Optimization of Quantum Circuits for Interaction Distance in Linear Nearest Neighbor Architectures

Alireza Shafaei Mehdi Saeedi Massoud Pedram

Department of Electrical Engineering
University of Southern California
Los Angeles, CA 90089
{shafaeib,msaeedi,pedram}@usc.edu

ABSTRACT

Optimization of the interaction distance between qubits to map a quantum circuit into one-dimensional quantum architectures is addressed. The problem is formulated as the Minimum Linear Arrangement (MinLA) problem. To achieve this, an interaction graph is constructed for a given circuit, and multiple instances of the MinLA problem for selected subcircuits of the initial circuit are formulated and solved. In addition, a lookahead technique is applied to improve the cost of the proposed solution which examines different subcircuit candidates. Experiments on quantum circuits for quantum Fourier transform and reversible benchmarks show the effectiveness of the approach.

Categories and Subject Descriptors

B.6.3 [**Logic Design**]: Design Aids—*Automatic synthesis*

General Terms

Algorithms, Design

Keywords

Logic synthesis, quantum circuits, interaction distance, quantum architectures.

1. INTRODUCTION

Current technologies for quantum computing often need gates that involve geometrically *adjacent* qubits. The architecture of a quantum computing system can be described by a simple connected graph $G = (V, E)$ where vertices V represent qubits and edges E represent adjacent qubit pairs where gates can be applied on [1]. Accordingly, a complete graph expresses the absence of any *constraints*. Quantum algorithms usually consider no interaction constraint between qubits. However, physical implementation may impose additional geometrical constraints. Therefore, the developed quantum algorithms or quantum circuits should be modified to consider the effect of various technological limitations.

DAC '13, May 29 - June 07 2013, Austin, TX, USA.

Quantum computation technologies arrange qubits of a physical layout in a one (1D), two (2D), or three (3D) dimensional architecture.[1] The *Linear Nearest Neighbor* (LNN) architecture corresponds to a graph where an edge exists between only neighboring vertices in a line. *Two-dimensional square lattices* (2DSL) corresponds to a graph on a Manhattan grid with four neighboring qubits. The *three-dimensional square lattices* (3DSL) model is a set of stacked 2D lattices with six neighboring qubits. Generally, 3DSL is less restrictive. However, it can suffer from the difficulty of controlling 3D qubits. Several quantum computing systems of trapped ions [2] and liquid NMR [3] have been designed based on the interactions in a line. 2DSL proposals include arrays of trapped ions [2] and Josephson junctions [4]. The architecture in [5] is based on the 3DSL model.

Exploring an efficient realization of a given quantum algorithm or quantum circuit for a restricted architecture — the focus of this work — has been followed by different researchers during the recent years. Physical implementation of the quantum Fourier transformation (QFT) [6, 7], Shor's factorization algorithm [8–10], quantum addition [11], quantum error correction [12], and general reversible circuits [13] for the LNN/2DSL architectures have been explored in the past. Worst-case synthesis cost of a general/Boolean unitary matrix under the nearest neighbor restriction has been discussed in [14–17]. In [18, 19] heuristic methods for converting an arbitrary quantum circuit to its equivalent circuit on the LNN architectures have been proposed.

In this work, we model the problem of improving *locality*, i.e., reducing *interaction distance*, of a given quantum circuit by graph theory. Precisely, we use the *minimum linear arrangement* (MinLA) problem in graph theory to find optimized *local quantum computation*, in terms of the total synthesis cost or latency, in architectures with qubits arranged in a line. The rest of this paper is organized as follows. In Section 2, basic concepts are introduced. Prior work is discussed in Section 3. Section 4 describes the proposed approaches for locality improvement of quantum circuits. Experimental results are given in Section 5 and finally Section 6 concludes the paper. We also discuss how the proposed (MinLA)-based techniques can be generalized for 2D quantum architectures.

2. BASIC CONCEPTS

[1]Quantum technologies proposed mainly for quantum communication, such as photon-based model, is not considered here.

In the following two subsections, we briefly discuss related concepts in quantum circuits and quantum architectures.

