Combining SWAPs and Remote Toffoli Gates in the Mapping to IBM QX Architectures

Philipp Niemann[*†], Chandan Bandyopadhyay[*], and Rolf Drechsler[*†]

[*]Department of Computer Science, University of Bremen, Bremen, Germany
[†]Cyber-Physical Systems, DFKI GmbH, Bremen, Germany
{pniemann,chandan,drechsler}@uni-bremen.de

***Abstract*—Quantum computation received a steadily growing attention in recent years, especially supported by the emergence of publicly available quantum computers like the popular IBM QX series. In order to execute a reversible or quantum circuit on those devices, a mapping is required that replaces each reversible or quantum gate by an equivalent cascade of elementary, i.e. directly executable, gates—a task which tends to induce a significant mapping overhead. Several approaches have been proposed for this task which either rely on the swapping of physically adjacent qubits or the use of precomputed templates, so-called remote CNOT gates.**

In this paper, we show that combining both, swapping and remote gates, at the reversible circuit level has the prospect of significantly reducing the mapping overhead. We propose a methodology to compute the optimal combination of swaps and templates for Multiple-Controlled Toffoli gates. By using a formulation as a single-source shortest-path problem, a complete database of optimal combinations can be computed efficiently. Experimental results indicate that the mapping overhead can be significantly reduced.

I. Introduction

Quantum computation [1] received a steadily growing attention in recent years, especially supported by the emergence of first publicly available quantum computers like the popular IBM QX series [2], and substantial progress has been achieved in developing quantum computers with increasing computational power. As a consequence, the automatic generation of quantum circuits realizing a given functionality on some targeted quantum computer architecture has become an active research area.

This quantum logic synthesis is a very complex and challenging task. In order to leverage this complexity, Boolean functions—which constitute a major component in many quantum algorithms—are usually treated separately using a two-step approach: the desired Boolean function is first realized in terms of a reversible circuit, i.e., by means of classical reversible logic gates. These circuits are then transformed to a functionally equivalent quantum circuit by mapping each reversible gate to a corresponding cascade of quantum gates [3]–[5]. However, as the limitations of the targeted quantum computer architecture, particularly regarding the (non-)availability of certain gates, are usually not completely taken into account during this transformation, a so-called *technology mapping* is required afterwards in order to make the circuit executable on the targeted architecture. This leads to an overhead, since gates that are not natively available have to be rewritten as a functionally equivalent and typically much longer cascade of native gates.

In fact, there is usually only one multiple-qubit gate available in present quantum computer architectures, e.g. the two-qubit controlled-NOT or controlled-Z gates, and the connectivity is limited meaning that this gate can only be applied on a subset of qubit pairs. This also holds for the IBM quantum computers that are available to the public via free-of-charge cloud access and are also considered here.

Researchers have developed different strategies to solve such quantum architecture constraints [6]–[12], most of which rely on inserting a (minimal) number of SWAP gates that move the desired qubits close to each other. Aside from that, there are also promising proposals to not permute the qubits using SWAP gates, but to use precomputed, highly optimized sequences of native gates to realize CNOT gates as *remote* gates, i.e. without moving the involved qubits closer [13], [14].

In this paper, we are targeting the mapping of reversible circuits to the quantum level with the aim of obtaining *architecture-compliant* quantum circuits, i.e. circuits which can be executed directly on the targeted quantum computer and do not require a technology mapping to be applied. For this purpose, we generalize the idea of precomputed sequences and remote gates to the reversible circuit level, i.e. circuits composed of multiple-controlled Toffoli (MCT) gates. We propose an approach to efficiently determine optimal combinations of SWAPs and remote Toffoli gates by using a formulation as a single-source shortest-path problem. An experimental evaluation clearly demonstrates the feasibility and prospects of this approach.

The remainder of this paper is structured as follows. The next section introduces notations and preliminaries needed in this paper. Section III discusses related work, outlines the general idea and states the considered optimization problem. This is followed by Section IV which presents the details of our approach to determining optimized mappings for MCT gates. Experimental results are presented in Section V and the paper is finally concluded in Section VI.

II. Background and Preliminaries

To keep the paper self-contained, this section briefly introduces the basics of reversible and quantum circuits.

