Synthesis of CNOT minimal quantum circuits with topological constraints through ASP

Carla Piazza[1], Riccardo Romanello[1,*]

[1]Dipartimento di Scienze Matematiche, Informatiche e Fisiche, Università degli Studi di Udine

Abstract

Although with some yet severe limitations, Physical working Quantum Computers are becoming available for end users. Such devices are based on the rules of Quantum Mechanics, which state that physical systems evolve through unitary transformations. However, as in the classical case, in order to have a model of computation, such unitary evolutions are expressed/approximated inside Quantum Computers in terms of a finite set of one/two qubit operations (i.e., through a universal set of gates). Single qubit gates are fault-tolerant, while the same cannot be said for two qubit gates. Hence, unitary matrices adopted in Quantum Algorithms must be synthesized in terms of this universal set of operations to obtain a quantum circuit. In such synthesis procedure we prefer circuits with minimum number of qubits and with minimum circuit depth. Clifford+T universal set is one of the most adopted in the literature for synthesis. In such set we have 3 single qubit gates and the CNOT, which is a two qubit gate. Many efforts have been directed to devise algorithms that synthesize general unitary matrices into Clifford+T circuits. These algorithms usually tend to optimize circuit depth or eventually the number of T gates. Since two qubit gates are not fault tolerant, in this work we propose an ASP based technique to minimize the number of CNOT gates inside a Clifford+T circuit. Moreover, in real world quantum computers, qubit are usually connected to each other according to some particular topology, thus providing further constraints. Two qubit gates —hence, CNOT gate— can be directly applied only to pair of gates that are connected. Such constraint has to be taken into account during the synthesis of CNOT minimal circuits. We propose an ASP model to solve the problem of synthesizing CNOT minimal circuits under topological constraints. We provide experimental evidence of the scalability of our proposal.

Keywords

Quantum Circuit Synthesis, Answer Set Programming, CNOT Minimization, Clifford+T Synthesis, Topological Constraint

1. Introduction

Quantum computations are becoming more and more available to end users thanks to online platforms that allow to define and simulate quantum algorithms. However, in order scale on larger circuits it is necessary to increase the number of qubits Quantum Computers are able to manage. In fact, the so called *Quantum speed-up* [1] becomes interesting when a reasonably high number of qubits come into play. Once the problem of manipulating a a reasonable amount

AIxIA 23: International Conference of the Italian Association for Artificial Intelligence, November 6–9, 2023, Rome, Italy

*Corresponding author.

†This work is partially supported by PRIN project NiRvAna and GNCS-INdAM project “Algoritmi Quantistici su Grafi” CUP_E53C22001930001.

carla.piazza@uniud.it (C. Piazza); riccardo.romanello@uniud.it (R. Romanello)

https://users.dimi.uniud.it/~carla.piazza (C. Piazza); https://riccardoromanello.github.io (R. Romanello)

of qubits has been solved, it remains to consider the difficulty of compiling a quantum algorithm over a real physical device. In this paper we take into account Quantum Computers that are built using the superconducting qubits technology. In such architecture, qubits are edited using gates—unitary matrices. Nevertheless, not all unitary operations can be directly implemented in physical hardware. Up to now, only single and two qubit gates can be physically manufactured. The former are more reliable, but may still fail. The latter are error prone in the superconducting qubit environment, while are more reliable in architectures like photonics or Rydberg atoms. The operations that can be physically exploited inside a Quantum Computer are the elements of the *base of gates*. If a base can synthesize correctly any unitary $U \in \mathbb{C}^{2^n \times 2^n}$, then it is a *universal set of gates*. When a generic unitary matrix $U \in \mathbb{C}^{2^n \times 2^n}$ has to be used, it must be rewritten in terms of the given base, i.e., it has to be *synthesized*. The problem of synthesizing a generic unitary has a complexity that is polynomial in the size of the input unitary. Hence, it has a $\Omega(2^n)$ lower bound running time.

