Improved Cost-Metric for Nearest Neighbor Mapping of Quantum Circuits to 2-Dimensional Hexagonal Architecture

Kamalika Datta[1,2], Abhoy Kole[2], Indranil Sengupta[3], and Rolf Drechsler[1,3]

[1] Institute of Computer Science, University of Bremen, Bremen, Germany
{kdatta,drechsler}@uni-bremen.de
[2] Cyber-Physical Systems, DFKI GmbH, Bremen, Germany, abhoy.kole@dfki.de
[3] Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India, isg@iitkgp.ac.in

Abstract. Quantum computing offers substantial speedup over conventional computing in solving certain computationally hard problems. The emergence of quantum computers in recent years has motivated researchers to develop design automation tools to map quantum circuits to such platforms. One major challenge is to limit the noise or computational error during gate operations; in particular, errors are higher when gates operate on non-neighbor qubits. A common approach to tackle this problem is to make the circuits *Nearest-Neighbor* (NN) compliant by inserting either *Swap gates* or *CNOT templates.* Reduction of gate overhead also becomes important as it serves to limit the overall noise and error. In some recent works, mapping of quantum circuits to hexagonal qubit architecture have been investigated. Hexagonal layout of qubits offers extended neighborhood that helps to reduce the number of Swap or additional CNOT gates required for NN-compliance. Existing approaches incur high gate overheads that can be reduced by improved gate mapping strategies with better cost metrics. The present work proposes one such approach using a priority-based cost metric. The proposed cost-metric is general and can be applied to any architectures; however, in this work we show its benefit for hexagonal architecture. Experiments on benchmark circuits confirm that the proposed method reduces gate overhead by 29% over a very recent work based on greedy mapping.

Keywords: Quantum circuits · Architecture-aware decomposition · Qubit mapping · Clean and dirty ancilla.

1 Introduction

Quantum computing has been projected to solve some computationally hard problems in appreciably less time as compared to classical computing. Some of the well-known quantum algorithms include Shor's factorization [18], Grover's database search [9], quantum simulation and annealing [14], etc. Recent developments have allowed industry giants like IBM, Google, Microsoft, etc. to come up with demonstrable quantum computers. Various technologies are used for implementation, like superconducting [8], trapped ion [10], photonic [16], etc.; however, all of them suffer from problems like limited coherence period and noisy gate operations [1].

The architecture of quantum computers can be classified as per the physical layout of the qubits, and also the way they interact among themselves. These generally include regular arrangement of qubits in a wo-dimensional (2-D) plane, on which 1- and 2-qubit gate operations are carried out. One of the major challenges is the noise generated during computation, which often restricts gate operations to neighboring or coupled qubits only. To operate on non-coupled qubits, two broad approaches are followed. In one approach, the states of neighboring qubits are exchanged using *Swap gates*, thereby bringing the states of a pair of interacting qubits to adjacent locations. As an alternative, a sequence of controlled-NOT (CNOT) gates are used to implement the gate operation on non-neighbor qubits, referred to as *CNOT templates* [17]. Both the approaches require the insertion of additional gates in the netlist, which again result in further accumulation of noise. Clearly, we need clever qubit mapping strategies to minimize the number of additional gates required to make a given quantum circuit NN-compliant.

In recent implementations, qubits are arranged in a sparse 2-D grid, with limited coupling between them. In addition, there can be directional constraints between certain pairs of qubits. Recently, the 2-D hexagonal qubit architecture has been explored [12, 19], which offers promise due to the extended qubit neighborhood that it supports. However, not much work has been done on the mapping of quantum circuits to such architectures. In a recent work [4], a greedy approach has been proposed for the mapping of qubits to hexagonal architecture based on an axial coordinate system using well-known heuristics like global and local ordering. In another work [7], an evolutionary algorithm is used to generate placement of qubits in hexagonal architecture. The gate overhead in these methods is less as compared to existing methods based on 2-D qubit architectures; however, there are further scopes for improvement.

In this paper, we propose an efficient NN-compliant quantum circuit mapping approach on the hexagonal qubit architecture. The main contributions of the paper are:

a) A new cost estimate has been formulated based on the notion of *priority* of a gate in the quantum circuit, which correlates well with the actual gate overhead for NN-compliant mapping.
b) A gate lookahead approach has been used for optimal insertion of Swap gates in the netlist for NN-compliance.

Though we focus on the hexagonal architecture in this paper, the proposed method is general and can be used for other architectures as well. A wide range of benchmarks of various sizes have been used for experimentation, which shows an average improvement of 29% in terms of gate overhead over [4].

The rest of the paper is organized as follows. Section 2 discusses the general background of the work. Section 3 presents the hexagonal qubit architecture and the coordinate system, and the proposed mapping strategy. Section 4 presents the experimental results, and finally section 5 concludes the paper.

2 Background

In this section, we present a brief introduction to quantum circuits and gates, followed by a brief discussion on emerging quantum computing architectures. Finally, the issue of logical to physical qubit mapping subject to architectural constraints is discussed.