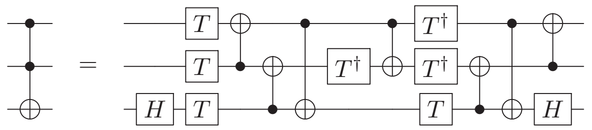

Fig. 1. Mapping of 2-controlled Toffoli gate into Clifford+T circuit [16].

A. Reversible Functions and Circuits

Reversible Boolean functions on n inputs, i.e. bijections on $\{0,1\}^n$, are realized by reversible circuits consisting of at least n lines (carrying binary values). These reversible circuits have no fan-out or feedback and typically consist of multiple-controlled Toffoli (MCT) gates which are defined as follows:

Definition 1 (Multiple-Controlled Toffoli gate)**.** *Given a set of circuit lines $X = \{x_1, x_2, \ldots, x_n\}$, a multiple-controlled Toffoli gate $T(C;t)$ is given by a set of control lines $C = \{x_{c_1}, \ldots, x_{c_m}\} \subset X$ (with $|C| \geq 0$), and a target line $t \in X \setminus C$. On the target line, the gate performs the mapping $t \mapsto t \oplus (x_{c_1} \wedge \ldots \wedge x_{c_m})$, i.e. the target line is inverted if, and only if, all control lines are in the 1-state. All other lines pass through unaltered.*

Example 1. *An MCT gate with no control line always inverts the target line and is thus the well-known* NOT gate*. An MCT gate with a single control line is called a* controlled-NOT (CNOT) gate*. When drawing reversible and quantum circuits, a $\bigoplus$ indicates the target line and a $\bullet$ indicates the control connection of an MCT gate. For instance, the left-hand side of Fig. 1 represents a 2-controlled Toffoli gate.*

B. Quantum Circuits

A quantum circuit is a model of quantum computation representing a sequence of quantum operations [1]. Each operation is represented by a quantum gate and the circuit is a cascade of quantum gates where the circuit lines represent the *qubits (quantum bits)* of a quantum system.

In this work, we consider the Clifford+T gate library [15] which is popular due to the fact that it is universal, i.e. it allows for the realization of any kind of quantum computation, even though only contains a small, discrete number of gates and due to its compatibility to schemes for error correction which are required for large-scale quantum computing. More precisely, the library contains the $CNOT$ gate, the NOT gate as well as some other single-qubit gates ($H, S, S^\dagger, T, T^\dagger$) whose precise semantics is not relevant for the purpose of this work.

Example 2. *Figure 1 shows a realization of the two-controlled MCT gate in Clifford+T [16, Fig. 7(a)]. The gates are to be applied sequentially from left to right.*

C. Decomposing MCT circuits to Clifford+T

The decomposition of reversible MCT circuits to Clifford+T quantum circuits is typically conducted in two steps. First, MCT gates with $c \geq 3$ controls are decomposed into MCT gates with less than three controls, i.e. NOT, CNOT, and 2-controlled Toffoli gates (which constitute the so-called *NCT* library). Afterwards, each NCT gate is mapped individually to an equivalent cascade of Clifford+T gates.

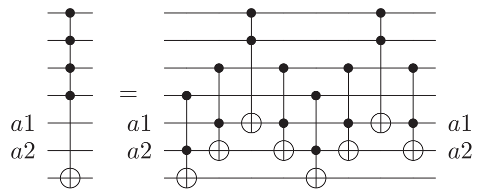

Fig. 2. Decomposition of MCT to NCT gates using ancillae [17].

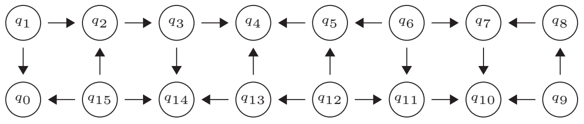

Fig. 3. IBM QX5 architecture.

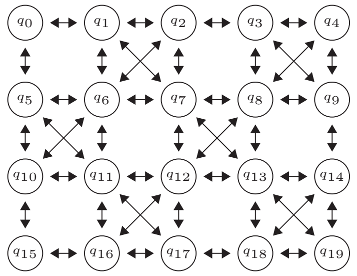

Fig. 4. IBM Q20 architecture.

Usually, the well-known decomposition of MCT into NCT gates proposed by Barenco et al. [17] is used. The basic construction requires $c-2$ additional circuit lines/qubits (so-called *ancilla* lines/qubits) that are used to store intermediate computations, but are finally restored to their initial value/state (cf. Fig. 2 for the case $c = 4$). This basic construction can also be employed as a building block in order to perform a decomposition even when less than $c-2$ ancillae are available [17, Lemma 7.3].