The synthesis problem is however more challenging than that, since circuit depth and width have to be taken into account. The circuit depth is the length of the circuit, while the width is another term for the number of qubits used. The former has to be kept low since the coherence time of quantum states is not really high, hence circuits that are too long are doomed to fail in their computation. The latter is strictly related to the issue we addressed at the beginning: number of qubits in physical Quantum Computers is currently limited. So, it is fundamental to design algorithms for synthesizing unitaries while minimizing the depth and width of the resulting circuit.

Most of the synthesis algorithms that have been defined are based on the Clifford+T set of universal gates. Such set is composed by 3 single qubit gates and a two-qubit gate. Different techniques have been adopted for the synthesis problem. For example, exact techniques with QMDD have been proposed in [2, 3, 4]. Also heuristics have been exploited trying to obtain a good balance between depth and width of the output circuits [5, 6]

Given the complexity of the synthesis problem it is also reasonable to consider sub-problems. One possibility is that of focusing into minimizing of the number of occurrences of one fixed gate of the universal set. In [7], the authors devised an algorithm to minimize the number of T gates into a circuit. A similar problem was tackled in [8] where the authors proposed a SAT approach for the minimization of the number of CNOT gates in a T-optimal circuit — a circuit where the optimal number of T gates was already achieved.

Such kind of problem is really important since the CNOT gate is the only two-qubit gate in the Clifford+T base set. Two qubit gates cannot be physically implemented without introducing errors. Hence, their minimization could also lead to a reduction of the effort required for Quantum Error Correction [9] and Verification [10].

For this reason, we consider the same problem as in [8], but we try to solve it with a different approach. We take as input a T-optimal circuit $\mathcal{C}$ made only of CNOT and T gates. We want to produce a circuit $\mathcal{C}'$ that preserves the behaviour of $\mathcal{C}$ and with the minimum number of CNOT gates. The approach in [8] starts by reducing the problem to a classical one using the notion of phase polynomial representation. Secondly, a satisfiability model is developed to obtain a solution to the following problem: can the circuit $\mathcal{C}$ be written with l CNOT gates? Hence, the model is queried for each possible value of l starting from 1. When the first solution is found, it is given as output. Starting from this approach, we propose a solution to the same problem with two major differences. The first is to encode the problem in an Answer Set Programming model instead that using a SAT encoding. The second is to introduce topological constraints on the CNOT that can be applied. Roughly speaking, in real world quantum computers, qubits are not all connected to each other, but they obey some particular topology. Due to this limitation, two-qubit gates cannot be applied to any pair of qubits. They can be applied only to qubits that are adjacent in such topology. Therefore, in this work we will give as input to the ASP model also a graph describing which pairs of qubits can actually be put in the same CNOT.

We want to stress the fact that this is work extends a previous proposal presented at CILC23 [11]. In such work we introduced two different models to solve the CNOT minimization problem without taking into account the topological constraints. Here, we introduce the topological constraints on one of such models and test it in comparison to the case without topological constraints.

The paper is structured as follows: Section 2 contains the definition of the problem and its translation to a classical one. Some results about circuit synthesis coming from the literature are presented in Section 3. Section 4 is used to describe the encoding we propose to solve the given problem. Notice that this section mirrors one section from [11] with an extra addition to handle the topological constraints. After that, Section 5 is devoted to experimental evaluation. Some conclusions and ideas for future works are drawn in Section 6. For space reason we omit the introduction of ASP, the reader may refer to [12].

2. Setting the Context

Modern Quantum Computers (like Osprey from IBM) have two important limits:

- Only a finite set of operations can be applied to the qubit
- The quantum coherence can be maintained for a short amount of time

The finite set of operations a Quantum Computer can apply is usually called *Universal Set of Gates*. Quantum Algorithms can be described as a single $2^n \times 2^n$ unitary matrix $\mathcal{U}$.

The process of rewriting $\mathcal{U}$ in terms of gates of a Universal Set of Gates is called *synthesis*.

