Problem Specific Communications in Distributed Quantum Computing

Joyanta Basak
School of Computing
University of Connecticut
Storrs, USA
joyanta.basak@uconn.edu

Bing Wang
School of Computing
University of Connecticut
Storrs, USA
bing@uconn.edu

Sanguthevar Rajasekaran
School of Computing
University of Connecticut
Storrs, USA
sanguthevar.rajasekaran@uconn.edu

***Abstract*—When all the qubits needed for solving a problem are not located in a single quantum computer, qubits from different quantum computers can be collectively utilized. In this case, quantum communication is needed for the multiple quantum computers to communicate with each other. Several studies address the problem of minimizing the number of quantum communications when evaluating a general quantum circuit. The solutions proposed typically involve solving some intractable problems. In this paper, we show that we can obtain much better solutions when we focus on solving specific problems (instead of seeking solutions for generic circuits). Specifically, we consider several fundamental quantum circuits and identify communication protocols that need a much smaller number of communication steps than those offered by generic solutions. Our work is in line with traditional parallel and distributed computing research where typically scientists focus on solving specific problems (such as sorting, matrix multiplication, network flow, etc.) in a parallel or distributed setting.**

***Index Terms*—distributed quantum computing, parallel algorithms, quantum fourier transform, quantum network.**

I. Introduction

Quantum algorithms have been proposed to solve a myriad of problems including search, integer factorization, combinatorial optimization, option pricing, etc. Two of the most well-known algorithms in quantum computing are the prime factorization algorithm of Shor [12] and the search algorithm of Grover [7]. Shor's algorithm was the first one to show that prime factorization and discrete logarithms problems can be solved in polynomial time. Given an unsorted sequence X of length N, Grover showed how to perform search in this sequence in $O(\sqrt{N})$ time using a quantum algorithm. Any traditional algorithm will need $\Omega(N)$ time to solve this problem in the worst case.

Despite rapid progress, current quantum processors can only support a limited number of qubits (up to a few thousand). However, many problems of interest need significantly more qubits. One way of dealing with this shortcoming is to employ distributed quantum computing [3] over multiple quantum processors. Communications among these quantum processors are through quantum networks, which consume quantum resources (e.g., entanglement) and adds noise to computation. Therefore, one goal of distributed quantum computing is to reduce the amount of communication needed.

Researchers have proposed many solutions to address this problem (see e.g., [1], [2], [4], [5], [9], [10], [13]–[15]). In all existing studies, however, the proposed solutions are generic, i.e., they are designed to work for any problem. This is in contrast to traditional parallel computing, where algorithms are typically proposed for specific problems, not broadly for a class of problems. For example, numerous parallel sorting algorithms are known, and many parallel matrix multiplication algorithms have been published.

In this paper we demonstrate that creating quantum distributed algorithms for specific problems is much more desirable than generic solutions. In particular, we show that we can minimize the communication cost more effectively when we focus on individual problems. We exemplify this approach using three important problems: quantum Fourier transform (QFT) that is used in Shor's algorithm, Grover's search algorithm, and Satisfiability problem. Our evaluation shows that our approach requires significantly less teleportations than a random local search algorithm for all three problems. We believe our work points to an important direction for further improving the efficiency of distributed quantum computing.

II. Background

Quantum circuit. Quantum algorithms can be represented as quantum circuits. We can think of a quantum circuit as a leveled graph. If there are n qubits, we can represent them as n horizontal lines/wires. Each vertical level corresponds to a set of gates. We can correlate each level with a time step. The qubits go through several levels of gates. At the end, we can measure the circuit. If the circuit computes a function f on n qubits, the measurement will give us the value of $f(i)$ for some i, $0 \leq i \leq (2^n - 1)$. The specific value of i will depend on the amplitudes associated with the different states of the qubits.

Distributed quantum computing. A large-scale quantum circuit that cannot run on a single quantum machine can be split to run on multiple quantum machines that are connected by a quantum network.