To map the resulting NCT circuit to Clifford+T, usually the mapping from Fig. 1 is employed for 2-controlled Toffoli gates, while NOT and CNOT gates do not have to be mapped.

D. IBM Quantum Computers

The IBM Q project [2] provides public access to quantum computers via a cloud service in order to give researchers the opportunity to *experiment* with real devices. In spite of continuous improvement, the available devices are still in development and have limitations regarding the number and the fidelity of qubits and also regarding types of quantum gates that can be used in the circuits. All architectures essentially support arbitrary single-qubit gates (especially the ones from the Clifford+T library) as well as the CNOT gate, although, its availability is limited. More precisely, the architectures restrict the interaction of the physical qubits, i.e., it is only possible to apply a CNOT gate to a defined set of qubits.

These CNOT restrictions are expressed in a *coupling graph*. Figures 3 and 4 shows the coupling graph of the 16-qubit IBM QX5 (Rueschlikon) architecture and the 20-qubit IBM Q20 (Toyko) architecture, respectively. The circles represent the physical qubits $(q_0, q_1, \ldots)$ and the arrows indicate the availability of a CNOT gate between the qubits. To this end, a CNOT gate can only be applied if the qubit at the base of the arrow is the control qubit and the tip of the arrow represents the target qubit. For instance, $T(\{q_1\}; q_0)$ is natively available in both depicted architectures, but $T(\{q_0\}; q_1)$ only in QX5.

III. Motivation and General Idea

So far, quantum circuits are often generated in a *topology-agnostic* way, i.e. without taking into account the connectivity and coupling constraints of the targeted quantum computer. As a consequence, a transformation of the quantum circuit—often termed technology mapping—is required in order to satisfy the coupling constraints and enable the circuit's actual execution. There are essentially two approaches to this transformation: using SWAPs or *remote CNOTs*.[1]

When using SWAPs, the mapping of logical to physical qubits (briefly denoted as *qubit mapping*) is permanently updated in order to make control and target of the following CNOT gate adjacent [6]–[12]. Usually the approaches perform a look-ahead in order to make sure that the updated qubit mapping is not only suitable for the next CNOT, but also beneficial for subsequent gates. In contrast, remote CNOTs are highly optimized templates, i.e. cascades of *native* gates, that realize a CNOT between distant control and target qubits without changing the qubit mapping.

Example 3. *Consider the IBM QX5 quantum computer architecture shown in Fig. 3. Assume that a CNOT gate with control on q_1 and target on q_3 is to be executed (as shown on the left-hand side of Fig. 5). Using SWAPs, q_1 is swapped with q_2 making control and target of the desired gate adjacent. Then, the desired CNOT can be executed from q_2 to q_3 and the original qubit mapping can be restored by swapping back q_1 and q_2 (as shown in the center of Fig. 5).*

Alternatively, the CNOT can be executed as a remote CNOT using the cascade of four CNOT gates shown on the right-hand side of Fig. 5. This is especially beneficial if the original qubit mapping is to be restored, since a single SWAP gate requires seven elementary gates on IBM QX5 as shown in Fig. 6.

Approaches relying only on remote CNOTs have shown very promising results [13], [14]. However, one can easily think of cases, e.g. when the same CNOT gate (with the same control and target) is to be applied multiple times in a row, where it would be better to move control and target closer to each other in order to reduce the template cost of each instance. Accordingly, some recent proposals use a combination of SWAPs and remote CNOTs [18].

The idea of this paper is to use the benefit of a variable qubit mapping together with the optimization potential of templates and lift the combination of SWAPs and remote gates

[1]Some authors prefer the terms *long-distance CNOTs* or *CNOT templates*.

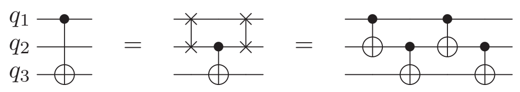

Fig. 5. Realizing a CNOT between distant qubits

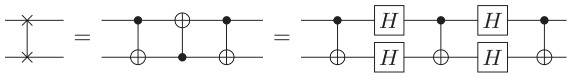

Fig. 6. SWAP gate realized in Clifford+T.

to a higher level of abstraction, namely to the level of reversible/MCT gates. By this, *architecture-compliant* quantum circuits are obtained directly from the decomposition, i.e. these do not require any further technology mapping in order to be executed at the targeted quantum computer architecture.