The most used Universal Set of Gates for synthesis is the Clifford + T, which contains the following single qubit gates:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} \qquad S = \begin{pmatrix}1 & 0\\ 0 & i\end{pmatrix} \qquad T = \begin{pmatrix}1 & 0\\ 0 & e^{i\frac{\pi}{4}}\end{pmatrix}$$

and the two-qubit gate CNOT:

$$CNOT = \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}$$

With the following effect:

$$CNOT\,(a, b) = (a, a \oplus b)$$

where a, b are the *control* and the *target* of the CNOT gate, respectively. Its effect is twofold:

- it leaves the control qubit unchanged
- the target is flipped if the control is 1 — a XOR between control and target

Single qubit gates can be implemented in physical quantum computers with no errors. On the other hand, two-qubit gates introduce error that must be corrected using Quantum Error Correction (QEC) techniques. Hence, it is reasonable to think about synthesis techniques that aim at minimizing the number of CNOT gates in the resulting circuit.

In this paper, we will start from a circuit made of CNOT and T gates (same problem tackled in [8]), and we will minimize the number of CNOT gates. To do so, we first turn a *Quantum* problem, into a classical one using the following definition from [13]:

Definition 1 (phase polynomial representation). *The action of a {CNOT, T} circuit on the initial state $|x_1, x_2, \cdots x_n\rangle$ has the form:*

$$|x_1, x_2, \cdots x_n\rangle \mapsto e^{i\frac{\pi}{4}p(x_1,x_2,\cdots,x_n)} |g(x_1, x_2, \cdots x_n)\rangle$$

with $p(x_1, x_2, \cdots, x_n)$ defined as:

$$p(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{k} (c_i \; mod \, 8) f_i(x_1, x_2, \cdots x_n)$$

where $g : \mathbb{B}^n \to \mathbb{B}^n$ is a linear reversible function and p is a linear combination of linear boolean functions $f_i : \mathbb{B}^n \to \mathbb{B}$.
The coefficients c_i (reduced modulo 8) measure the number of $\pi/4$ rotations applied to each f_i. The number k represents the number of T gates in the circuit.

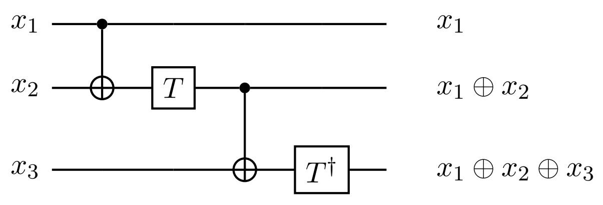
Each circuit has its phase polynomial representation, uniquely defined by

$$g, f_i, c_i \text{ for } i = 1, 2, \ldots k.$$

More than one {CNOT, T} circuit can share the same phase polynomial representation.

Since g is a linear reversible function with n inputs and n outputs, it can be written as a $n \times n$ boolean matrix G. On the other hand, each f_i can be expressed as a boolean row vector F_i.

Example 1. *Consider the following circuit:*



Its phase polynomial representation is the following:

$$G = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \qquad F_1 = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \qquad F_2 = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

□

Since the number of T gates in the input circuit is supposed to be optimal, the problem we want to solve is the following:

- **INPUT**: $G, S = (V_S, E_S), F_1, F_2, \cdots F_k$
- **OUTPUT**: a sequence of CNOT gates to be applied such that the final behaviour of the circuit is the one described by G.

The constraints we must fulfill are the following:

- We can only apply CNOT gates that are *legal* according to S.
- For each F_i with $i \in \{1, 2, \cdots, k\}$, there must exist a moment during the computation in which a row of G is exactly F_i.

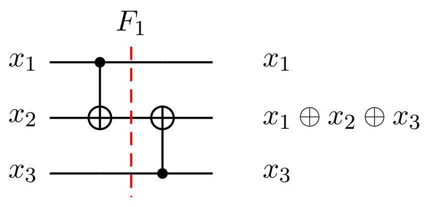
The two constraints allow us to avoid caring about the T gates. We just have to create the *correct* slots for them to be applied, but nothing more.

Example 2. *Let G and F_1 be defined as follows:*

$$G = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \qquad F_1 = (1\,1\,0)$$

We must find a circuit such that x_1 and x_3 are unchanged while x_2 becomes the XOR of all the three variables. Moreover, it must be true that at some point one of the variables must be the XOR between x_1 and x_2 to match the constraint on F_1.

The following circuit, made only of CNOT gates, solves the problem.



