T. Toffoli, "Conservative logic," *International Journal of Theoretical Physics*, vol. 21, pp. 219–253, 1982.

[9] D. M. Miller, M. Soeken, and R. Drechsler, "Mapping NCV circuits to optimized Clifford+T circuits," in *Reversible Computation - 6th International Conference, RC 2014, Kyoto, Japan, July 10-11, 2014. Proceedings*, 2014, pp. 163–175.

[10] D. M. Miller, R. Wille, and Z. Sasanian, "Elementary quantum gate realizations for multiple-control toffoli gates," in *41st IEEE International Symposium on Multiple-Valued Logic, ISMVL 2011, Tuusula, Finland, May 23-25, 2011*, 2011, pp. 288–293. [Online]. Available: https://doi.org/10.1109/ISMVL.2011.54

[11] "IBM Q," https://www.research.ibm.com/ibm-q/, accessed: 2017-09-05.

[12] P. Selinger, "Efficient Clifford+T approximation of single-qubit operators," *Quantum Information and Computation*, vol. 15, no. 1–2, pp. 159–180, 2015.

[13] "IBM QX2: Sparrow," https://github.com/QISKit/ibmqx-backend-information/blob/master/backends/ibmqx2/README.md, accessed: 2018-03-20.

[14] "IBM QX4: Raven," https://github.com/QISKit/ibmqx-backend-information/blob/master/backends/ibmqx4/README.md, accessed: 2018-03-20.

[15] M. M. Rahman and G. W. Dueck, "Synthesis of linear nearest neighbor quantum circuits," *CoRR*, vol. abs/1508.05430, 2015. [Online]. Available: http://arxiv.org/abs/1508.05430

[16] A. Zulehner, A. Paler, and R. Wille, "Efficient mapping of quantum circuits to the IBM QX architectures," *Design Automation and Test in Europe*, 2018.

[17] D. Maslov, G. W. Dueck, and D. M. Miller, "Toffoli network synthesis with templates," *Transactions on Computer Aided Design*, vol. 24, no. 6, pp. 807–817, 2005.

[18] R. Li, U. Alvarez-Rodriguez, L. Lamata, and E. Solano, "Approximate quantum adders with genetic algorithms: An IBM quantum experience," *Quantum Measurements and Quantum Metrology*, vol. 4, no. 1, pp. 1–7, 2017.

[19] M. Soeken, S. Frehse, R. Wille, and R. Drechsler, "RevKit: a toolkit for reversible circuit design," in *Proceedings of the International Symposium on Multiple-Valued Logic*, 2008, RevKit is available at http://www.revkit.org.

[20] M. Roetteler, M. Soeken, and N. Wiebe, "Reversible Logic Synthesis and Quantum Computing Benchmarks," http://quantumfpl.stationq.com/, accessed: 2017-09-19.