More precisely, each MCT gate is realized separately as a building block using pre-computed templates, while SWAPs are employed in-between the building blocks to change the qubit mapping in order to reach a configuration for the next MCT gate that has a reduced template cost.

Let us illustrate this idea by means of standard Toffoli gates, i.e. 2-controlled MCT gates. In order to compute templates for remote Toffoli gates, in a first step templates for all possible CNOT gate configurations (control and target) need to be computed in order to realize them as a remote CNOT, e.g. using the approach proposed in [13]. Recall that these templates are preserving the qubit mapping. We define the *cost* of a remote CNOT to be its overhead in terms of the number of additionally required 1- and 2-qubit gates such that the cost of a natively available CNOT is 0.

Now, given the configuration of a 2-controlled Toffoli gate, i.e. the corresponding set of controls as well as the target qubit, an architecture-compliant template realization is obtained by replacing each CNOT gate in the decomposition from Fig. 1 by the corresponding remote CNOT. The cost of this template is accordingly given by the sum of the cost of those remote CNOTs. Note that there is a priori no order of the two controls and any of the two possible orders can be used for the decomposition from Fig. 1—without any additional SWAP cost. As a consequence, the cheaper of both templates can be used for the realization of the MCT gate.

Similarly, for MCT gates with more controls, there is a large degree of freedom regarding the order of controls and choice of ancilla qubits to use for the decomposition—again without any additional SWAP cost, but with potentially significantly different realization cost.

However, even if the cheapest such template has been determined for the given configuration, it might be possible that moving around the controls or target of the MCT gate (using SWAP gates) leads to a configuration whose template realization is cheaper, even if one takes the cost of the additional SWAP gates into account.

Example 4. *Consider again the IBM QX5 quantum computer architecture shown in Fig. 3 and assume that a 2-controlled Toffoli gate with controls on q_0, q_1 and target on q_8 is to be executed, i.e. $T(\{q_0, q_1\}; q_8)$. The cost for this realization,* *i.e. realizing all CNOTs from the decomposition as remote CNOTs using the templates from [13], is* 236. *When the target is moved from* q_8 *to* q_2 *(using six SWAP gates with a total cost of* $6 \cdot 7 = 42$*), the MCT gate to be realized changes to* $T(\{q_0, q_1\}; q_2)$ *which can be realized using remote CNOTs at a cost of only* 28. *So, the swapping reduces the cost from 236 to* $28 + 42 = 70$. *However, we can get even better by moving* q_0 *to* q_{15}, q_1 *to* q_2*, and* q_8 *to* q_3 *(which requires seven SWAP gates in total with a cost of* 49*). Then, the MCT gate to be realized becomes* $T(\{q_2, q_{15}\}; q_3)$ *which only has a cost of* 18 *when realizing it using remote CNOTs, such that the total cost becomes* $18 + 49 = 67$.

As demonstrated by the example, it can sometimes be better to apply many swaps to reach a configuration with minimal template cost, but in contrast there might be a configuration with slightly higher cost that can be reached with fewer swaps such that the sum of SWAP and template cost is smaller for this combination.

Overall, these considerations give rise to the following

Optimization problem: given an MCT gate configuration $T(C; t)$, determine a qubit permutation $\pi_{\min}$ with the *minimal* sum of SWAP cost (for realizing $\pi_{\min}$) and template cost for realizing $T(\pi_{\min}(C); \pi_{\min}(t))$ as a remote Toffoli gate.

IV. Proposed Approach

In order to find the optimal combination of swaps and templates, we propose to model this as a shortest path problem. More precisely, we consider the undirected graph $G = (V, E)$ whose nodes are given by all possible Toffoli gate configurations for a fixed number of controls as well as one additional cost node v_c. All nodes are connected to v_c and these edges (termed *template edges* in the following) are weighted by the respective template cost for realizing this MCT gate as a remote Toffoli gate. Aside from that, two nodes are connected by an edge if, and only if, the corresponding configurations can be transformed into each other by applying a single SWAP gate. The weights of these edges (termed *swap edges* in the following) are given by the cost of the corresponding SWAP gate. In general, SWAPs have a constant cost that only depends on the quantum computer, but not the qubits involved.