We pointed out with the red line the moment in which the constraint on F_1 is satisfied.

For what concerns the topological constraint, we adopted a graph based encoding.
Let $S = (V_S, E_S)$ be a directed graph. The set of nodes is $V_S = \{1, 2, \cdots, n\}$, where n is the number of qubit. The set of edges E_S is such that a CNOT(x_i, x_j) can be applied if and only if $(i, j) \in E_S$.

In Figure 1, we show an example of a topology turned into a graph. Only two CNOT gates are allowed: with control 1 and target 2 or with control 2 and target 3.

With this additional constraint, it is straightforward to see that the problem may become unsolvable in some cases. For example, if we need a CNOT between qubit x_i and x_j, there must exist a path in G from i to j. If such path does not exists, the problem is unsolvable.

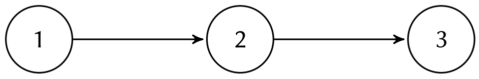


Figure 1: An example of a topology constraint depicted as a graph.

3. Related Works

Circuit synthesis has become a more and more prominent problem since the first physically working quantum computers were manufactured. A generic synthesis algorithm takes two inputs: a generic unitary matrix $U \in \mathbb{C}^{2^n \times 2^n}$ and a universal set of gates $\mathcal{B}$. The output must be a circuit made only of gates from $\mathcal{B}$ that implements exactly U. Clearly, we are not interested in any type of circuit. We are interested in keeping the circuit (i) as short as possible, because of the very volatile qubit state, (ii) as thin as possible to use the minimum number of qubits.

Different approaches and algorithms have been proposed. A large branch of them is composed by the ones using Quantum Multivalued Decision Diagrams (QMDD) [2].

QMDDs have been introduced as a means for the efficient representation and manipulation of quantum gates and circuits. The fundamental idea is a recursive partitioning of the respective transformation matrix and the use of edge and vertex weights to represent various complex-valued matrix entries. More precisely, a transformation matrix of dimension $2^n \times 2^n$ is successively partitioned into four sub-matrices of dimension $2^{n-1} \times 2^{n-1}$. This partitioning is represented by a directed acyclic graph - the QMDD.

In [4], the authors exploited the properties of QMDDs to devise a classical algorithm for the synthesis of circuits made of Clifford gates (Hadamard, Phase and CNOT). Such algorithm was extended for the Clifford+T case in [3].

Not only exact algorithms have been defined. In [5], the authors proposed an evolutionary (genetic) algorithm to solve the problem of circuit synthesis. Similar problem is solved in [14]. The optimization technique adopted was a Meet-in-the-Middle one, and in this case the problem of reducing the number of qubits used by synthesized circuit was tackled. An evolutionary algorithm was again used in [6]. In this case the authors proposed a quantum-inspired algorithm. It is worth noticing that in that proposal more than one universal set of gates is considered.

Synthesis problem of Clifford+T circuits is not always addressed as one big problem. A lot of authors also tackled some sub-problems like the minimization with respect to one single gate in the universal set. One example is the minimization of the CNOT gates.

For such problem, an algebraic approach is given in [15] where the authors start by first describing the group structure underlying quantum circuits generated by CNOT gates. Such results are then exploited to create heuristics useful to reduce the number of CNOT in circuits.

A technique based on Steiner Trees has been adopted in [16]. In this case, the authors also took into account the problem of the actual neighbour relation between qubits. When working with physical quantum devices, not every qubit is connected to all the others. Hence, when applying a CNOT between two qubit, we must be sure that these two qubits are linked. Otherwise, we must apply also swap gates, which correspond to 3 CNOT gates.

Last but not least, the SAT based approach has been proposed in [8]. Authors aimed at minimizing the number of CNOT gates in a circuit by setting up the solution of a satisfiability

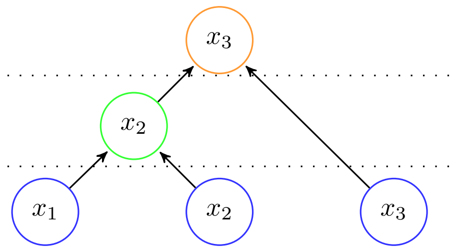


Figure 2: Blue nodes are in layer 0, the green node composes layer 1, and layer 2 is composed by the single orange node. The circuit induced by $\mathcal{G}$ applies CNOT(x_1, x_2) followed by CNOT(x_2, x_3).