2.1 Quantum Circuits and Gates

In quantum computing, the basic unit of information is the quantum bit or *qubit*. A qubit can exist in one of the basis states, typically denoted as $|0\rangle$ and $|1\rangle$, or in a state of *superposition* denoted as $\psi = \alpha|0\rangle + \beta|1\rangle$, where α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. Another important concept in quantum computing is *entanglement*, whereby two or more qubits can exist in entangled states and no such qubit can be measured independently of the others.

A quantum circuit consists of a set of qubits on which a set of 1- and 2-qubit gate operations are carried out in sequence, as shown in Fig. 1. Some of the quantum gate libraries that have been used by researchers include the NCV library [3], Clifford+T library [13], etc. The native gate library that is supported by any quantum computing hardware mainly depends on the technology used to implement and control the qubits. To execute a quantum circuit on a target platform, one of the important steps is to map the circuit qubits (viz., *logical qubits*) to a set of *physical qubits* as supported by the target architecture subject to architectural constraints.

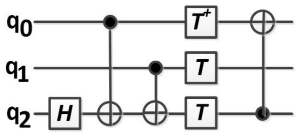

Fig. 1: An example quantum circuit.

2.2 Quantum Computing Architectures

The power and capability of a quantum computer depends on the number of qubits and the coupling constraints between them, which specifies the way the qubits are interconnected. Several initial works have been reported that assume regular arrangement of qubits on 1-, 2- or 3-D grid structures. In recent years, several practical realizations of quantum computers have been reported by IBM, Google, Microsoft, and many others [5]. For instance, the IBM-QX series of quantum computers use a 2-D grid-like structure, with degree of coupling of each qubit as 2 or 3.

To increase the degree of coupling, alternate 2-D structures like the hexagonal architecture has been explored [4, 6, 7]. Such layout with width w and height h contains $w \times h$ qubits. This allows a maximum qubit coupling of 6, as compared to 4 in standard Cartesian 2-D architecture. This increase in qubit coupling allows added flexibility in performing 2-qubit gate operations. Fig. 2 shows an example hexagonal layout of 24 (6×4) qubits.

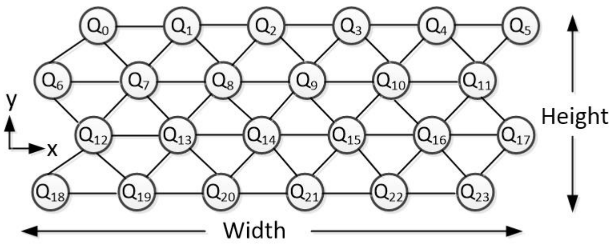

Fig. 2: A 6×4 hexagonal qubit architecture.

In this work, we consider the hexagonal qubit layout for nearest-neighbor mapping of quantum circuits to compare with the other recently introduced works that are discussed next.

2.3 Nearest-Neighbor Mapping of Qubits

When a 2-qubit gate operation is executed, a typical constraint imposed by the target architecture is that the interacting qubits must be neighbors (i.e. coupled). This is referred to as the *nearest-neighbor* (NN) constraint.

Consider a quantum circuit with five 2-qubit gates as shown in Fig. 3(a), with the target architecture shown in Fig. 3(b). In general, the logical qubits are mapped to physical qubits using global (initial) ordering or local ordering of qubits, or a combination of both. For the logical to physical mapping of qubits $(q_0, q_1, q_2, q_3) \xrightarrow{\pi} (Q_0, Q_1, Q_2, Q_3)$, some of the gate operations (viz., $G2$ and $G5$) violate NN-constraints. We typically insert Swap gates[4] as shown in Fig. 3(c) or Remote CNOT (RCNOT) [17] templates to address such violations.

In practical realizations, quantum gate operations are non-ideal, and results in accumulation of errors. It is important to minimize the number of additional gates required for NN-compliance. Since the mapping problem is NP-hard, previous works generally consider heuristics for finding solutions to this problem,

[4] A Swap gate can be realized using three back-to-back CNOT gates.

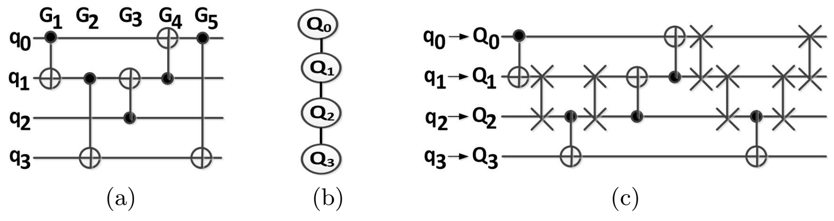

Fig. 3: (a) A quantum circuit, (b) Linear arrangement of physical qubits, (c) Swap gate insertion for NN-compliance.

e.g. [2, 11, 15, 21, 22]. The quality of the obtained solutions directly depends on the effectiveness of the adopted heuristic cost function. The main motivation of the present work is to propose a heuristic cost function that provides better NN-mapping, which leads to reduced gate overhead.