Then, the shortest path from a node to v_c corresponds to a realization of the corresponding MCT gate that consists of SWAPs (i.e., traversing a swap edge) and applying a template (i.e., traversing a template edge). Note that we do not restore the original qubit mapping by undoing the performed SWAPs, since this would make it much more sensitive to the initial qubit mapping. However, our approach could simply be adapted to this by doubling all swap edge weights.

Example 5. *A small subgraph of the graph for* 2*-controlled Toffoli gates in the IBM QX5 architecture is shown in Fig. 7. It shows all configurations and the swapping discussed in Example 4. For instance, the swap edge between the two left-most nodes in the top row corresponds to a SWAP of* q_8 *and* q_7*. Swap edges are shown as dashed edges, while template edges to the cost node* v_c *are drawn solid. The edge weights are annotated to each edge and the shortest paths to* v_c *are indicated by directed edges in blue color.*

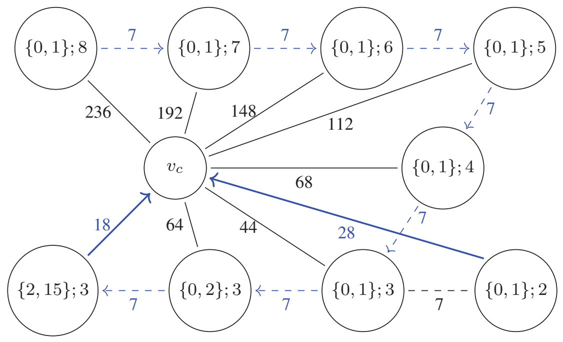

Fig. 7. Subgraph for MCT gates with $c = 2$ controls in IBM QX5

Determining the shortest paths *to* v_c for all nodes is equivalent to determining the shortest *from* v_c to all nodes. The latter constitutes a *single-source shortest path* problem which can be solved efficiently using Dijkstra's well-known algorithm:

- Initialize the *realization cost* of all nodes with the weight of their template edge and build a priority queue of all nodes sorted by their cost in ascending order.
- While the queue is not empty, take the first node v from the queue. Check all neighbors of v and update their cost, if the cost of v plus the cost of the corresponding swap edge is smaller than their current cost.

Since the cost of a SWAP is typically constant for a given quantum computer architecture and does not depend on the particular qubits being swapped, determining the weights of the swap edges is straightforward. In order to determine the weights of the template edges, i.e. the template cost for all MCT gate configurations, there is a huge design space to be explored, since there is an exponentially growing number of possibilities to choose the order of controls and ancilla qubits for each of the $m \cdot \binom{m-1}{c}$ configurations, namely $c!$ control permutations and $\binom{m-1-c}{c-2}$ choices of the ancilla qubits if the decomposition from Fig. 2 is used (where m denotes the total number of qubits in the considered architecture).

In order to cope with this, we propose to restrict to a subset of templates for each MCT gate configuration (e.g., a fixed number of random samples) and use the cheapest template among them for the weight of the corresponding template edges. Then, after computing the shortest paths for a first time, we consider only the *local minima* more exhaustively, i.e. those configurations which would be implemented directly without applying SWAPs as their shortest path is simply the corresponding template edge.

Example 6. *In the subgraph shown in Fig. 7 only the configurations* $T(\{2, 15\}; 3)$ *and* $T(\{0, 1\}; 2)$ *constitute local minima and would be explored in more detail.*

The justification/motivation for this procedure is that improvements in the template cost of a local minimum affects not only its own cost, but also all MCT gate configurations whose shortest path run through that node. In turn, it is not very promising to obtain new local minima by improving the template cost of the other nodes.

Once the template cost of all local minima are updated, we run Dijkstra's algorithm again to update the shortest paths accordingly. Overall, the steps of the algorithm are as follows:

1) Determine preliminary template costs for all configurations (by random sampling) and store them as template edge weights.
2) Perform Dijkstra's algorithm to determine shortest paths from v_c to all other nodes.
3) Restrict to the obtained *local minima* and update their template cost (by an exhaustive design space exploration or increasing the number of random samples).
4) Perform Dijkstra's algorithm again as in Step 2.