2.1 Quantum gates and circuits

A quantum bit, *qubit*, can be considered as a mathematical object which represents a quantum state with two basic states $|0\rangle$ and $|1\rangle$. In addition to the selected basis, a qubit can get any linear combination of its basic states. A quantum system which contains n qubits is often called a *quantum register* of size n. An n-qubit *quantum gate* performs a specific $2^n \times 2^n$ unitary operation on selected n qubits. The unitary matrix implemented by several gates acting on different qubits independently can be calculated by the tensor product of their matrices. Two or more quantum gates can be cascaded to construct a *quantum circuit*. For a set of k gates $g_1, g_2, \cdots, g_k$ cascaded in a quantum circuit C in sequence, the matrix of C can be calculated as $M_k M_{k-1} \cdots M_1$ where M_i is the matrix of the i-th gate ($1 \leq i \leq k$). Given any unitary U over m qubits $|x_1 x_2 \cdots x_m\rangle$, a controlled-$U$ gate with k control qubits $|y_1 y_2 \cdots y_k\rangle$ may be defined as an $(m+k)$-qubit gate that applies U on $|x_1 x_2 \cdots x_m\rangle$ iff $|y_1 y_2 \cdots y_k\rangle$=$|11 \cdots 1\rangle$. For example, CNOT is the controlled-NOT with a single control, Toffoli is a NOT gate with two controls. A multiple-control Toffoli gate C^{k}NOT is a NOT gate with k controls. In circuit diagrams, $\bullet$ is used for conditioning on the qubit being set to value one. A SWAP gate maps $|ab\rangle$ into $|ba\rangle$. We use $\times$ on qubits of a SWAP gate in circuit diagrams. More information is in [20].

2.2 Physical layout

In a particular realistic physical layout, e.g., in an ion-trap quantum architecture [21], each qubit has a specific *physical* location at each time step. To apply a 2-qubit gate in a quantum computing system which uses *mobile* qubits, both qubits should be available at one location.[2] This is done by moving qubits from one physical location to another during the computation. Some quantum architectures provide a "MOVE" operator which moves one qubit at a time. Other ones provide a "SWAP" operator which exchanges the location of two adjacent qubits at one time step. We will discuss these cases later in the paper. If all gates use local (adjacent) qubits, physical implementation can be done with no further effort; otherwise *additional* movements are necessary.

For a given quantum circuit, an initial "trivial" qubit ordering $1, 2, \cdots, n$ is usually assumed independent of the circuit. This qubit ordering reflects the physical locations of qubits in the physical layout, i.e., qubit $\#i$ is assigned to the i-th physical location ($1 \leq i \leq n$). However, for a given quantum circuit this arrangement may not be the best in terms of the resulting *latency*.[3]

To improve the total latency, or equivalently the number of 2-qubit local gates, following [18], one can *globally* reorder qubits to change the initial physical locations as

[2]Quantum technologies with constantly moving phenomena (e.g., photons) use "flying" qubits. In this case, gates are fixed and qubits are affected upon flowing through the gates. In quantum technologies with physical locations for qubits, gates are applied in fixed locations and "mobile" qubits may travel between locations. More detail is in [22].

[3]After arranging qubits, gates should be applied. This may need to move qubits from one physical location to another. Latency is defined as the total number of time steps required to perform all gates. This time includes the time required to move qubits between gate operations, and the time required to apply all gates.

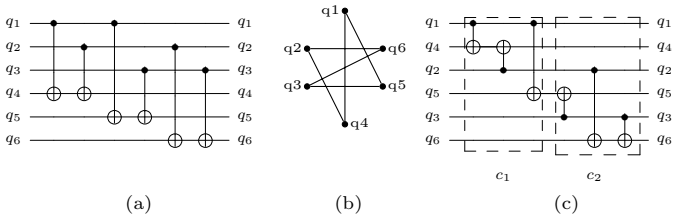

Figure 1: (a) A sample circuit, and its interaction graph (b). (c) Circuit in (a) after applying global reordering.

$l_1, l_2, \cdots, l_n$ for $1 \leq l_i \leq n$. Global reordering consumes no additional gates. To illustrate, consider a sample circuit in Figure 1(a) which is the X error syndrome using Shor-EC method for $[[9, 1, 3]]$ code [23]. The resulting circuit after applying global reordering is shown in Figure 1(c). As can be seen, the original circuit has 6 non-local gates, where four of them are local in the new circuit. Since local gates use adjacent qubits, there is no need to move the involved qubits for a gate application. This reduces the circuit latency. The problem is that even after global reordering, some gates may remain (or be) non-local. In this case, one needs to add additional local operations, MOVE or SWAP, to *move* or *permute* qubits such that the qubits that are involved in the original non-local gates will be local afterward. This is called *local reordering* [18]. Note that after a local qubit reordering, qubit locations may be changed and working with the remaining gates may need additional orderings. This is done by applying extra SWAP gates.

3. PRIOR WORK

For quantum architectures which support SWAP operation, a straightforward method to overcome the interaction constraints is to insert local SWAP gates in front of a non-local gate to permute lines (qubits) and move the involved lines toward each other. This should be followed by adding SWAP gates after the computation to recover the initial qubit ordering. For specific quantum circuits, one can explore more efficient implementations [6–9, 11, 12]. Additionally, one may try to use local gates "during" a general synthesis [13] instead of trying to reduce SWAP gates by a post-process approach. Although this seems interesting, considering locality besides other important metrics during the synthesis can complicate the overall process significantly. On another side, several researchers considered the overall impact of the interaction constraints on their developed circuits/constructions instead of working with actual circuits. In this case, they may "prove" that total cost may increase by a *constant* factor (e.g., 10 in [15] and < 2 in [14, 16]).