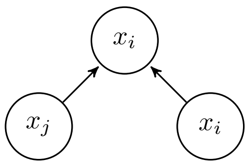


Figure 3: Generic DAG node describing CNOT(x_j, x_i).

problem. Such solution is then iteratively queried until a first feasible model is found and returned as result to the main problem. The SAT solver will eventually be queried for at most n^2 times, where n is the number of qubit, because of this the following result from [17]:

Theorem 1. *Any n-qubit circuit of CNOT gates is equivalent to one composed of at most n^2 gates.*

A SAT encoding is used also in [18]. The authors noticed that synthesis can be related to the problem of finding best boolean chains to represent functions. Hence, they proposed a DAG based approach to generate all possible DAG structures with a fix number of nodes and then query the satisfiability model to check if such structure would solve the problem.

4. ASP Model

The model we propose is based on a Directed Acyclic Graph. The circuit from Example 1 (without the T gates) can be interpreted as the labelled graph $\mathcal{G} = (V, E)$ depicted in Figure 4.

The leaves of the DAG represent the problem variables. Hence, we will have n leaves labelled $x_1, x_2, \cdots, x_n$. A CNOT with control x_j and target x_i is represented by a node x_i with two incoming edges. One edge comes from the closer node labelled x_i. The other from the closer node labelled x_j. The generic structure of a node is depicted in Figure 3.

The DAG structure induces a *layering* of the nodes. Leafs are at layer 0. A node at layer j, can only have incoming edges from nodes in smaller layers. One layer can contain more than a single node. Nevertheless, any layer l can contain at most one node labelled x_i, for any $i \in \{1, 2, \cdots, n\}$.

We solve the problem introduced in Section 2 by computing $\mathcal{G} = (V, E)$ starting from $G \in \{0, 1\}^{n \times n}$, a set of F_i, and some topological constraint $S = (V_S, E_S)$. The set of nodes V is always $\{x_1, x_2, \cdots x_n\}$. Our aim is to build the set E of minimum size l.

First of all, it must hold that for each layer $t \leq l$, each variable can be the target of at most one CNOT gate. After all the l steps, the following has to be true:

$$\textsc{val}_l(x_i) = G_i \qquad \forall i \in \{1, 2, \cdots, n\}$$

where VAL is defined as follows:

$$\begin{aligned}
&\textsc{val}_0(x_i) = x_i && \forall i \in \{1, 2, \cdots, n\} \\
&\textsc{val}_t(x_i) = \textsc{val}_{t-1}(x_i) && \text{if } \nexists x_k \mid (x_k, x_i, t) \in E \wedge t > 0 \\
&\textsc{val}_t(x_i) = \textsc{val}_{t-1}(x_i) \oplus \textsc{val}_{t-1}(x_k) && \text{if } \exists x_k \mid (x_k, x_i, t) \in E \wedge t > 0
\end{aligned}$$

With the notation $x_j \in \textsc{val}_t(x_i)$, we mean that the XOR operation described by $\textsc{val}_t(x_i)$ contains x_j.

The above constraint was necessary to ensure that the final circuit respects the limitations imposed by G.

As far as F_i is concerned, the following must hold:

$$\forall F_i \; \exists t \leq l \; \exists x_j \mid \textsc{val}_t(x_j) = F_i$$

In the end, we must ensure that CNOT are applied only to pair of qubits that are connected according to the input topology $S = (V_S, E_S)$. Hence, it must hold that if at layer $\hat{l}$ there is a CNOT(x_i, x_j), then $(i, j) \in E_S$.

4.1. Encoding in CLINGO

We now briefly describe how we encoded the aforementioned solution in the CLINGO solver.