3 Proposed Mapping Method

In this work we propose an efficient scheme for mapping qubits to a hexagonal architecture, and use a priority-based cost function for NN-compliance with Swap and RCNOT gate insertion using a gate lookahead approach.

3.1 Heuristic Cost Function for NN-Mapping

The heuristic cost metric used for logical to physical qubit mapping and subsequent Swap and RCNOT insertion for NN-compliance has a direct impact on gate overhead. There are $n!$ ways (say, $\pi_0, \pi_1, \pi_2, \ldots, \pi_{n!-1}$) to map n logical qubits $\{q_0, q_1, \ldots, q_{n-1}\}$ to equal number of selected physical qubits $\{Q_0, Q_1, \ldots, Q_{n-1}\}$. A poorly selected mapping (π_i) may increase the additional gate overhead.

The cost metric used to evaluate the quality of mapping takes as input: (i) a quantum circuit in the form of a *Qubit Interaction Graph* (QIG), (ii) a *Coupling Graph* (CG) of physical qubits, and (iii) a logical to physical qubit mapping π_k. The cost metric can be expressed as:

$$cost = C \sum_{\substack{q_i, q_j \in QIG \\ \pi_k(q_i), \pi_k(q_j) \in CG}} D\left(\pi_k(q_i), \pi_k(q_j)\right) W(q_i, q_j) \tag{1}$$

where C is a constant, mapping of a logical qubit q_l to a physical qubit Q_p is denoted as $\pi_k(q_l) = Q_p$, the function $D()$ denotes the distance between nodes in CG, and $W(q_i, q_j)$ indicates the edge weight between qubits q_i and q_j in QIG. The value of the constant C gives an overhead estimate in terms of RCNOT templates or Swap gates. We have taken $C = 1$ in our experiments.

For a 2-D arrangement of physical qubits, the CG can be regular or irregular. In case of irregular layout, the distance between each physical qubit pair, $D(Q_i, Q_j)$, can be obtained using *Floyd-Warshall algorithm.* Since the algorithm is computationally expensive (i.e. $\mathcal{O}(n^3)$ for a graph with n vertices), we run it once for each irregular layout to compute a database of the distances for use in future mapping. On the other hand, for regular layout such distance $D(Q_i, Q_j)$ can be computed directly by imposing a co-ordinate system on the layout. On the hexagonal layout, we use the Cartesian co-ordinates of qubits as introduced in [6]. Considering a similar 2-D arrangement of $w \times h$ as shown in Fig. 2, the co-ordinates of a qubit Q_i can be computed as:

$$x_i = (1 - (y_i \ \mathbf{mod}\ 2)) + 2(i \ \mathbf{mod}\ w), \qquad\qquad y_i = i/w. \tag{2}$$

With this co-ordinate system, the distance between a pair of qubits (Q_i, Q_j) can be estimated as:

$$D(Q_i, Q_j) = max\left(|y_i - y_j|, \frac{MD(Q_i, Q_j)}{2}\right) - 1 \tag{3}$$

where $MD(Q_i, Q_j)$ denotes the *Manhattan Distance* between qubits Q_i and Q_j located at (x_i, y_i) and (x_j, y_j) respectively, i.e.

$$MD(Q_i, Q_j) = |x_i - x_j| + |y_i - y_j|. \tag{4}$$

The edge weight $W()$ of qubit pairs in QIG plays an important role in discriminating the mapping of qubits using the cost metric defined in Eqn. (1). We now explain how the QIG can be exploited to obtain a good qubit mapping.

3.2 Qubit Interaction Graph (QIG)

The *qubit interaction graph* (QIG) captures the degree of interaction among qubit pairs in a given quantum circuit. The vertices of QIG represent logical qubits and edges represent number of 2-qubit gates between qubit pairs.

Consider a quantum circuit with m logical qubits $\{q_0, q_1, \ldots, q_{m-1}\}$ and d gates $\{G_1, G_2, \ldots, G_d\}$. In a previous work [6], the weight of an edge (q_i, q_j) in the QIG has been defined as the number of gate operations between q_i and q_j in the circuit. In other words,

$$W(q_i, q_j) = \sum_{1 \le k \le d} GO(q_i, q_j, k) \tag{5}$$

$$\begin{aligned} \text{where } GO(q_i, q_j, k) &= 1, \ \text{if } G_k \text{ operates on } q_i \text{ and } q_j \\ &= 0, \ \text{otherwise.} \end{aligned}$$

The main drawback of this measure is that equal weightage is given to all the gates irrespective of their position in the netlist. Intuitively, less weight should be given to gates that are further away from the current position. We propose that all the 2-qubit gates are initially prioritized based on their level in the circuit netlist, such that for a given circuit of depth d, the priority p_i of the gate at level i must satisfy the following criteria:

$$p_i > p_{i+1} + p_{i+2} + \cdots + p_d \tag{6}$$

We have chosen the priority assignment $p_i = M^{d-i}$, for some real number $M \geq 2$, which satisfies Eqn.(6). For our evaluation, we have chosen $M = 2$.