To work with arbitrary circuits, the authors in [18] developed exact and heuristic post-synthesis methods to reduce the number of SWAP gates. The exact method, which is limited to small circuits, was used in a peephole optimization approach. Along with 3 templates, the authors suggested two reordering strategies, global and local, where in global a qubit with the highest interaction impact is placed at the middle line continuingly until no further improvement can be achieved. In local, the algorithm inserts SWAP gates only before a non-local gate, and the new ordering is used for the remaining gates.

In [19], the authors showed that a bubble sort generates

the minimum number of SWAPs required to construct an arbitrary permutation of qubits for each gate. They additionally showed that in an n-qubit circuit, for two qubits of the i-th gate positioned at locations q_1^i and q_2^i only qubits placed between q_1^i and q_2^i should be considered instead of working with all qubits (i.e., $|q_1^i - q_2^i|!$ permutations instead of $n!$ permutations). Note that finding the *best* local orderings for all 2-qubit gates needs considering all $|q_1^i - q_2^i|!$ permutations for all gates at the same time (i.e., $|q_1^1 - q_2^1|! \times |q_1^2 - q_2^2|! \cdots$. To avoid this huge exponential search, authors worked with at most w consecutive gates.

The authors of [24] considered circuits that perform "specified" operations *spanning* n wires with focus on *depth*. They showed that *rotation* of n wires with local gates can be done in depth $n+5$, *reversing* n wires with local gates is possible with depth $2n+2$, *swapping* across n wires by local gates can be done in depth $n+7$ for even n and in depth $n+8$ for odd n with size $6n-9$. More information is in [24].

4. THE PROPOSED METHOD

Basically, a quantum computer technology, e.g., architectures based on ions, may support the MOVE operation to transform a qubit from one physical location to another. A physical location may also be shared by several qubits at the same time. In this case, to make a local two-qubit gate, one needs to change the locations of far qubits, by applying a sequence of single-qubit MOVE operations. Note that there should be enough room to hold the moving qubit(s) in intermediate and final physical locations. On the other hand, a quantum architecture may not provide the MOVE operation e.g., in architectures based on superconductors. For this case, one needs to apply SWAP gates which physically change the locations of both involved qubits. For quantum architectures with 1D interaction distance, the MOVE operation and a physical location for multiple qubits are not usual, and the previous approaches discussed in Section 3 worked with SWAP gates. The same limitations exist for some 2D quantum architectures too. For other 2D quantum architectures, the MOVE operation and a multi-qubit physical location exist. In the following sections, we propose our methods to make gates local in 1D quantum architectures. Potential problems to extend our approach for 2D quantum architectures are outlined in Section 6.

The MinLA problem is defined for a weighted graph $G = (V, E)$. The goal is to arrange the vertices V of G on an integer line by a one-to-one function $f : V \rightarrow [1 \cdots |V|]$ to minimize $\sum_{\{u,v\} \in E} w_{\{u,v\}} |f(u) - f(v)|$ where $w_{\{u,v\}}$ is the weight of the edge between nodes u and v. This problem can be considered as a *label assignment* of the given graph G. The MinLA problem is NP-hard in general. However, polynomial time algorithms to compute exact solutions for some particular graphs are known. In addition, some approximation algorithms have been proposed in the past. More information can be found in e.g., [25]. In this paper, the degree of a vertex v in graph G is represented as $\deg(v)$. The maximum degree of a graph G, denoted by $\Delta(G)$, is the maximum degree of its vertices.

4.1 Label assignment for qubit reordering

Consider a given circuit $\mathcal{C}$ with n qubits $q_1, q_2, \cdots, q_n$ and m 2-qubit gates $g_1, g_2, \cdots, g_m$.[4] Working on $\mathcal{C}$, we construct a weighted graph G, *called interaction graph*, with n vertices $v_1, v_2, \cdots, v_n$ corresponding to qubits in $\mathcal{C}$. Additionally, we add one edge with weight $w_{i,j}$ between nodes v_i and v_j if there are $w_{i,j}$ 2-qubit gates between qubits q_i and q_j in the circuit. If $w_{i,j} = 0$, we omit the edge and for $w_{i,j} = 1$, we omit the weight. Figure 1(b) illustrates the interaction graph for the circuit in Figure 1(a). For a gate g_i with qubits i and j (and $i \neq j$), interaction distance (ID) is defined as $|i-j-1|$. A gate with ID= 0 is a local gate. Total interaction distance of a circuit is a summation over the interaction distances of its gates. Accordingly, minimizing the interaction distance in $\mathcal{C}$ is equivalent to solving the MinLA problem for G. Figure 1(c) shows the circuit in Figure 1(a) after applying the MinLA algorithm where the initial ID = 12 in (a) is improved to ID = 4 in (b). While applying the MinLA problem on the whole circuit can improve the total interaction distance, some gates may remain non-local, see Figure 1(c). Next, we show how the MinLA problem can be used to make all gates local.