In the last step, we store the predecessor at each node, such that the shortest paths can easily be constructed from the graph. Moreover, we make use of an efficiently computable enumeration of the configurations which allows for an index-based access, e.g. $T(\{q_1, q_2, q_3\}; q_0) \mapsto 1$, $T(\{q_1, q_2, q_4\}; q_0) \mapsto 2$, and so on, such that the node for a desired configuration can later on be found efficiently.

Finally, we thus end up with an efficient data structure containing a database of optimal combinations of swaps and remote Toffoli gates for all MCT gate configurations.

While Dijkstra's algorithm scales well also for larger graphs and the effort for determining the template cost for individual configurations can be controlled very precisely, other strategies might need to be developed for the case that the entire graph becomes too big and it is no longer feasible to completely determine template cost for all nodes. In this setting, a lazy exploration of the graph might be considered, i.e. starting at the desired node and looking for a short path to the cost node. However, this is beyond the scope of this paper and is left for future work.

V. Experimental Evaluation

In this section, we evaluate the potential of the proposed approach in comparison with previous work. To this end, the proposed scheme has been implemented in C++. As target architectures, we employed IBM QX5 with 16 qubits and IBM Q20 with 20 qubits.

A. Constructing the MCT gate graph

In order to construct the MCT gate graphs, we made use of the remote CNOTs as proposed in [13] for QX5. For Q20, where all CNOTs are available in both directions, we used the generalization of the remote CNOTs shown in Fig. 5 to qubits with larger distance $d > 2$ which requires $4(d-1)$ native CNOTs as discussed in [19].

For each MCT gate configuration we generated 1,000 random samples in the first step and 100,000 random samples for local minima in the second step. The complete generation of the graphs for up to 14 controls took a bit less than 1 hour for QX5 and around 4 hours for Q20, while Dijkstra's algorithm never took more than a few seconds.

TABLE I
Single MCT gates on IBM QX5

			Cost Reduction		#SWAPS	
c	#Configs	#Minima	Average	Maximum	Average	Maximum
2	1680	114	-60.7%	-81.3%	3.38	7
3	7280	173	-56.7%	-81.0%	3.98	10
4	21840	212	-45.5%	-73.3%	4.54	12
5	48048	294	-43.6%	-68.9%	5.71	13
6	80080	439	-45.9%	-69.4%	6.86	17
7	102960	724	-45.7%	-72.5%	8.09	20
8	102960	954	-47.7%	-75.1%	9.69	25
9	80080	564	-44.9%	-70.1%	9.92	24
10	48080	236	-50.3%	-71.0%	12.20	31
11	21840	154	-36.2%	-60.2%	10.48	23
12	7280	61	-30.6%	-52.5%	8.93	18

TABLE II
Single MCT gates on IBM Q20

			Cost Reduction		#SWAPS	
c	#Configs	#Minima	Average	Maximum	Average	Maximum
2	3420	246	-69.3%	-88.0%	2.23	5
3	19380	250	-79.2%	-93.2%	2.76	5
4	77520	655	-81.5%	-96.4%	3.18	6
5	232560	1270	-83.2%	-98.1%	5.02	11
6	542640	1841	-75.9%	-93.2%	6.02	14
7	1007760	4465	-75.1%	-90.4%	8.75	20
8	1511640	7035	-65.3%	-80.0%	7.37	17
9	1847560	5521	-67.8%	-82.1%	8.87	28
10	1847560	6670	-65.7%	-79.9%	8.52	30
11	1511640	6638	-61.3%	-73.8%	9.86	29
12	1007760	4776	-56.3%	-70.1%	11.61	40

The results are summarized in Tables I and II. Here, the first two columns list the number of controls (c) and the corresponding MCT gate configurations (#Configs). The next column shows the number of local minima, i.e. MCT gate configurations that are realized directly as a remote Toffoli gate without applying SWAPs. The remaining columns show the average and maximum reduction in realization cost compared to realizing all gates as remote Toffoli gates and the average and maximum number of SWAPs performed.

As we can see, the number of local minima determined in the first round is always rather small and typically less than 1% of all configurations. The average reduction in comparison to only using remote Toffoli gates is significantly higher for Q20 (55% to 83%) as compared to QX5 (30% to 60%). This might be explained by the fact that SWAP gates are cheaper in Q20 (only 3 CNOTs as compared to 7 gates in QX5). Overall, the reduction is getting smaller for larger number of controls.