In this paper, we propose a priority-based approach to estimate the edge weights in the QIG that specifically addresses this issue. We present the following alternate weight measure based on the above argument:

$$W(q_i, q_j) = \sum_{1 \leq k \leq d} 2^{d-k} \; GO(q_i, q_j, k) \tag{7}$$

$$\begin{aligned} \text{where } GO(q_i, q_j, k) &= 1, \text{ if } G_k \text{ operates on } q_i \text{ and } q_j \\ &= 0, \text{ otherwise.} \end{aligned}$$

Here, the gates that are closer to the point of reference are given higher priority in the weight calculation as compared to those that are further away. We shall refer to the two approaches of weight calculation shown in Eqn.(5) and Eqn.(7) as *Normal* and *Priority-based* respectively. Once we have the QIG with the weights defined using one of the approaches, the mapping cost is obtained using Eqn. (1). The effectiveness of the two cost metrics are analyzed in the following subsection.

3.3 Analysis of Priority-based Cost Metric

Consider the quantum circuit shown in Fig. 4(a) comprising of four 2-qubit gates, G_1, G_2, G_3, and G_4, operating at level 1, 2, 3, and 4 respectively. Fig. 4(b) shows

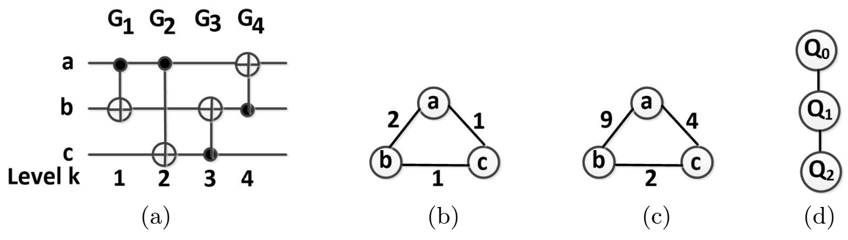

Fig. 4: (a) A quantum circuit of depth $d = 4$, (b) QIG based on Normal approach, (c) QIG based on Priority-based approach, (d) A 3-qubit physical architecture.

the QIG in which the edge weights are computed using *Normal* approach. Since the 2-qubit gates G_1 and G_4 operate on qubits a and b, the weight of the edge (a, b) is 2. Similarly, the weights of other two edges (a, c) and (b, c) are 1 due to gates G_2 and G_3 respectively.

Fig. 4(c) shows the QIG, where the edge weights are calculated using *Priority-based* approach. The calculation of the edge weights is illustrated below:

$$\begin{aligned} W(a,b) &= p_1 + p_4 = 2^{4-1} + 2^{4-4} = 8 + 1 = 9 \\ W(b,c) &= p_3 = 2^{4-3} = 2 \\ W(c,a) &= p_2 = 2^{4-2} = 4 \end{aligned}$$

This follows from the fact that between a and b there are two gates at levels 1 and 4, between b and c there is one gate at level 3, and between c and a there is one gate at level 2.

To demonstrate the benefit of the priority-based approach, we consider the mapping of the quantum circuit of Fig. 4(a) to the physical layout shown in Fig. 4(d). We consider two alternate mappings as discussed below.

a) *M1*: We use the mapping $(a, b, c) \rightarrow (Q_0, Q_1, Q_2)$. The total cost using the normal and priority-based methods will be:

$$C_1^N = 0 \times 2 + 0 \times 1 + 1 \times 1 = 1 \quad (8)$$
$$C_1^P = 0 \times 9 + 0 \times 2 + 1 \times 4 = 4 \quad (9)$$

b) *M2*: We use the mapping $(a, b, c) \rightarrow (Q_1, Q_0, Q_2)$. The total cost using the normal and priority-based methods will be:

$$C_2^N = 0 \times 2 + 1 \times 1 + 0 \times 1 = 1 \quad (10)$$
$$C_2^P = 0 \times 9 + 1 \times 2 + 0 \times 4 = 2 \quad (11)$$

The mapping *M1* requires 2 Swap operations for NN-compliance irrespective of the ways swapping is conducted as shown in Fig. 5(a) and 5(b). On the other hand, *M2* requires only 1 Swap operation for NN-compliance as shown in Fig. 5(c). This indicates that *M2* is better than *M1* in reducing gate overhead. However, the *Normal* approach is unable to discriminate this, as it gives the cost measure of 1 in both the cases as shown in Eqn. (8) and Eqn. (10). On the other hand, the *Priority-based* approach can distinguish the better mapping between *M1* and *M2* with costs of 4 and 2 respectively, as shown in Eqn. (9) and Eqn. (11).