4.2 Subcircuits with consecutive gates

Consider a set of w consecutive 2-qubit gates $A = \{g_1, g_2, \cdots, g_w\}$ for an n-qubit LNN architecture. Assume that the gate g_i works on qubits q_1^i and q_2^i (and $q_1^i \neq q_2^i$). There are many different qubit arrangements. For an interaction graph G, we may have:

- $\Delta(G) = 0$. This is a trivial case with no gate.
- $\Delta(G) = 1$. In this case, all gates use distinct qubits. Accordingly, one can find a qubit (re)ordering when for each gate g_i, the qubits q_1^i and q_2^i are adjacent. To achieve this, group q_1^i and q_2^i as a new qubit for all gates in A for a total of $n - w$ qubit groups — each group can include either one qubit (if the qubit is not used by any gates in A) or two qubits (if exactly one gate in A uses the two qubits). There are $2 \times (n-w)!$ qubit orderings to make all gates in A local. No SWAP gate is required in this case.
- $\Delta(G) = 2$ and there is no cycle in G. For this case, there is a "staircase" construction where all gates are local. Assume there are k_0 vertices with $\deg(v) = 0$, k_1 vertices with $\deg(v) = 1$, and k_2 vertices with $\deg(v) = 2$ — equivalently k_0 qubits with no interaction, etc. For $k_0 > 0$ or $k_1 > 2$ (k_1 is even) the interaction graph is unconnected and the number of connected components is $k_0 + k_1/2$. To construct a local circuit, place all qubits (equivalently vertices in G) with $\deg(v) = 0$ close to each other, and remove such vertices from G. For a vertex with $\deg(v) = 1$, place the related qubit at the next available physical location, and remove the vertex and its edge from G. Continue the same approach for the "new" vertex (created after removing the previous vertex) with $\deg(v) = 1$. Apply this approach for all connected components in G. After all, group qubits which belong to one connected component. This leads to $k_0 + k_1/2$ groups in total. Exchanging all physical locations of one group

[4] We ignore single-qubit gates for locality improvement since they have no effect on circuit locality. Another reason is that single-qubit gates can be absorbed into surrounding two-qubit gates. The resulting circuit is called a "skeleton" circuit in [7]. Throughout the paper, we simply use circuit to mean a skeleton circuit.

with the ones for another group, in $(k_0 + k_1/2)!$ total ways, has no effect on total interaction distance. Overall, no SWAP gate is required in this case.

- In any other non-trivial cases with $\Delta(G) \geq 3$ or $\Delta(G) = 2$ and with cycle(s) in G, at least one SWAP gate is required. Assume that we divide all 2-qubit gates in $\mathcal{C}$ into k sets with only local gates, each set with at most s SWAP gates. For $s = 0$, all sets belong to either case 1 or case 2. Otherwise, one needs to add SWAPs, called **intra-set** SWAP gates. If it is not possible to make the current w gates local with s SWAP gates, decrement w to $w-1$ and recheck. If not, re-decrement and proceed. It can be verified that at least for $w = 2$, we can find a local subcircuit with no SWAP gate, i.e., $s = 0$. A larger s limit leads to a larger w.

Inter-set SWAP gates. For a circuit $\mathcal{C}$ with m gates and n qubits, assume that one finds $w_1, w_2, \cdots, w_k$ consecutive gates, $w_1 + w_2 + \cdots + w_k = m$, such that all w_i gates in set i need at most s SWAPs to be local. Assume all w_i gates in the i-th set work on $n_i \leq n$ qubits. Each set i with w_i gates needs a new qubit reordering for the involved n_i qubits. Accordingly, to have a local circuit $\mathcal{C}'$ for $\mathcal{C}$, one needs to add SWAP gates between sets i and $i+1$ for $1 \leq i \leq w-1$ to change the qubit ordering in set i to the one in set $i+1$. These SWAP gates are called **inter-set** SWAP gates. Working with an unlimited s leads to no inter-set SWAP gates. Figure 2(a) illustrates the concept. Note that the methods in [18,19] are special cases of the method discussed above in the sense that they assume m sets for a circuit with m non-local gates, no intra-set SWAPs, and then apply inter-set SWAPs to locally construct adjacent gates.