B. Mapping of Reversible Circuits

Having constructed the required MCT graphs, we evaluated the proposed approach on a suite of benchmarks taken from Revlib [20]. To this end, we processed each circuit gate by gate, looked up the optimal realization in the corresponding MCT graph and updated the qubit mapping accordingly. We observed that the results were a bit sensitive to the initial qubit mapping. Thus, we performed 10 iterations with random qubit mappings and only list the best result. The SWAP cost of the first MCT gate is not counted since these SWAPs correspond to starting with a slightly different initial qubit mapping.

TABLE III
EXPERIMENTAL RESULTS

Benchmark		Technology Mapping Overhead					
		QX5			**Q20**		
ID	#CNOTs	[18]	Prop.	Δ	[18]	Prop.	Δ
rd53_130	1043	2067	1315	-36,4%	171	67	-60,8%
f2_232	1206	2421	1377	-43,1%	213	60	-71,8%
hwb5_53	1336	2606	1689	-35,2%	174	157	-9,8%
z4_268	3073	6922	3982	-42,5%	546	241	-55,9%
radd_250	3213	7227	4187	-42,1%	729	222	-69,5%
adr4_197	3439	8084	4407	-45,5%	711	210	-70,5%
sym6_145	3888	7906	3919	-50,4%	744	225	-69,8%
misex1_241	4813	10901	6047	-44,5%	921	339	-63,2%
rd73_252	5321	12014	6620	-44,9%	1125	523	-53,5%
hwb6_56	6723	13848	8252	-40,4%	1104	555	-49,7%
cm85a_209	11414	27202	15952	-41,4%	2337	1399	-40,1%
rd84_253	13658	32626	16849	-48,4%	3198	1809	-43,4%
root_255	17159	41138	21398	-48,0%	3525	2598	-26,3%
co14_215	17936	43424	51005	17,5%	4356	8517	95,5%
mlp4_245	18852	44865	27579	-38,5%	4116	2979	-27,6%
urf2_277	20112	50031	45596	-8,9%	5934	7038	18,6%
sym9_148	21504	45903	24699	-46,2%	2172	924	-57,5%
hwb7_59	24379	51226	27612	-46,1%	4602	1885	-59,0%
clip_206	33827	81390	49407	-39,3%	6843	5869	-14,2%
dist_223	38046	92045	47492	-48,4%	6936	6062	-12,6%
sao2_257	38577	93956	65374	-30,4%	7827	8850	13,1%
urf5_280	49829	122929	98311	-20,0%	13065	12305	-5,8%
urf1_278	54766	141286	118516	-16,1%	15678	16051	2,4%
sym10_262	64283	153981	92045	-40,2%	11697	13070	11,7%

In Table III, we report the results for those benchmarks for which results are reported for both QX5 *and* Q20 in [18]. Here, the first two columns describe the benchmark in terms of its name (ID) and, as a baseline, the number of CNOT gates when applying the usual, topology-agnostic decomposition of MCT gates to Clifford+T. The next columns denote the technology mapping overhead that results when transforming the resulting quantum circuit in order to satisfy the architectural constraints (we show the results reported in [18] which can be considered the state-of-the-art in quantum circuit transformation) as well as obtained by the proposed approach, for both QX5 and Q20.[2] Run-times are not reported, since they were less than one CPU second for each benchmark—despite the 10-fold repetition. Note that in order to enable a fair comparison, no kind of further optimization has been applied to the resulting circuits as they would be performed by comprehensive quantum compilers like Qiskit, quilc, or $t|ket\rangle$.

For some benchmarks, only a marginal reduction or—in few cases—even a significantly higher overhead has been achieved by the proposed approach. However, for the vast majority of benchmarks we observed reductions of the technology mapping overhead compared to the state-of-the-art by more than 30%.

VI. CONCLUSIONS

In this paper, we considered the mapping of reversible circuits to IBM quantum computers. So far, the topological constraints of the targeted quantum hardware are hardly taken into account when decomposing reversible circuits to the quantum level such that a circuit transformation is required in order to satisfy the coupling constraints of the quantum computer. We showed that is is possible to gain architecture-compliant quantum circuits, which do not require this post-processing, directly from reversible circuits by using optimal combinations of SWAPs and remote Toffoli gates. Experimental results showed that the resulting overhead—which is key for the execution on current NISQ devices—is in most cases significantly smaller than the overhead of the best known approach for circuit transformation.