Two subproblems in distributed quantum computing are: (i) *qubit allocation*, which allocates each qubit of the quantum circuit to a certain quantum processor in a given set of quantum processors, so that the number of qubits allocated to quantum processor r does not exceed its storage capacity n_r, and (ii) *non-local gate operation*, which determines how to support gate operations that involve qubits on two different quantum processors. One mechanism to realize non-local gate operation is through teleportation, which is used in this paper due to its many advantages over cat-entanglement [13]. Specifically, suppose a gate operation involves two quantum processors, r_1 and r_2. Then r_1 can teleport its qubit $|x\rangle$ to r_2; r_2 in turn completes the gate operation locally, and then teleports $|x'\rangle$ back to r_1. The goal of the above two subproblems is to reduce the number of teleportations needed.

In this paper, we solve both of the above two subproblems. We assume that each quantum processor has a single communication qubit (which differs from computation qubits), and hence can only perform one teleportation at a time. In addition, each quantum processor can store a single qubit (separate from its own computation qubits) that is teleported from other processors.

III. Quantum Fourier Transform (QFT)

QFT is a very fundamental problem not only in traditional computing but also in quantum computing. For example, Shor's

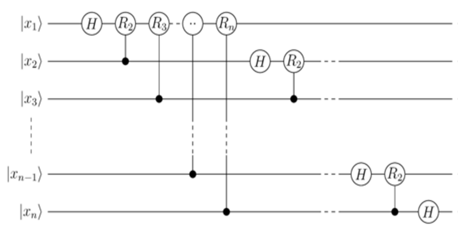

Fig. 1: Quantum circuit for Fourier transforms.

algorithm for factorization employs QFT as one of the components [12]. Figure 1 shows the circuit for QFT, when the size of the input is $N = 2^n$. We employ n qubits. In this figure, R_s stands for the unitary operator:

$$R_s = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^s} \end{bmatrix},$$

for $s = 1, 2, \ldots$. Let the number of processors available be p. Also, let the number of qubits available in processor i be n_i, for $1 \leq i \leq p$ such that $\sum_{i=1}^{p} n_i = n$. Let $m_j = \sum_{i=1}^{j} n_i$, for $j = 1, 2, \ldots, p$, with $m_0 = 0$. Without loss of generality, assume that processor i has the qubits $x_{m_{i-1}+1}, x_{m_{i-1}+2}, \ldots, x_{m_i}$, for $1 \leq i \leq p$. For this circuit, we propose the following communication protocol.

Handling gates involving $|x_1\rangle$: Consider $|x_1\rangle$. In the first n time steps, all the gates applied involve $|x_1\rangle$. Specifically, the gates applied in the first n time steps involve: $|x_1\rangle$, $(|x_1\rangle, |x_2\rangle)$, $(|x_1\rangle, |x_3\rangle), \ldots, (|x_1\rangle, |x_n\rangle)$. In the first n_1 time steps, there is no need for teleporting $|x_1\rangle$ since all the qubits participating in the gates are with processor 1. At time step $n_1 + 1$, we teleport $|x_1\rangle$ to processor 2. Assume that a teleportation and gate application can be done in the same time step. (If not, we can expand the time scale by a factor of 2). From time step $n_1 + 1$ through $n_1 + n_2$, the qubits participating in the gates are with processor 2. At time step $n_1 + n_2 + 1$, we teleport $|x_1\rangle$ to processor 3, and so on. In summary, we perform $p - 1$ teleportations in the first n time steps.

Handling gates involving $|x_2\rangle$: In time steps $n + 1$ to $2n - 1$, all the gates in the circuit involve $|x_2\rangle$. Specifically, the gates applied in these $n - 1$ time steps involve $|x_2\rangle$, $(|x_2\rangle, |x_3\rangle)$, $(|x_2\rangle, |x_4\rangle)$, ... , $(|x_2\rangle, |x_n\rangle)$. From time step $n + 1$ through $n + n_1 - 1$, the qubits participating in the gates are with processor 1. At time step $n + n_1$, we teleport $|x_2\rangle$ to processor 2. From time step $n + n_1$ through $n_1 + n_2 - 1$, the qubits participating in the gates are with processor 2. At time step $n + n_1 + n_2$, we teleport $|x_1\rangle$ to processor 3, and so on. In summary, we perform $p - 1$ teleportations in times steps $n + 1$ to $2n - 1$.