Label assignment for local reordering. To find the best possible ordering for each set, we use the MinLA problem. This is done by constructing an interaction graph G for gates in each set (vs. the whole circuit) and applying the MinLA problem for each set accordingly. The solution to the MinLA problem may not be unique, in this case the one that leads to the minimum number of inter-set SWAP gates is preferred. In case of a tie, one is selected randomly. To select sets, one can try $w = 3$ consecutive gates starting from gate i in the circuit, and then apply the MinLA problem. If a relabeling and at most s SWAP gates are sufficient to make the gates local, increment w and redo. Otherwise, use $w-1$ and restart with $i+w$. Note that the exact solution of the MinLA problem for some particular graphs can be found in a polynomial time.[5] Hence, one may be able to find optimal MinLA solutions in several sets.

Intra-set SWAP & inter-set SWAP minimization. To minimize the number of SWAP gates one should particularly consider the value of s for each set. Consider k sets indexed from 1 to k each of which with w_i gates and working on n_i qubits. Assign s_i as the maximum number of intra-set SWAP gates for set i. The value of s_i affects the final qubit ordering of set i as well as its following sets, and also the total number of sets. Figure 2 illustrates this effect with one example. Accordingly, s_i values and the total number of sets should be carefully determined. On the other hand, after distinguishing sets and appropriate qubit orderings for each set, we apply [19, Theorem 1] to find the minimum number of inter-set SWAP gates. This method is based on simulating the bubble sort algorithm.

[5]Trees, rectangular and square meshes and hypercubes are examples of graphs that can be solved in polynomial time with optimal MinLA solution [25].

4.3 SWAP minimization with lookahead

Consider an n-qubit subcircuit $\mathcal{C}$ with W gates divided into a set of k smaller subcircuits $c_1, c_2, \cdots, c_k$ each of which with w_i gates ($\Sigma_{i=1}^{i=k} w_i = W$). Applying the MinLA problem on set i leads to a qubit ordering $\mathcal{O}_i$ for this set. Given that the ordering $\mathcal{O}_i$ for set i has no effect on its successive qubit orderings for sets $i+1, \cdots, k$ (i.e., solutions of the MinLA problem in the successive sets), all k MinLA problems can be theoretically solved in parallel. To find a local subcircuit c_i' for c_i after considering the effect of $\mathcal{O}_i$, we apply the approach in [18]. Additionally, the number of inter-set SWAP gates is determined by considering qubit orderings $\mathcal{O}_i$ and $\mathcal{O}_{i+1}$ ($1 \leq i \leq k-1$) and the method in [19, Theorem 1].

Altogether, one can find the number of intra-set SWAP gates for all sets and the number of inter-set SWAPs accordingly to determine the total number of SWAP gates for the subcircuit $\mathcal{C}$. However, w_i values are not pre-determined and are subject to an optimization (Figure 2). To improve, one can move the last gate of set i to set $i+1$ (or the first gate to set $i-1$) and update the number of intra-set and inter-set SWAP gates accordingly as discussed. This should be followed by selecting the best sets to minimize the total number of SWAP gates for $\mathcal{C}$. To Achieve this, we apply a lookahead. Starting from the first gate of subcircuit $\mathcal{C}$, one needs to determine initial set borders. We begin from the gate at position $i = 0$ and include gates at positions $1 \cdots j$ until the added gate at position j leads to an interaction graph G with $\Delta(G) > 2$, or creation of a cycle in the graph. Then, we close the current set without the j-th gate, and restart the same approach from the gate at position j and a new set. Continuing this process leads to an *initial set configuration*. Then, we increment/decreamnet the number of gates at each set by $\min(L, w_i)$ for a lookahead of size L, and construct a search tree to obtain a minimized number of SWAPs for $\mathcal{C}$.

5. EXPERIMENTAL RESULTS

We implemented the proposed optimization method in C++ and all experiments were done on an Intel Core i7-3770 machine with 16GB memory. To achieve this, the program initially extracts an interaction graph from a given circuit. Then, it determines the number of sets and set borders in such a way that each set can be implemented locally with no SWAP gate. This step is followed by running several instances of the MinLA algorithm for each set. Finally, the algorithm inserts SWAP gats between adjacent sets as discussed. To reduce the number of SWAPs, for each set the ordering which leads to the minimum number of inter-set SWAP gates is selected.

To evaluate the proposed interaction distance optimiza-

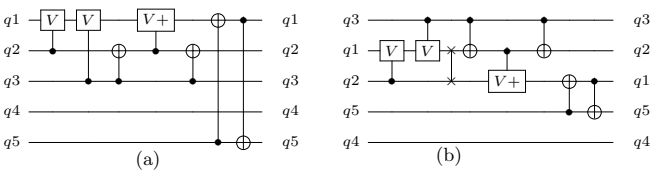

Figure 4: The result of applying the proposed method on the 4gt11_84 benchmark. (a) Non-local circuit, (b) local circuit.