The input has been encoded in a very simple way. We provide the model with n (the size of G) and k, the number of different F. Then, with a predicate of the form $\textsc{g}(i, j, b)$ we describe the matrix G, with the following rules:

$$\begin{aligned}
G_{i,j} = 1 \leftrightarrow \textsc{g}(i, j, 1) \\
G_{i,j} = 0 \leftrightarrow \textsc{g}(i, j, 0)
\end{aligned}$$

where $i, j \in \{1, 2, \cdots, n\}$

The same relation holds between predicate $\textsc{f}(i, j, b)$ and F. The only difference is that the domain of variable i is $\{1, 2, \cdots, k\}$.

The variables we adopted to solve the problem have two types:

- NODE that describes leafs of the DAG. Given an $n \times n$ boolean matrix, we will have exactly n NODES with labels $1, 2, \cdots, n$. The mapping from nodes to the variables x_i is straightforward: node i describes the variable x_i—the variable that will have to match G_i.
- LAYER, which describes the layers of the DAG.

Inner nodes of the DAG are described by a predicate of the form $\text{XOR\_NODE}(I, J, L)$ which has to be interpreted as follows: at layer L, there is a node labelled x_I which is the result of the XOR between x_I and x_J – CNOT(x_j, x_i). Given a variable X and a layer L, we imposed that a single XOR_NODE with target X can exist at layer L.

For what concerns the VAL of a node, we introduced the predicate

$$\text{VALUE}(X, I, Y)$$

where X, Y are NODES and I is a STEP.

$\text{VALUE}(X, I, Y)$ holds if and only if $x_Y \in \text{VAL}_I(x_X)$. We then encoded the update of VALUE using the recursive definition introduced above.

We used the predicate VALUE to check the final solution of the model at the l-th step. For all $i, j \in \{1, 2, \cdots, n\}$ the following must hold:

$$\text{G}(i, j, 1) \leftrightarrow \text{VALUE}(I, l, J)$$
$$\text{G}(i, j, 0) \leftrightarrow \neg\text{VALUE}(I, l, J)$$

Forcing the model to search for graphs that satisfy the total behaviour imposed by G.

Something similar has been done for the constraint on F, with the only difference that F_i can match the VAL of a NODE at any time. Hence, for all F_i there must exists a $t \leq l$ and a J such that for all $k \in \{1, 2, \cdots, n\}$:

$$\text{F}(i, k, 1) \leftrightarrow \text{VALUE}(J, t, K)$$
$$\text{F}(i, k, 0) \leftrightarrow \neg\text{VALUE}(J, t, K)$$

Handling the topological constraints induced by S is pretty straightforward. We encoded the edges of S with a binary predicate s.

$$\text{s}(x, y) \leftrightarrow (x, y) \in E_S$$

Using such predicate, we can enforce that $\text{XOR\_NODE}(X, Y)$ can hold if and only if $\text{s}(x, y)$.

5. Results

We tested the proposed model in two way. The former allowing any CNOT between the gates and the latter using the topological constraint forced by S. For each $n \in \{4, 5, 6, 7, 8\}$ we generated 10 different random tests. For each test we created an $n \times n$ matrix, representing a linear reversible function, that will serve as G – in the quantum case, n is the number of qubits, that evolve through $2^n \times 2^n$ unitary matrices. Moreover, for each test, we picked a number k between 1 and n, that will be the number of different F_i we give as input to each test. The topology of the qubit that we adopted is depicted in Figure 4. It is a smaller version of the topology adopted in the 16-qubit guadalupe Quantum Computer from IBM.

We tested the proposed models with the following procedure. We then run the two different models with an iterative procedure that works as follow. Let G, $\{F_i\}$ be the input for the current test case. We initialized a counter l to 1. We then run the graph model with input G, $\{F_i\}$ and