[2]Note that in [18], the total number of gates is reported, i.e. the original number of CNOTs plus the technology mapping overhead, while we list the pure overhead.

REFERENCES

[1] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000.

[2] IBM. Project IBM Q. [Online]. Available: https://www.ibm.com/quantum-computing/

[3] A. A. A. de Almeida, G. W. Dueck, and A. C. R. da Silva, "Efficient realization of toffoli and NCV circuits for IBM QX architectures," in *Reversible Computation*, 2019, pp. 131–145.

[4] R. Wille, M. Soeken, C. Otterstedt, and R. Drechsler, "Improving the mapping of reversible circuits to quantum circuits using multiple target lines," in *ASP Design Automation Conf.*, 2013, pp. 85–92.

[5] N. Abdessaied, M. Amy, M. Soeken, and R. Drechsler, "Technology mapping of reversible circuits to Clifford+T quantum circuits," in *Int'l Symp. on Multiple-Valued Logic*, 2016, pp. 150–155.

[6] J. Ding and S. Yamashita, "Exact synthesis of nearest neighbor compliant quantum circuits in 2D architecture and its application to large-scale circuits," *IEEE Trans. on CAD*, vol. 39, no. 5, pp. 1045–1058, 2020.

[7] S. Hu, D. Maslov, M. Pistoia, and J. Gambetta, "Efficient circuits for quantum search over 2D square lattice architecture," in *Design Automation Conf.*, 2019, pp. 1–2.

[8] A. Zulehner, A. Paler, and R. Wille, "An efficient methodology for mapping quantum circuits to the IBM QX architectures," *IEEE Trans. on CAD*, vol. 38, no. 7, pp. 1226–1236, July 2019.

[9] A. Matsuo, W. Hattori, and S. Yamashita, "Reducing the overhead of mapping quantum circuits to IBM Q system," in *Int'l Symp. Circ. and Systems*, Sapporo, May 2019, pp. 1–5.

[10] M. Y. Siraichi, V. F. d. Santos, S. Collange, and F. M. Q. Pereira, "Qubit allocation," in *International Symposium on Code Generation and Optimization*, ser. CGO 2018. Vienna: ACM, 2018, pp. 113–125.

[11] A. Ash-Saki, M. Alam, and S. Ghosh, "QURE: Qubit re-allocation in noisy intermediate-scale quantum computers," in *Design Automation Conf.* New York, NY, USA: ACM, 2019, pp. 141:1–141:6.

[12] R. Wille, L. Burgholzer, and A. Zulehner, "Mapping quantum circuits to IBM QX architectures using the minimal number of SWAP and H operations," in *Design Automation Conf.* Las Vegas: ACM, 2019, pp. 142:1–142:6.

[13] A. A. A. de Almeida, G. W. Dueck, and A. C. R. da Silva, "CNOT gate mappings to Clifford+T circuits in IBM architectures," in *Int'l Symp. on Multiple-Valued Logic*, 2019, pp. 7–12.

[14] A. A. A. de Almeida, G. W. Dueck, and A. C. R. da Silva, "Finding optimal qubit permutations for ibm's quantum computer architectures," in *2019 32nd Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2019, pp. 1–6.

[15] P. O. Boykin, T. Mor, M. Pulver, V. Roychowdhury, and F. Vatan, "A new universal and fault-tolerant quantum basis," *Information Processing Letters*, vol. 75, no. 3, pp. 101–107, 2000.

[16] M. Amy, D. Maslov, M. Mosca, and M. Roetteler, "A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits," *IEEE Trans. on CAD*, vol. 32, no. 6, pp. 818–830, 2013.

[17] A. Barenco, C. H. Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *Phys. Rev. A*, vol. 52, pp. 3457–3467, 1995.

[18] X. Zhou, S. Li, and Y. Feng, "Quantum circuit transformation based on simulated annealing and heuristic search," *IEEE Trans. on CAD*, vol. 39, no. 12, pp. 4683–4694, 2020.

[19] B. Nash, V. Gheorghiu, and M. Mosca, "Quantum circuit optimizations for nisq architectures," *Quantum Science and Technology*, vol. 5, no. 2, p. 025010, Mar 2020.

[20] R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler, "RevLib: an online resource for reversible functions and reversible circuits," in *Int'l Symp. on Multiple-Valued Logic*, 2008, pp. 220–225.