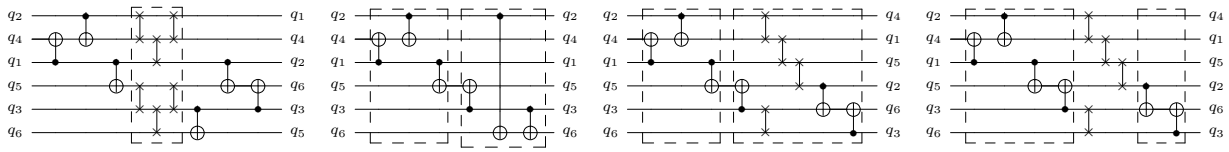

Figure 2: (a) Circuit in Figure 1(c) with only local gates. In this circuit, our algorithm was applied for subcircuits c_1 and c_2 in Figure 1(c), and 6 inter-set SWAP gates are added and $s_1 = s_2 = 0$. (b) The value of s_i in each set affects the total number of SWAP gates. Circuit in (b) has the same functionality in (a) but with $s_1 = s_2 = 4$ (redrawn in (c) with SWAP gates). The second subcircuit in (b) uses intra-set SWAP gates as shown in (c). No inter-set SWAP gate is used in (c). Note that in (c) one may consider two sets with four and two gates, shown in (d). In this case, $s_1 = s_2 = 0$, and four inter-set SWAP gates are applied. Comparing circuits in (d) and (a) further reveals the effect of selecting appropriate subcircuits on total cost.

Table 1: The synthesis results (# of SWAPs) for benchmarks in [26] as well as for quantum Fourier transform circuits after applying the method in [18] and ours. Runtime results (all in second) for [18] vary from ≈ 0 for small circuits to 1300 for large circuits. On average, the results in [18] are improved by 28%. S, m/M/A/T, and % represent # of sets, minimum/maximum/average/total numbers of SWAP gates in each set, and improvement.

			Ours							Ours			
Circuit	n	[18]	S	(m,M,A,T)	Time	%	Circuit	n	[18]	S	(m,M,A,T)	Time	%
3_17_13	3	6	5	(1,1,0.8,4)	0.007	33	hwb8_118	8	24541	6205	(1,9,2.3,14361)	389.1	41
4_49_17	4	20	10	(1,2,1.2,12)	0.006	40	hwb9_123	9	36837	7344	(1,14,2.9,21166)	1200	43
4gt10-v1_81	5	30	14	(1,3,1.4,20)	0.013	33	mod5adder_128	6	85	31	(1,4,1.6,51)	0.064	40
4gt11_84	5	3	2	(2,2,1,1)	0.01	67	mod8-10_177	5	77	43	(1,3,1.8,72)	0.02	6
4gt12-v1_89	5	35	23	(1,3,1.5,35)	0.021	0	rd32-v0_67	4	2	3	(1,1,0.6,2)	0.006	0
4gt13-v1_93	5	11	6	(1,2,1,6)	0.11	45	rd53_135	7	76	29	(1,7,2.3,66)	0.097	13
4gt4-v0_80	5	34	16	(1,5,2.2,34)	0.035	0	rd73_140	10	62	22	(1,8,2.5,56)	12.472	10
4gt5_75	5	17	9	(1,2,1.3,12)	0.015	29	sym9_148	10	5480	1736	(1,8,1.9,3415)	833.33	38
4mod5-v1_23	5	16	8	(1,2,1.2,9)	0.015	44	sys6-v0_144	10	62	21	(1,7,2.8,59)	9.814	5
4mod7-v0_95	5	28	15	(1,2,1.4,21)	0.012	25	urf1_149	9	60235	19952	(1,11,2.2,,44072)	896.1	27
aj-e11_165	4	39	23	(1,4,1.5,36)	0.018	8	urf2_152	8	25502	8652	(1,9,2.0,17670)	61.14	31
alu-v4_36	5	23	10	(1,5,1.8,18)	0.017	22	urf5_158	9	52440	17705	(1,9,2.2,39309)	1191.7	25
decod24-v3_46	4	4	3	(1,2,1,3)	0.01	25	QFT5	5	12	3	(3,3,2,6)	0.008	50
ham7_104	7	84	32	(1,7,2.1,68)	0.082	19	QFT6	6	22	4	(2,7,3,12)	0.053	45
hwb4_52	4	14	8	(1,2,1.2,10)	0.005	29	QFT7	7	39	5	(3,10,5.2,26)	0.253	33
hwb5_55	5	79	42	(1,5,1.5,63)	0.037	20	QFT8	8	60	5	(5,13,6.6,33)	1.77	45
hwb6_58	6	136	54	(1,6,2.1,118)	0.049	13	QFT9	9	87	6	(4,16,9,54)	22.332	38
hwb7_62	7	3660	961	(1,8,2.2,2128)	11.99	42	QFT10	10	123	7	(2,18,10,70)	4.261	43