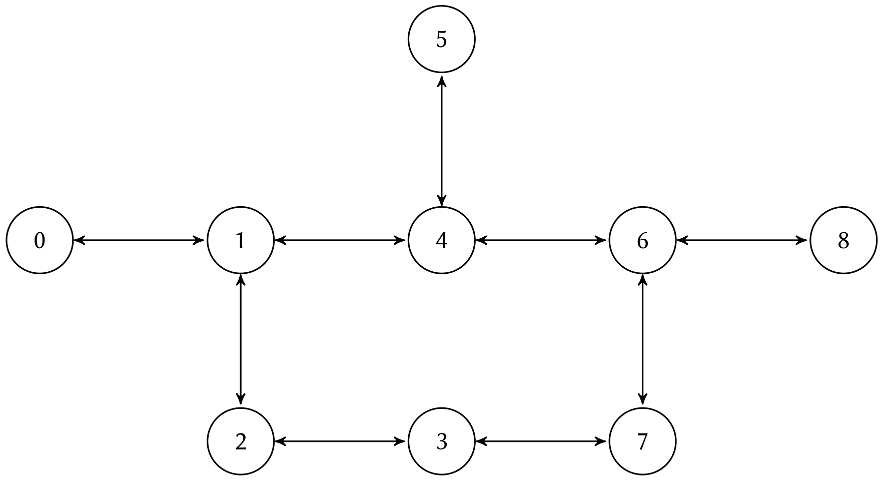


Figure 4: Guadalupe Quantum Computer topology reduced to 8-qubit.

l, to see if the problem was solvable in l steps. If true, then we stored the running time and move to next sample. Otherwise, we increased l and queried the model again. Notice that, by Theorem 1, l will at most be n^2.

In Table 1 we report the obtained results without the topology, while in Table 2 the results with the topology taken into account.

In both tables, the first column is the quantity n, the number of qubits. The second one is the average running time (in seconds) in the 10 tests. For each test, the time we measured is the overall time for the iterative procedure to find the solution, and not only the time for the satisfiable instance — even if we noticed that the unsatisfiability is found in less than 0.01 seconds in most of the cases. The tests have been run in a 2021 MacBookPro, with a 2 GHz Intel Core i5 quad-core CPU and 16GB of LPDDR4X Ram.

n	avg time (seconds)
4	0.023
5	0.702
6	20.212
7	45.548
8	98.721

Table 1
Results without the topology.

Results depicted in Table 1 show that the model with no topological constraint outperforms the constrained one (Table 2). Even if the test cases are rather small, it seems that forcing the model to use only a particular subset of all the possible CNOT makes the search for the optimal model harder. It is correct to notice that we only took into account one single ASP model and only a single qubit topology. In the future we would like to extend this round of tests by implementing faster models and testing them against different topologies.

n	avg time (seconds)
4	0.020
5	0.85
6	60.357
7	200.948
8	> 500

Table 2
Results with the topology.

6. Conclusions and Future Works

In this paper we extended the work presented in [11]. Our main goal was to introduce the notion of topological constraint into the ASP model. Doing so, we could mimic and study real Quantum Computer scenarios were qubits are connected according to some particular and strict topology. The model we presented is the same as in [11]. We only added some extra constraints to take the topology of the qubit into account. The results obtained show that the model with no topological constraint is faster than the other one. Despite this outcome, it is worth to study this complete version of the problem since it better describes real-life synthesis scenarios.

Despite that, it seems that Answer Set Programming can be fruitfully applied to the synthesis problem of quantum circuits in general, and not only in the CNOT case. Moreover, the solutions to this problem and solutions to problems such as the minimization of the number of T gates, paves down the road to the idea that the synthesis problem is not a monolith. Fast and not optimal techniques may be adopted to obtain a first and rough synthesis of some unitary $\mathcal{U}$. Starting from such synthesis, algorithms can be applied to systematically minimize parts of the circuits to obtain a reduction that is close to the optimal one. A fast and reliable synthesis (re-synthesis) algorithm would also allow to test a great variety of algorithms on real world quantum computers. For example, reachability on graphs [19] or the efficiency of quantum automata [20].

References

[1] L. K. Grover, A fast quantum mechanical algorithm for database search, 1996. arXiv:quant-ph/9605043.

[2] D. Miller, M. Thornton, Qmdd: A decision diagram structure for reversible and quantum circuits, in: 36th International Symposium on Multiple-Valued Logic (ISMVL'06), 2006, pp. 30–30. doi:10.1109/ISMVL.2006.35.

[3] P. Niemann, R. Wille, R. Drechsler, Advanced exact synthesis of clifford+t circuits, Quantum Information Processing 19 (2020). doi:10.1007/s11128-020-02816-0.