3.4 Nearest-neighbor mapping

We consider the problem of nearest-neighbor mapping of a given quantum circuit with m logical qubits on a hexagonal qubit architecture with n physical qubits. We assume that the physical qubits are laid out on a $n = w \times h$ 2-D hexagonal structure. The mapping problem is addressed as per the following steps:

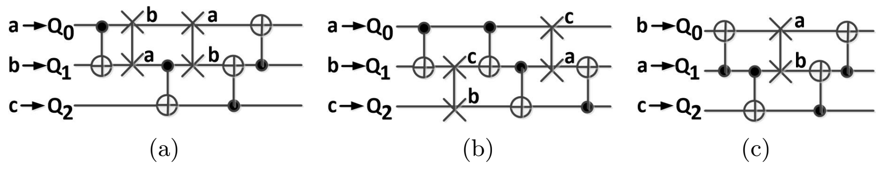

Fig. 5: (a) Swap insertion for qubit mapping *M1*, (b) Alternate Swap insertion for the same mapping, (c) Swap insertion for qubit mapping *M2*.

a) In order to map a m-qubit quantum circuit in a n-qubit layout, the physical qubits can be selected in $\binom{n}{m}$ ways. In doing this, a greedy algorithm is used, that starts with selecting a qubit located at the center of the layout i.e., $Q_c = center(w \times h)$ and the rest of $m-1$ qubits are then picked up from the layout by searching the nearmost un-mapped neighbors of Q_c.
b) There are $m!$ ways in which the m logical qubits can be mapped (π) to selected m physical qubits. We start with a set of k random mappings, $\{\pi_0, \pi_1, \ldots, \pi_{k-1}\}$, and apply an evolutionary approach (i.e. *Genetic algorithm*) over a maximum of N generations. In populating the next generation, 13% best members from the current population are directly copied, while the remaining ones are generated using crossover and mutation operations with probability values of 70% and 10% respectively. The fitness of each member in a population is measured using the cost metric defined in Eqn. (1). The best mapping π_i from the N^{th} generation is considered for further processing.
c) The mapping π_i selected in previous step may not satisfy the NN-constraint for all the 2-qubit gates present in the circuit. The circuit is traversed from left to right, and all such violations identified. For each violation, we insert Swap gates or replace with RCNOT template or use a combination of both that minimizes the NN-violations of subsequent gates in the circuit by looking ahead a further L gates from the current gate position. When Swap gates are inserted, the current mapping π_i is updated accordingly to π_i' and subsequent gates are mapped considering the updated mapping π_i'.

For the present experiment, the parameters are set as follows: population size $K = 30$, number of generations $N = 500$, and lookahead factor $L = 20$.

4 Experimental Evaluation

The proposed approach has been implemented in C++ and run on a computing system with an AMD Ryzen 7 PRO 5850U processor with Radeon Graphics running at 1.90GHz, 48GB RAM, 1TB SSD and Windows 10 Pro operating system. A comprehensive set of benchmark functions available in RevLib [20] is used for the experimentation. The benchmarks are categorized as *tiny*, *small*, *medium* and *large* [4]. In this work we have implemented two approaches for NN-compliant mapping, using RCNOT and priority-based cost function respectively.

Fig. 6 shows the gate overheads in terms of CNOT gates for method [4], RCNOT, and priority-based methods for various benchmarks. It can be seen that the proposed approaches give significant improvements as compared to [4]. Moreover, the priority-based method gives better results as compared to RCNOT-based approach for most of the benchmarks.

The results for *large* benchmarks are summarized in Table 1. In the table, the first four columns respectively denote the name of the benchmarks, number of qubits, number of 2-qubit CNOT gates in the original gate netlist, and the chosen size of the hexagonal array for mapping. The next two columns show the number of additional CNOT gates required and the run time in seconds respectively for the method in [4]. The next four columns shows the number of additional CNOT gates required and the run time in seconds respectively for the RCNOT and priority-based methods. The last three columns shows the % improvements in terms of CNOT gates. R_T is the % improvement of RCNOT method over [4], P_T is the % improvement of priority-based method over [4], and P_R is the % improvement of priority based method over RCNOT based method. Out of total 66 *large* benchmarks, 11 benchmarks gives better results using [4] and the remaining benchmarks provide better results using RCNOT method. Similarly, out of total 66 large benchmarks, 8 benchmarks gives better results using [4] and the remaining benchmarks provide better results using RCNOT method. An average improvement of 23% and 29% are observed for RCNOT and priority-based methods over [4]. This clearly shows the effectiveness of the proposed priority-based cost metric.