We can handle the other gates in a similar manner. To sum up, for the qubits $|x_1\rangle$ through $|x_{n_1}\rangle$, we do $p - 1$ teleportations. For qubits $|x_{n_1+1}\rangle$ through $|x_{n_1+n_2}\rangle$, we perform $p - 2$ teleportations. As a result, the total number of teleportations done is $n_1(p-1) + n_2(p-2) + \cdots + n_{p-1}$. If QFT is the only operation we are interested in, then there is no need for any other teleportations. If there are other operations to be performed on the qubits, then they will have to be teleported back to their original processors. This will take a total of $n - n_p$ teleportations. Therefore, the total number of teleportations taken in the general case is $n_1(p-1) + n_2(p-2) + \cdots + n_{p-1} + (n - n_p)$.

Consider the special case of $n_i = n/p$, for $1 \leq i \leq p$. In this case, the total number of teleportations is $\frac{n}{p}[(p-1) + (p-2) + \cdots + 1 + (p-1)] = \frac{n(p-1)}{2} + (n - n/p)$.

We can transform the circuit shown in Figure 1 to an equivalent circuit by replacing the R binary gates with CZ binary gates and adding up to $\frac{n}{2}$ swap operations at the end. The i-th swap operation swaps the quantum state of qubit q_i with the quantum state of qubit q_{n-i}. Each R gate can be implemented using two CZ gates and the swap operation is implemented using three CZ gates.

Consider a QFT setting with $n = 64$ qubits and $p = 4$ partitions. For this setting, our mentioned protocol takes 144 teleportations. On the contrary, the generic local search algorithm (Section VII-A) takes 336 teleportations, more than $2\times$ of what our approach needs.

IV. Grover's Algorithm

Given an unsorted sequence $A[1:N]$ and another element x, Grover's algorithm searches for x in $A[1:N]$ in $O(\sqrt{N})$ time using a quantum circuit. Clearly, any traditional algorithm will need $\Omega(N)$ time to solve this problem. We can also think of searching as the following problem: the input is a sequence $X = k_1, k_2, \ldots, k_n$ and a function $f : X \rightarrow \{0,1\}$, and the problem is to find an i such that $f(k_i) = 1$.

In this circuit, there are two main blocks that are repeated $O(\sqrt{N})$ times each. On any given input $|x\rangle$, the first block (labeled $O_f^{\pm}$) returns $(-1)^{f(x)}|x\rangle$ as the output. This block can be constructed from an oracle gate for computing $|f(x)\rangle$ on any $|x\rangle$. We assume that there exists such an oracle gate. Thus we only focus on the second block (labeled D) that is known as the diffusion block. This block modifies the amplitudes. The diffusion gate (see e.g., [11]) is $H^{\bigotimes n} Z_0 H^{\bigotimes n} = H^{\bigotimes n}(2|0^n\rangle\langle 0^n| - I)H^{\bigotimes n} = 2|+^n\rangle\langle +^n| - I$, where

$$Z_0|x\rangle = \begin{cases} |x\rangle \text{ if } |x\rangle = |0^n\rangle \\ -|x\rangle \text{ if } |x\rangle \neq |0^n\rangle. \end{cases}$$

One way of constructing Z_0 is shown in Figure 2. As we see from this picture, Z_0 can be realized if we have a circuit for computing the OR of n qubits. In the study of quantum communications, only binary gates are considered (see e.g., [13]). As a result, in the context of Grover's algorithm and quantum communication complexity, we conclude that the circuit of interest is one where we perform the OR of n bits. One such circuit is given in Figure 3. For simplicity, assume that when an OR gate is applied between $|x_i\rangle$ and $|x_{i+1}\rangle$, OR of these two qubits is available in $|x_{i+1}\rangle$. Here again, assume that the number of processors available is p. Also, let the number of qubits available in processor i be n_i, for $1 \leq i \leq p$ such that $\sum_{i=1}^{p} n_i = n$. Again, the qubits are allocated sequentially to the p processors, with n_i qubits for processor i.

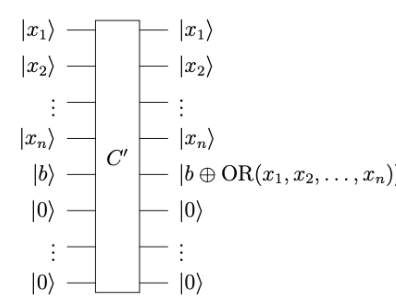

Fig. 2: Realizing Z_0 in the Diffusion gate.