tion method, we compared our results with those obtained by applying the method in [18] for reversible benchmarks in [26] as well as for the quantum Fourier transform circuits. For all cases, the number of SWAP gates added by each method to construct a local circuit was compared. In [18], quantum costs before and after the optimization were reported where SWAP gate was considered as a unit-cost gate (see [18, Table 3]). Accordingly, the number of SWAP gates is the difference of quantum cost before and after the optimization. We limited the runtime to 30 minutes and reported those cases that our algorithm leads to a solution. Table 1 reports the results.

For each circuit in Table 1, besides the number of SWAP gates we reported the number of sets and the minimum, maximum and average numbers of SWAP gates in each set. Note that we limited the algorithm to use $s = 0$ in each set. Accordingly, no intra-set SWAP gate is used and all SWAPs are the result of applying inter-set SWAP insertion method. We also limited the algorithm to use a lookahead of $depth = 1$ to reduce runtime. Increasing the lookahead depth improves the results with the penalty of runtime. As can be seen in Table 1, the average number of SWAP gates required to transform a local ordering from one set to another set is small, and our algorithm leads to a considerable reduction in the number of SWAP gates — 28%, on average and up to 60%. Figure 3 and Figure 4 illustrate the results of applying the proposed method on two benchmarks. In these circuits, all SWAP gates are inserted between sets.

6. CONCLUSION AND FUTURE WORK

In this paper, the interaction distance constraint in quantum architectures with 1D interactions was addressed. We modeled the interactions between gate qubits in a given circuit by an interaction graph, and used the well-known Minimum Linear Arrangement (MinLA) problem to find qubit reordering to improve circuit locality. The proposed approach divides a given circuit into several subcircuits and uses the MinLA instances within each subcircuit to find qubit locations for qubits involved in each subcircuit. Next, local SWAP gates are inserted inside each subcircuits to make the remaining non-local gates local. Finally, additional SWAP gates are inserted between subcircuits to transform one qubit ordering to another to keep circuit functionally unchanged. The proposed approach applies a lookahead to determine subcircuit borders and to minimize the total number of SWAP gates. Given that the MinLA is NP-hard, the problem of minimizing the number of necessary SWAP gates to run an arbitrary quantum circuits on LNN architectures is NP-hard. This addresses the **conjecture** in [19, page 25].

In addition to considering circuit depth and improving runtime to handle larger circuits, a major step in future direction is to consider the interaction distance constraint in 2D quantum architectures. This path has been followed for specific circuits in the past e.g., [10]. However, the case of general circuits needs attention.

- For architectures which do not support MOVE and use one physical location per qubit, we can extend MinLA to the 2-dimensional grid arrangement problem [27]. New methods are required to insert intra-set SWAPs and to determine initial qubit locations.

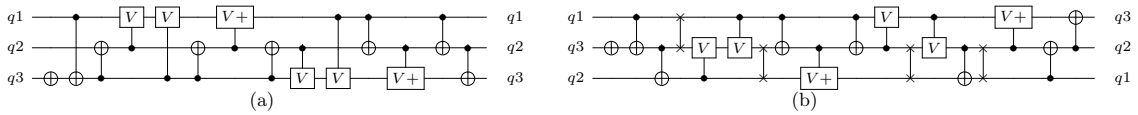

Figure 3: The result of applying the proposed method on the 3_17_13 benchmark. (a) Non-local circuit, (b) local circuit.

- For quantum architectures which support MOVE and may hold several qubits in one physical location our ideas should be revised. In this case, one can construct a graph for physical locations (vs. qubits) and use one node for qubits with the same location. This can be followed by a 2-dimensional grid arrangement problem to determine a qubit ordering. Besides the challenges stated above, the algorithm should handle the maximum size of intermediate and final physical locations when one qubit is passing from one physical location. Method in [28] is a related approach.

7. REFERENCES

[1] D. Cheung, D. Maslov, and S. Severini. Translation techniques between quantum circuit architectures. *Workshop on Quant. Inf. Proc.*, Dec 2007.

[2] H. Häffner et al. Scalable multiparticle entanglement of trapped ions. *Nature*, 438:643–646, Dec 2005.

[3] M. Laforest et al. Using error correction to determine the noise model. *Phys. Rev. A*, 75(1):133–137, 2007.