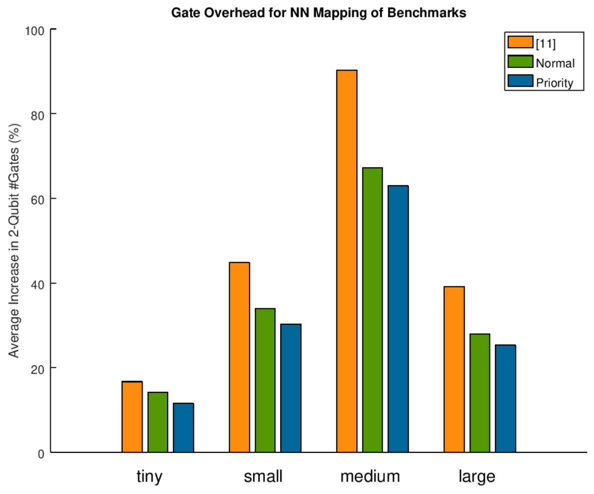

Fig. 6: Gate overhead comparison for the three methods.

Table 1: Analysis of SWAP, RCNOT and Priority cost metrics

				As per [4]		Proposed Approach						
				SWAP (T)		RCNOT (R)		Priority (P)		Improvement (%)		
Benchmark	Lines	#CNOT	Grid Size	#CNOT	Time	#CNOT	Time	#CNOT	Time	R_T	P_T	P_R
9symml_195	10	39296	4 × 3	8916	2.72	7099	1.67	6441	1.27	20.38	27.76	9.27
add6_196	19	24745	5 × 4	10218	2.20	6323	4.27	5514	3.00	38.12	46.04	12.79
alu2_199	16	44982	4 × 4	12354	3.38	9065	3.95	9513	3.40	26.62	23.00	-4.94
alu3_200	18	17787	5 × 4	5394	1.40	3823	2.96	3987	1.97	29.12	26.08	-4.29
alu4_201	22	1858472	5 × 5	439200	150.96	446213	238.70	436102	258.70	-1.60	0.71	2.27
apex4_202	28	1249592	6 × 5	467376	117.73	313290	272.64	300590	271.97	32.97	35.69	4.05
apla_203	22	26336	5 × 5	8496	2.30	6310	5.62	6377	4.81	25.73	24.94	-1.06
clip_206	14	39082	4 × 4	11388	2.85	7700	3.00	8237	2.54	32.38	27.67	-6.97
cm151a_211	28	12788	6 × 5	3336	1.24	3459	5.36	2953	5.34	-3.69	11.48	14.63
cm85a_209	14	9558	4 × 4	3522	0.78	2214	1.99	2003	0.90	37.14	43.13	9.53
cmb_214	20	49136	5 × 4	10410	3.97	11206	6.43	10584	5.56	-7.65	-1.67	5.55
co14_215	15	344008	4 × 4	75294	26.05	107222	29.65	84140	23.49	-42.40	-11.75	21.53
cu_219	25	16396	5 × 5	3972	1.27	3256	4.69	3167	3.25	18.03	20.27	2.73
cycle10_2_110	12	9122	4 × 3	1878	0.63	2386	1.47	1764	0.96	-27.05	6.07	26.07
cycle17_3_112	20	1376037	5 × 4	275616	114.71	440948	191.93	381315	216.62	-59.99	-38.35	13.52
dist_223	13	37510	4 × 4	10698	2.79	7493	2.87	8260	2.84	29.96	22.79	-10.24
dk17_224	21	13154	5 × 5	3990	1.08	3336	3.53	2995	2.53	16.39	24.94	10.22
ex1010_230	20	2165828	5 × 4	596298	174.21	496186	223.53	478578	239.60	16.79	19.74	3.55
example2_231	16	44982	4 × 4	12354	3.36	9145	4.75	8675	4.25	25.98	29.78	5.14
f51m_233	22	1759626	5 × 5	406422	139.57	507495	264.59	439626	264.25	-24.87	-8.17	13.37
hwb7_59	7	8423	3 × 3	3126	0.56	1917	0.63	1566	0.23	38.68	49.90	18.31
hwb8_113	8	29034	3 × 3	11856	2.17	7074	1.70	6012	1.23	40.33	49.29	15.01
hwb8_114	8	26169	3 × 3	9972	1.87	6046	1.18	5377	1.09	39.37	46.08	11.07
hwb8_116	8	12245	3 × 3	5310	0.99	3100	0.88	2811	0.39	41.62	47.06	9.32
hwb9_119	9	95787	3 × 3	38202	7.17	22967	3.87	20356	3.01	39.88	46.71	11.37
hwb9_121	9	95703	3 × 3	38118	7.21	23001	3.54	20123	2.97	39.66	47.21	12.51
hwb9_123	9	49777	3 × 3	19668	3.92	12025	2.21	10694	1.80	38.86	45.63	11.07
in0_235	26	1188570	6 × 5	281478	102.30	330120	253.71	295773	285.77	-17.28	-5.08	10.40
in2_236	29	1042525	6 × 5	247038	90.90	262098	257.42	270997	289.05	-6.10	-9.70	-3.40
life_238	10	19936	4 × 3	5364	1.46	3699	1.32	3093	0.77	31.04	42.34	16.38
max46_240	10	22924	4 × 3	5868	1.51	4230	1.49	4000	1.04	27.91	31.83	5.44
misex3c_244	28	5678530	6 × 5	1382388	500.65	1530202	1392.95	1490720	1474.21	-10.69	-7.84	2.58
mlp4_245	16	16940	4 × 4	5676	1.38	3677	2.91	4018	1.92	35.22	29.21	-9.27
plus127mod8192_162	13	526950	4 × 4	131688	40.66	104848	24.36	101677	23.15	20.38	22.79	3.02

continued . . .