In the first $n_1 - 1$ time steps, there is no need for any teleportations. At the end of step n_1, the OR of the first n_1 qubits is available in $|x_{n_1}\rangle$. In time step n_1, $|x_{n_1}\rangle$ is teleported to processor 2. For the next $n_2 - 1$ time steps, there is no need for any teleportation. In time step $n_1 + n_2$, $|x_{n_1+n_2}\rangle$ is teleported to processor 3, and so on. In total, the number of teleportations needed is $p - 1$.

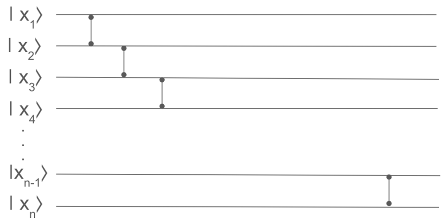

Fig. 3: Boolean OR of n qubits.

In a setting with $n = 64$ qubits and $p = 4$ partitions, our mentioned protocol takes 6 teleportations. On the contrary, the generic local search algorithm (Section VII-A) takes 76 teleportations.

V. Satisfiability

The satisfiability problem (SAT) has numerous applications and is known to be $\mathcal{NP}$-complete. Many other important problems can be reduced to SAT, and hence this problem has been studied extensively. Given a Boolean formula F on n variables, the SAT problem is to check if F has a satisfying assignment. A simple algorithm can be used to solve SAT in $O(2^n)$ time (see e.g., [8].

It has been shown that we can use Grover's algorithm to create a quantum algorithm for solving SAT whose runtime is $O(\sqrt{2^n})$. The idea is to replace the oracle in the Grover's algorithm with a boolean circuit for F. The diffusion module will remain the same. Call this quantum algorithm as QSAT.

In this section, we study the quantum communication complexity of QSAT. Assume that there is a neuron corresponding to each variable in the formula F. As earlier, we assume that the neurons are distributed across p machines.

Let $F = C_1 \wedge C_2 \wedge C_3 \wedge \cdots \wedge C_q$, where $C_1, C_2, \ldots, C_q$ are clauses (i.e., disjunctions of literals). The oracle circuit will have a component for each of the clauses and these segments will appear in sequence. For any C_i, the segment will be a boolean circuit for realizing a disjunction of the literals in C_i.

Note that we have to construct a specific quantum circuit for each input formula F. We won't be able to use the same quantum circuit to solve SAT on two different formulas (even if they have the same number of variables). Keeping this in mind, we look at some specific boolean formulas and compute the quantum communication complexity for each.

Example 1: In this example, there are 16 clauses on 16 variables. The clauses are: (x_1, x_2, x_5), (x_1, x_2, x_3, x_6), (x_2,x_3,x_4,x_7), (x_3,x_4,x_8), (x_1,x_5,x_6,x_9), (x_2,x_5,x_6,x_7,x_{10}), $(x_3, x_6, x_7, x_8, x_{11})$, (x_4, x_7, x_8, x_{12}), $(x_5, x_9, x_{10}, x_{13})$, $(x_6,x_9,x_{10},x_{11},x_{14})$, $(x_7,x_{10},x_{11},x_{12},x_{15})$, $(x_8,x_{11},x_{12},x_{16})$, (x_9, x_{13}, x_{14}), $(x_{10}, x_{13}, x_{14}, x_{15})$, $(x_{11}, x_{14}, x_{15}, x_{16})$, and (x_{12},x_{15},x_{16}).

Assume that we have 4 machines, each with 6 qubits. The solution we suggest is to assign qubits 1, 2, 5, and 6 to machines 1; qubits 3, 4, 7, and 8 to machine 2; qubits 9, 10, 13, and 14 to machine 3; and qubits 11, 12, 15, and 16 to machine 4.

In our example, consider C_1. All the qubits are with machine 1 and hence there is no need for any teleportation. Consider C_2. Only qubit 3 is needed from machine 2. This qubit will be moved to machine 1 and then moved to machine 2. Hence this is counted as two teleportations. Continuing in a similar manner, we realize that 32 teleportations are utilized. In contrast, the local search algorithm introduced in Section VII-A required 48 teleportations, 50% more than our solution.