[4] B. Douçot, L. B. Ioffe, and J. Vidal. Discrete non-Abelian gauge theories in Josephson-junction arrays and quantum computation. *Phys. Rev. B*, 69(21):214501, Jun 2004.

[5] C. A. Pérez-Delgado, M. Mosca, P. Cappellaro, D. G. Cory. Single spin measurement using cellular automata techniques. *Phys. Rev. Lett.*, 97(10):100501, 2006.

[6] Y. Takahashi, N. Kunihiro, and K. Ohta. The quantum Fourier transform on a linear nearest neighbor architecture. *Quant. Inf. Comput.*, 7:383–391, 2007.

[7] D. Maslov. Linear depth stabilizer and quantum Fourier transformation circuits with no auxiliary qubits in finite neighbor quantum architectures. *Phys. Rev. A*, 76, 2007.

[8] A. G. Fowler, S. J. Devitt, and L. Hollenberg. Implementation of Shor's algorithm on a linear nearest neighbour qubit array. *Quant. Inf. Comput.*, 4:237–245, 2004.

[9] S. A. Kutin. Shor's algorithm on a nearest-neighbor machine. *Asian Conf. on Quant. Inf. Sci.*, 2007.

[10] P. Pham, K. Svore. A 2D nearest-neighbor quantum architecture for factoring. *arXiv:1207.6655*, 2012.

[11] B.-S. Choi and R. Van Meter. On the effect of quantum interaction distance on quantum addition circuits. *J. Emerg. Technol. Comput. Sys.*, 7(3):11:1–11:17, August 2011.

[12] A. G. Fowler, C. D. Hill, L. Hollenberg. Quantum error correction on linear nearest neighbor qubit arrays. *Phys. Rev. A*, 69:042314.1–042314.4, 2004.

[13] M. Arabzadeh, M. Saheb Zamani, M. Sedighi, and M. Saeedi. Depth-optimized reversible circuit synthesis. *Quant. Inf. Proc.*, 2012.

[14] M. Möttönen, J. J. Vartiainen. Decompositions of general quantum gates. *Ch. 7 in Trends in Quantum Computing Research, NOVA Publishers*, 2006.

[15] V. V. Shende, S. S. Bullock, and I. L. Markov. Synthesis of quantum-logic circuits. *IEEE Trans. CAD*, 25(6):1000–1010, June 2006.

[16] M. Saeedi, M. Arabzadeh, M. Saheb Zamani, and M. Sedighi. Block-based quantum-logic synthesis. *Quant. Inf. Comput.*, 11(3-4):0262–0277, 2011.

[17] M. Saeedi, M. Saheb Zamani, M. Sedighi, and Z. Sasanian. Reversible circuit synthesis using a cycle-based approach. *J. Emerg. Technol. Comput. Sys.*, 6(4):13:1–13:26, 2010.

[18] M. Saeedi, R. Wille, and R. Drechsler. Synthesis of quantum circuits for linear nearest neighbor architectures. *Quant. Inf. Proc.*, 10(3):355–377, 2011.

[19] Y. Hirata, M. Nakanishi, S. Yamashita, and Y. Nakashima. An efficient conversion of quantum circuits to a linear nearest neighbor architecture. *Quant. Inf. Comput.*, 11(1–2):0142–0166, 2011.

[20] M. Saeedi and I. L. Markov. Synthesis and optimization of reversible circuits - a survey. *ACM Computing Surveys, to appear, arXiv:1110.2574*, 2012.

[21] D. Kielpinski, C. Monroe, and D. J. Wineland. Architecture for a large-scale ion-trap quantum computers. *Nature*, 417:709–711, Jun 2002.

[22] R. Van Meter and M. Oskin. Architectural implications of quantum computing technologies. *J. Emerg. Technol. Comput. Sys.*, 2(1):31–63, 2006.

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

[24] S. Kutin, D. Moulton, and L. Smithline. Computation at a distance. *Chicago J. of Theor. Comput. Sci.*, 2007.

[25] J. Petit. Experiments on the minimum linear arrangement problem. *J. Exp. Algorithmics*, 8, 2003.

[26] R. Wille et al. RevLib: An online resource for reversible functions and reversible circuits. *Int'l Symp. on Multiple-Valued Logic*, pages 220–225, May 2008.

[27] M. Oswald, G. Reinelt, and S. Wiesberg. Exact solution of the 2-dimensional grid arrangement problem. *Discrete Optimization*, 9(3):189–199, 2012.

[28] D. Maslov, S. M. Falconer, M. Mosca. Quantum circuit placement. *IEEE Trans. CAD*, 27(4):752–763, 2008.

Acknowledgments

This research was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20165. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.