[4] P. Niemann, R. Wille, R. Drechsler, Efficient synthesis of quantum circuits implementing clifford group operations, in: 2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC), 2014, pp. 483–488. doi:10.1109/ASPDAC.2014.6742938.

[5] C. Ruican, M. Udrescu, L. Prodan, M. Vladutiu, A genetic algorithm framework applied to

quantum circuit synthesis, in: Nature Inspired Cooperative Strategies for Optimization, 2007.

[6] Y.-H. Chou, S.-Y. Kuo, Y.-C. Jiang, C.-H. Wu, J.-Y. Shen, C.-Y. Hua, P.-S. Huang, Y.-T. Lai, Y. F. Tong, M.-H. Chang, A novel quantum-inspired evolutionary computation-based quantum circuit synthesis for various universal gate libraries, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 2182–2189. doi:10.1145/3520304.3533956.

[7] M. Amy, D. Maslov, M. Mosca, Polynomial-time t-depth optimization of clifford+t circuits via matroid partitioning, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 33 (2014) 1476–1489. doi:10.1109/tcad.2014.2341953.

[8] G. Meuli, M. Soeken, G. D. Micheli, Sat-based {CNOT, T} quantum circuit synthesis, in: International Workshop on Reversible Computation, 2018.

[9] J. Roffe, Quantum error correction: an introductory guide, Contemporary Physics 60 (2019) 226–245. doi:10.1080/00107514.2019.1667078.

[10] L. Anticoli, C. Piazza, L. Taglialegne, P. Zuliani, Towards quantum programs verification: From quipper circuits to QPMC, in: S. J. Devitt, I. Lanese (Eds.), Reversible Computation - 8th International Conference, RC 2016, Proceedings, volume 9720 of *Lecture Notes in Computer Science*, Springer, 2016, pp. 213–219. doi:10.1007/978-3-319-40578-0_16.

[11] C. Piazza, R. Romanello, R. Wille, An asp approach for the synthesis of cnot minimal quantum circuits, volume 3428, 2023. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85164570648&partnerID=40&md5=b43c2519b280693049364e26a7554169, cited by: 0.

[12] W. Faber, An Introduction to Answer Set Programming and Some of Its Extensions, Springer International Publishing, Cham, 2020, pp. 149–185. doi:10.1007/978-3-030-60067-9_6.

[13] M. Amy, D. Maslov, M. Mosca, M. Roetteler, A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2013).

[14] M. Amy, D. Maslov, M. Mosca, M. Roetteler, A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 32 (2013) 818–830. doi:10.1109/tcad.2013.2244643.

[15] M. Bataille, Quantum circuits of CNOT gates: optimization and entanglement, Quantum Information Processing 21 (2022) 269. doi:10.1007/s11128-022-03577-8.

[16] V. Gheorghiu, J. Huang, S. M. Li, M. Mosca, P. Mukhopadhyay, Reducing the cnot count for clifford+t circuits on nisq architectures, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2022) 1–1. doi:10.1109/tcad.2022.3213210.

[17] M. Bataille, Quantum circuits of cnot gates, 2020. arXiv:2009.13247.

[18] W. Haaswijk, A. Mishchenko, M. Soeken, G. De Micheli, Sat based exact synthesis using dag topology families, in: Proceedings of the 55th Annual Design Automation Conference, DAC '18, Association for Computing Machinery, New York, NY, USA, 2018. doi:10.1145/3195970.3196111.

[19] D. D. Giustina, C. Piazza, B. Riccardi, R. Romanello, Directed graph encoding in quantum

computing supporting edge-failures, in: C. A. Mezzina, K. Podlaski (Eds.), Reversible Computation - 14th International Conference, RC 2022, Urbino, Italy, July 5-6, 2022, Proceedings, volume 13354 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 75–92. doi:10.1007/978-3-031-09005-9_6.

[20] C. Piazza, R. Romanello, Mirrors and memory in quantum automata, in: E. Ábrahám, M. Paolieri (Eds.), Quantitative Evaluation of Systems, Springer International Publishing, Cham, 2022, pp. 359–380.