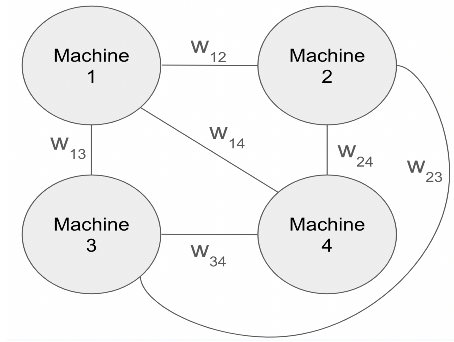

Fig. 4: An example of a fully connected weighted network with 4 machines, where w_{ij} represents the teleportation cost between machines i and j.

Algorithm 1: GreedyDeterministicMapping()

Input: Network as weighted graph, $G = (V, E)$ Communication requirement as weighted graph $G' = (P', E')$
Output: A map, $M : P' \to V$.
1: M: an empty map of size $|P'|$
2: L: list of edges in E sorted in ascending order of edge weights.
3: L': list of edges in E' sorted in descending order of edge weights.
4: **for** $i = 1$ to $|V|/2$ **do**
5: $(u,v) = L[0]$
6: $(u',v') = L'[0]$
7: $M[u'] = u$
8: $M[v'] = v$
9: Remove all edges containing u or v from L
10: Remove all edges containing u' or v' from L'
11: **end for**
12: **return** M

VI. Qubit Assignment in Weighted Network

Our circuit partitioning schemes introduced in Sections III and IV can partition the qubits into non-uniform sizes of partitions. They assume the underlying communication cost between any pair of machines is uniform. We now consider a heterogeneous setting where the quantum communication cost between a pair of machines is non-uniform. One example is shown in Figure 4. Non-uniform communication costs can happen due to various reasons in practice (e.g., different distances, and different number of quantum repeaters connecting two quantum computers). While non-uniform communication cost has also been considered in [13], [15], as we shall see below, our focus differs from them since we treat it as the second step, after assigning qubits to a set of quantum computers (following Sections III and IV).

Optimal circuit partitioning assuming non-uniform communication cost among machines is known to be intractable [2]. Hence, we propose a greedy algorithm to solve this problem as shown in Algorithm 1.

Our proposed greedy mapping algorithm looks at a pair of partitions and assigns them to a pair of machines (lines 7 & 8 in Algorithm 1). Given a pair of partitions (p_i, p_j) and a pair of machines (m_u, m_v), we can either map partition p_i to machine m_u and p_j to machine m_v, or we can map p_i to m_v and p_j to m_u. In our experiments on the weighted networks in Section VII-D, we explored two strategies: (i) following the mapping presented in Algorithm 1, and (ii) randomly picking one of the

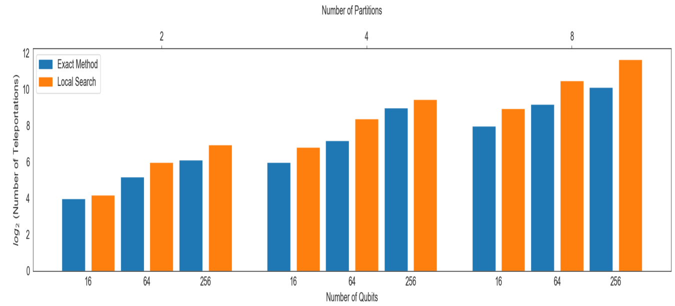

Fig. 5: Total teleportation cost (in $\log_2$ scale) incurred by our proposed exact method vs local search method for partitioning QFT circuit with 16, 64, 256 qubits into 2, 4, and 8 equal partitions.

two possible assignments. Our results show that for large circuits, the latter tends to outperform the former.

VII. Experimental Results

We conducted a series of experiments to partition the QFT circuit, diffusion module circuit of Grover's search algorithm, and QSAT circuits. In our experiments, we varied the number of qubits $n \in \{16,64,256\}$. For each number of qubits, we varied the number of partitions $p \in \{2,4,8\}$. We assumed the capacity of each partition is $\frac{n}{p}$. Hence, each machine would require only two more qubits in addition to the partition size – one for storing the teleported qubit and one for facilitating the communication.