. . . continued

				As per [4]		Proposed Approach						
				SWAP (T)		RCNOT (R)		Priority (P)		Improvement (%)		
Benchmark	Lines	#CNOT	Grid Size	#CNOT	Time	#CNOT	Time	#CNOT	Time	R_T	P_T	P_R
plus63mod4096_163	12	163663	4 × 3	43878	11.81	35368	7.77	32795	6.99	19.39	25.26	7.27
plus63mod8192_164	13	330835	4 × 4	81210	25.42	62713	14.63	65254	14.84	22.78	19.65	-4.05
rd84_253	12	9677	4 × 3	3120	0.76	1871	1.39	2043	0.82	40.03	34.52	-9.19
root_255	13	17103	4 × 4	5190	1.28	3580	1.84	3527	1.24	31.02	32.04	1.48
ryy6_256	17	257389	5 × 4	57648	19.82	70012	25.08	60008	24.62	-21.45	-4.09	14.29
sao2_257	14	76591	4 × 4	18108	5.56	15303	4.80	14954	4.17	15.49	17.42	2.28
sym10_262	11	99332	4 × 3	24108	7.09	18173	3.84	18470	3.54	24.62	23.39	-1.63
sym9_193	10	39296	4 × 3	8916	2.70	7064	1.73	6051	1.18	20.77	32.13	14.34
tial_265	22	1963642	5 × 5	462780	162.31	455134	243.02	454498	275.18	1.65	1.79	0.14
urf1_149	9	69324	3 × 3	56772	8.63	33975	5.21	28436	4.01	40.16	49.91	16.30
urf1_150	9	106039	3 × 3	40968	7.88	25482	4.00	22365	3.38	37.8	45.41	12.23
urf1_151	9	98477	3 × 3	38856	7.25	24156	3.83	21075	3.27	37.83	45.76	12.75
urf1_278	9	24916	3 × 3	22050	3.25	13035	2.36	11324	2.01	40.88	48.64	13.13
urf2_152	8	30180	3 × 3	21582	3.45	13725	2.01	11943	1.65	36.41	44.66	12.98
urf2_153	8	30013	3 × 3	12048	2.24	7167	1.38	6291	0.77	40.51	47.78	12.22
urf2_154	8	28403	3 × 3	11454	2.13	7071	1.46	5901	0.71	38.27	48.48	16.55
urf2_161	8	27460	3 × 3	18642	2.90	11409	1.83	10014	1.42	38.8	46.28	12.23
urf2_277	8	8873	3 × 3	8310	1.22	5132	1.09	4386	0.92	38.24	47.22	14.54
urf3_155	10	158808	4 × 3	134244	20.87	79233	12.28	71287	10.94	40.98	46.90	10.03
urf3_156	10	329500	4 × 3	114342	23.90	71026	11.66	64041	10.10	37.88	43.99	9.83
urf3_157	10	312746	4 × 3	109356	22.64	68380	10.75	60424	9.84	37.47	44.75	11.63
urf3_279	10	66435	4 × 3	51450	8.02	29220	5.08	26803	4.74	43.21	47.9	8.27
urf4_187	11	192024	4 × 3	131454	25.96	95514	16.48	85593	15.03	27.34	34.89	10.39
urf5_158	9	61656	3 × 3	48948	7.47	29276	4.49	25845	3.64	40.19	47.20	11.72
urf5_159	9	56096	3 × 3	18270	3.94	11604	2.15	10512	1.45	36.49	42.46	9.41
urf5_280	9	24200	3 × 3	17736	2.75	10620	2.09	9432	1.73	40.12	46.82	11.19
urf6_160	15	64440	4 × 4	67086	10.95	39401	11.48	36724	10.98	41.27	45.26	6.79
urf6_281	15	176567	4 × 4	57804	14.93	47302	14.5	44717	18.9	18.17	22.64	5.46

5 Conclusion

An improved cost metric for nearest-neighbor mapping of quantum circuits to hexagonal qubit architecture is presented in this paper. A new qubit numbering system has been used, which allows elegant computation of distance between pairs of qubits. To achieve NN-compliance, Swap gates or RCNOT templates are optimally inserted within a window of L gates using a L-gate lookahead approach. A new priority-based measure for estimating the weight of a gate in the netlist has been used, which has good correlation with the actual number of Swap gates required. Experiments on a large number of benchmark circuits show significant reduction in gate overhead over a recently published method. The method is general and can be applied to any other architectures as well.