A. Local Search Algorithm

Previously, local search methods were used to evaluate quantum circuit partitioning methods [4]. Similarly, we propose a local search algorithm to compare against our proposed problem-specific partitioning schemes.

In our proposed local search algorithm, we follow a simple strategy to generate candidate solutions which are partitions of the qubits. Initially, we randomly assign qubits to the partitions till a partition is full to the capacity. We produce a candidate solution by randomly picking two partitions and exchanging one randomly picked qubit from each partition. We measure the fitness of the candidate solution by counting the number of qubit teleportation required by the candidate solution. A candidate solution with lesser teleportation has a higher fitness. We select the higher fitness candidate solution as the current solution with high probability. With a small probability, we would select the existing candidate solution. Finally, after a certain number of iterations, we output the best-found solution.

In our experiments, we ran the local search algorithm for 10,000 iterations. In each iteration, the candidate solution is selected with the probability $\alpha = 0.2$. If the candidate solution is not selected on the first attempt, but the candidate solution is more fit than the current solution, it will be selected with a probability of $\beta = 0.8$.

In our experiments on QSAT, we calculate the minimum number of required qubit teleportation for each clause. The total required qubit teleportation is the sum of the required qubit teleportation for all clauses.

B. Experiments on QFT and Grover's Circuits

We proposed exact partitioning methods in Section III for QFT circuits and Grover's search algorithm in Section IV.

The results for the experiments on the QFT circuit are presented in Figure 5 and the results for the experiments on Grover's search algorithm are shown in Figure 6.

Our experiments show that the exact method can partition the qubits such that it incurs significantly fewer qubit teleportations.

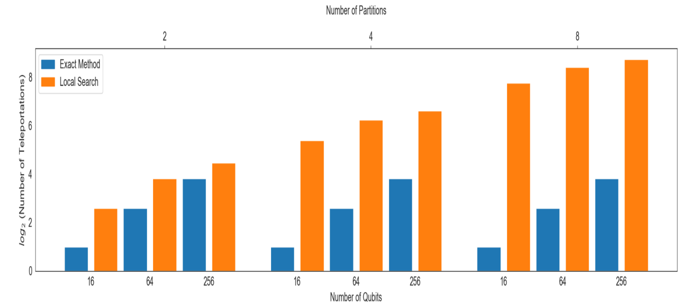

Fig. 6: Total teleportation cost (in $\log_2$ scale) incurred by our proposed exact method vs local search method for partitioning diffusion gate circuit in Grover's search algorithm with 16, 64, 256 qubits into 2, 4, and 8 equal partitions.

This is because our protocol exploits the symmetry of qubit gate operations in the circuit. Whereas, the local search algorithm can only search the solution space without any explicit knowledge of the symmetry. Although the local search algorithm runs for a large number of iterations (10,000), it fails to find a similarly good solution to the exact method. Furthermore, the solution space increases exponentially when increasing the number of qubits or partitions. With a fixed number of iterations, the local search algorithm performs comparatively worse when the number of qubits or the partitions is higher.

C. Experiments on QSAT

We conducted two series of experiments on QSAT circuits to partition them into 4 partitions. We assumed that each literal is mapped to a qubit. These qubits are arranged in a mesh topology. In the first experiment, we assumed each clause is constituted by literals whose mapped qubits are on a path in the mesh. We generated 1000 clauses by randomly varying the length of each clause between 3 to 5 literals. We divided the mesh into four quartets as done in the example shown in Section V. We ran the simulation 5 times. The average performance of each of the methods is shown in Figure 7.

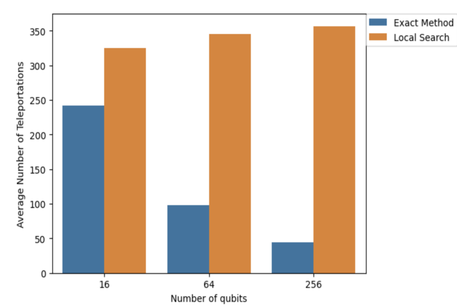

Fig. 7: Average teleportation cost incurred by a fixed partitioning method vs local search method for satisfying 1000 randomly generated clauses over the mesh topology.

In the second series of experiments, clauses constituted randomly picked literals instead of following the inter-connectedness of the mesh. The comparative performance of the fixed partitioning scheme and the local search algorithm is shown in Figure 8.

Our results in Figure 7 show that the fixed partitioning scheme requires fewer teleportations when clauses are sampled from a mesh topology. In fixed partitioning, each partition contains all the qubits in a quarter of the mesh. The clauses are also constituted by literals that are mapped to neighboring qubits. Hence, some clauses have only literals that are mapped to qubits residing in the same partition, incurring no teleportation cost. However, when we generated the clauses by sampling literals randomly, irrespective of the inter-connectedness of their mapped qubits, we see in Figure 8 that

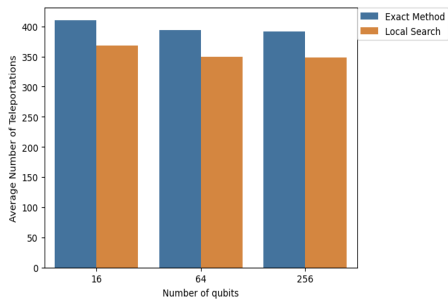

Fig. 8: Average teleportation cost incurred by a fixed partitioning method vs local search method for satisfying 1000 randomly generated clauses.

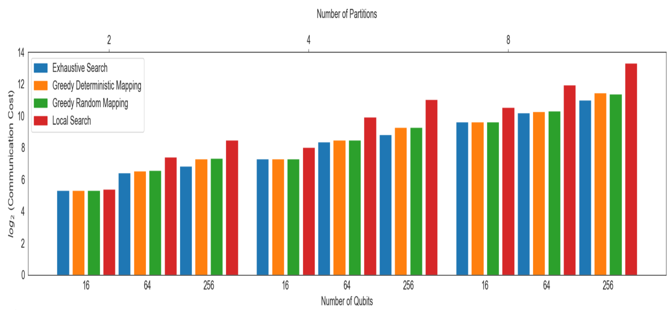

Fig. 9: Average Communication cost (in $\log_2$ scale) incurred by various partition mapping methods for 30 weighted networks and partitioning QFT circuit with 16, 64, 256 qubits into 2, 4, and 8 equal partitions.

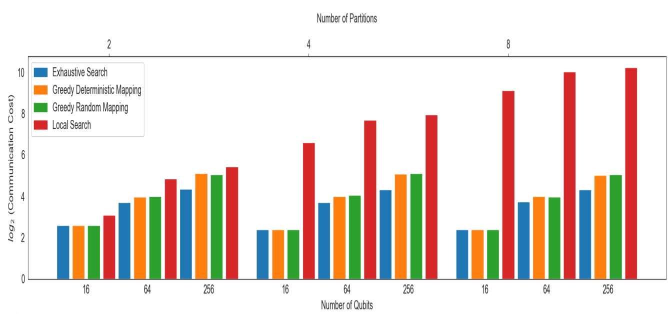

Fig. 10: Average Communication cost (in $\log_2$ scale) incurred by various partition mapping methods for 30 weighted networks and partitioning diffusion gate circuit in Grover's search algorithm with 16, 64, 256 qubits into 2, 4, and 8 equal partitions.

the local search algorithm fares better than our scheme. It shows that a partitioning scheme tailored for the SAT problem instance can result in less communication as opposed to a generic scheme.

D. Experiments with Non-uniform Communication Cost

We experimented with weighted networks that represent heterogeneous quantum networks as discussed in Section VI. For each experiment, we generated a fully connected undirected weighted network with nodes representing the machines and weights of the edges (in range $[1,5]$) representing the communication costs between machines. We repeated each experiment 30 times and reported the average communication cost. We compared four methods for each network configuration. First, we partitioned the QFT or Grover's circuit with our proposed partitioning schemes. As the first method (termed *Exhausting Search*) we conducted a exhaustive search to find optimal partition to machine (node) mapping. The second method (termed *Greedy Deterministic Mapping*) followed Algorithm 1. The third method (termed *Greedy Random Mapping*) picked one of the two possibilities in partition assignment randomly (line 7 & 8 in Algorithm 1). Lastly, we employed our local search algorithm. Results are shown in Figure 9 and Figure 10.