References

1. Ahsan, M., Naqvi, S.A.Z., Anwer, H.: Quantum circuit engineering for correcting coherent noise. Phys. Rev. A **105**, 022428 (Feb 2022). https://doi.org/10.1103/PhysRevA.105.022428
2. de Almeida, A.A.A., Dueck, G.W., da Silva, A.C.R.: CNOT gate mappings to Clifford+T circuits in IBM architectures. In: Int'l Symp. on Multiple-Valued Logic. pp. 7–12 (2019)
3. Barenco, A., et al.: Elementary gates for quantum computation. Phys. Rev. A **52**(5), 3457–3467 (Nov 1995)
4. Chang, K.Y., Lee, C.Y.: Mapping nearest neighbor compliant quantum circuits onto a 2-d hexagonal architecture. IEEE Trans. on CAD of Integrated Circuits and Systems pp. 1–14 (2021)
5. Chow, J., Dial, O., Gambetta, J.: IBM Quantum breaks the 100-qubit processor barrier. https://research.ibm.com/blog/127-qubit-quantum-processor-eagle/ (2021), [Online; accessed 16-November-2021]
6. Datta, K., Kole, A., Sengupta, I., Drechsler, R.: Mapping quantum circuits to 2-dimensional quantum architectures. In: GI-Jahrestagung Workshop 2022. pp. 1109–1120 (September 2022)
7. Datta, K., Kole, A., Sengupta, I., Drechsler, R.: Nearest neighbor mapping of quantum circuits to two-dimensional hexagonal qubit architecture. In: Int'l Symp. on Multiple-Valued Logic. pp. 35–42 (May 2022)
8. Delaney, R.D., Urmey, M.D., Mittal, S., et al.: Superconducting-qubit readout via low-backaction electro-optic transduction. Nature **606**(7914), 489–493 (Jun 2022). https://doi.org/10.1038/s41586-022-04720-2
9. Grover, L.: A fast quantum mechanical algorithm for database search. In: ACM Symp. on Theory of computing. pp. 212–219 (Jul 1996)
10. Hilder, J., Pijn, D., Onishchenko, O., et al.: Fault-tolerant parity readout on a shuttling-based trapped-ion quantum computer. Phys. Rev. X **12**, 011032 (Feb 2022). https://doi.org/10.1103/PhysRevX.12.011032
11. Kole, A., Hillmich, S., Datta, K., Wille, R., Sengupta, I.: Improved mapping of quantum circuits to IBM QX architectures. IEEE Trans. on CAD of Integrated Circuits and Systems **39**(10), 2375–2383 (2020)
12. Litinski, D., Kesselring, M.S., Eisert, J., von Oppen, F.: Combining topological hardware and topological software: Color-code quantum computing with topological superconductor networks. Physical Review **7**(3), 031048 (2017)

13. Matsumoto, K., Amano, K.: Representation of quantum circuits with Clifford and $\pi/8$ gates. arXiv preprint arXiv 0806.3834 (2008)
14. Nielsen, M., Chuang, I.: Quantum Computation and Quantum Information. Cambridge Univ. Press (Oct 2000)
15. Niemann, P., Bandyopadhyay, C., Drechsler, R.: Combining SWAPs and remote Toffoli gates in the mapping to IBM QX architectures. In: Design Automation and Test in Europe. pp. 1–6 (2021)
16. Omkar, S., Lee, S.H., Teo, Y.S., Lee, S.W., Jeong, H.: All-photonic architecture for scalable quantum computing with greenberger-horne-zeilinger states. PRX Quantum **3**, 030309 (Jul 2022). https://doi.org/10.1103/PRXQuantum.3.030309
17. Rahman, M., Dueck, G.W.: Synthesis of linear nearest neighbor quantum circuits. In: 10th Int'l Workshop on Boolean Problems. pp. 1–9 (2012)
18. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: Symp. on Foundations of Computer Science. pp. 124–134 (Nov 1994)
19. Tang, H., et al.: Experimental quantum fast hitting on hexagonal graphs. Nature Photonics **12**(12), 754–758 (2018)
20. Wille, R., Große, D., Teuber, L., Dueck, G.W., Drechsler, R.: RevLib: An online resource for reversible functions and reversible circuits. In: Proc. Intl. Symposium on Multiple-Valued Logic. pp. 220–225. Texas, USA (May 2008)
21. Zhou, X., Li, S., Feng, Y.: Quantum circuit transformation based on simulated annealing and heuristic search. tcad **39**(12), 4683–4694 (2020). https://doi.org/10.1109/TCAD.2020.2969647
22. Zulehner, A., Paler, A., Wille, R.: An efficient methodology for mapping quantum circuits to the IBM QX architectures. IEEE Trans. on CAD of Integrated Circuits and Systems **38**(7), 1226–1236 (2019), http://iic.jku.at/eda/research/ibm_qx_mapping/