In our experiments with both the QFT circuits and Grover's diffusion gate circuits, we found both of our greedy mapping algorithms find near optimal mapping. These costs are much lower than the local search method, particularly for cases with a high number of qubits and a high number of partitions. Between the two of our proposed greedy algorithms, none performs better than the other consistently. However, for circuits with a large number of qubits (e.g., 256 qubits), the Greedy Random Mapping performs marginally better than the Greedy Deterministic Mapping.

VIII. Conclusions

In this paper, we have addressed an important problem in distributed quantum computing. The problem is minimizing the complexity of communication. Our main goal is to emphasize that, in line with traditional parallel computing, we can reduce the communication complexity more effectively if we create algorithms specific to individual problems. We have demonstrated our thesis with three different problems: QFT, Searching, and SAT. We could not get the source code for any of the algorithms published in the literature. As a result, we have created our own generic randomized algorithm. The performance of this algorithm has been compared with our solutions that are designed specific to the problems. This comparison convincingly proves our thesis.

References

[1] P. Andres-Martinez, T. Forrer, D. Mills, J.-Y. Wu, L. Henaut, K. Yamamoto, M. Murao, and R. Duncan, Distributing circuits over heterogeneous, modular quantum computing network architectures, *Quantum Science and Technology*, October 2024, Vol. 9. No. 4.

[2] P. Andres-Martinez and C. Heunen, Automated distribution of quantum circuits via hypergraph partitioning, *Phys. Rev. A*, September 2019, 100:032308, doi: 10.1103/PhysRevA.100.032308.

[3] M. Caleffi, M. Amoretti, D. Ferrari, D. Cuomo, J. Illiano, A. Manzalini, and A.S. Cacciapuoti, Distributed quantum computing: a survey, arXiv preprint arXiv:2212.10609, 2022.

[4] O. Daei, K. Navi, and M. Zomorodi-Moghadam, Optimized quantum circuit partitioning, *Int J Theor Phys*, December 2020, 59(12):3804-3820, ISSN 1572-9575. doi: 10.1007/s10773-020-04633-8.

[5] Z. Davarzani, M. Zomorodi-Moghadam, M. Houshmand, and M. Nouribaygi, A dynamic programming approach for distributing quantum circuits by bipartite graphs, *Quantum Information Processing*, 19, 2020. doi: 10.1007/s11128-020-02871- 7.

[6] D. Fernandes, C. Silva, and I. Dutra, Using Grover's search quantum algorithm to solve boolean satisfiability problems, part 2, XRDS, 26(2):68–71, November 2019.

[7] L.K. Grover, A fast quantum mechanical algorithm for database search, Proc. of STOC, 1996, Philadelphia, PA, pp. 212–219.

[8] E. Horowitz, S. Sahni, and S. Rajasekaran, *Computer Algorithms*, Silicon Press, 2008.

[9] M.Z. Moghadam, M. Houshmand, and M. Houshmand, Optimizing teleportation cost in distributed quantum circuits, *International Journal of Theoretical Physics*, 2018, 57(3):848–861.

[10] E. Nikahd, N. Mohammadzadeh, M. Sedighi, and M.S. Zamani, Automated window-based partitioning of quantum circuits, *Phys. Scr.*, January 2021, 96(3):035102, ISSN 1402-4896. doi: 10.1088/1402-4896/abd57c.

[11] R. O'Donnel and J. Wright, Quantum Computation and Information, Lecture Notes, 2015, CMU, https://www.cs.cmu.edu/~odonnell/quantum15/.

[12] P.W. Shor, Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, *SIAM Journal on Computing* 26 (5): 1484–1509, October 1997.

[13] R.G. Sundaram and H. Gupta, Distributing Quantum Circuits Using Teleportations, arXiv:2306.00195v1, May 31, 2023.

[14] R.G. Sundaram, H. Gupta, and C.R. Ramakrishnan, Efficient Distribution of Quantum Circuits, in DISC 2021.

[15] R.G. Sundaram, H. Gupta, and C.R. Ramakrishnan, Distribution of Quantum Circuits Over General Quantum Networks, in Proc. of QCE, 2022.