MSc in Photonics

PHOTONICS**BCN**

Universitat Politècnica de Catalunya (UPC)
Universitat Autònoma de Barcelona (UAB)
Universitat de Barcelona (UB)
Institut de Ciències Fotòniques (ICFO)

http://www.photonicsbcn.eu

Master in Photonics

MASTER THESIS WORK

Contraction strategies of Tensor Networks for Quantum Circuit simulations

Marina Orta Terré

Supervised by Dr. Artur García Sáez, (BSC)

Co-supervised by Prof. Bruno Juliá Díaz, (UB)

Presented on date 31st September 2021

Registered at

Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona

Contraction strategies of Tensor Networks for Quantum Circuit simulations

Marina Orta Terré

BSC - Barcelona Supercomputing Center, 08034 Barcelona (Spain)

E-mail: marina.orta.terre@gmail.com

Abstract. In the last years, the classical simulation of quantum systems is growing as a good approach to provide efficient approximations to quantum computers that are in the way of reaching quantum supremacy. An efficient way of performing simulations is through the use of tensor networks which can be described as a countable collection of tensors connected by contractions. The main advantage of this approach is that a quantum circuit can be easily mapped into a tensor network. This work studies the contraction of tensor networks that correspond to quantum circuits with different geometries and sizes. The method used to find the contraction path also affects the results, so the same circuits have been contracted for each path finding method. The information extracted is the size of the largest intermediate tensor in the contraction rather than computing the contraction itself. In the results it has been seen that the method used, the geometry and the size of a circuit have an impact on the size of the largest intermediate tensor of a contraction.

Keywords: quantum computation, tensor network, classical simulation, quantum circuit, quantum supremacy

1. Introduction

Quantum computing is a type of computation that exploits the properties of quantum states and information theory [1, 2]. It is currently growing into a new field thanks to the promise that quantum computers might be able to outperform a classical processor and even solve previously thought intractable problems. This improvement over the classical computation is widely known as quantum supremacy and it implies the necessity for for quantum computers to be very difficult to simulate classically. The researchers that work on quantum technologies still have issues solve like decoherence, noise and errors that limit the possible entanglement and arise as the size and depth of the circuits increase [3], so this stage is still not a reality.

Since these issues make the actual quantum computers not perfect, the classical simulation opens up as a viable approach when providing efficient approximations of quantum systems. Also, they can be used to verify results from quantum computers and plays an important role on determining whether quantum supremacy is reached or not. In the simulation of quantum circuits, the tensor network (TN) approach plays an important role since there is a very direct relation between a quantum circuit and its corresponding TN.

1.1. Tensor networks

Tensors are a mathematical concept that generalises the idea of multilinear maps between sets of algebraic objects related to a vector space [4]. Following this, a TN is a countable collection of tensors connected by contractions. This concept can be graphically represented in a simple way using diagrams in which the tensors are represented by solid shapes and the tensor indices are represented by lines coming out from these shapes [5]. Following this, a scalar is a shape with no lines, a vector is a shape with a line and a matrix is a shape with two lines. The connection of two index lines implies a summation (or contraction) over the connected indices.

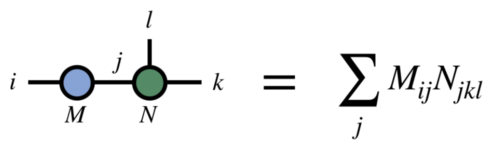

Figure 1. Diagram representation of a TN containing two tensors and the mathematical notation corresponding to the contraction. In this particular case, M is a matrix and N is a rank-3 tensor. Image extracted from [5].

Quantum circuits are a special class of tensor networks in which the arrangement of the tensors and their types are restricted. The graphical representation used is the same as before and, since in quantum information the operators can be expressed as matrices, the correspondence between circuit scheme and TN is quite direct. There are the single-qubit gates, which are 2×2 matrices and remain the same in a TN representation. Also there are two-qubit gates which are 4×4 matrices which in the TN representation are reshaped into rank-4 tensors with all 4 indexes having a dimension of 2.

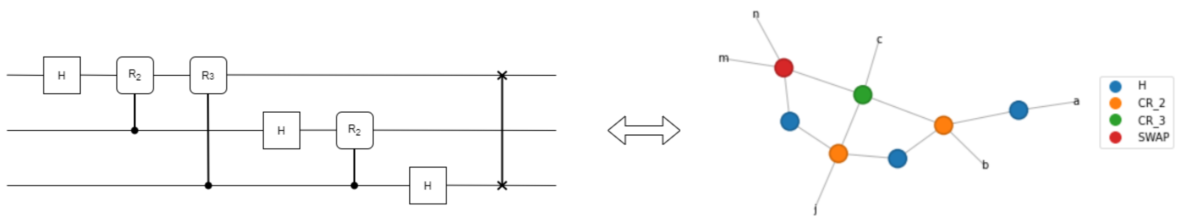

Figure 2. Quantum Fourier Transform circuit with $N = 3$ qubits (left) and its equivalent diagram representation of the TN (right). In the diagram representation becomes clear how the single-qubit gates (blue) are matrices and the 2-qubit gates (the rest) are rank-4 tensors, when looking at the indexes connected.

The contraction of a TN is done not directly but by steps. This generates new intermediate tensors that represent a part of the summation and which grow larger as the total size of the TN increases. The size of these intermediate tensors also varies depending on the order of the contraction steps, so finding the best possible contraction order (or path) becomes a central problem, with an exponential effect on computation time and memory footprint [6]. Taking this effect into account, new contracting methods and algorithms have arisen in order to increase the quality and speed of the contractions [7].

1.2. State of the art

In the last years, many simulators have been developed, with a large part of them that work by storing and evolving the whole state vector that describes the whole system [8, 9]. They implement different optimisations with special interest in running parallel operations, but this way of computing comes at an exponential memory cost. Consequently, these methods are very limited by the number of qubits that they can simulate.

At the same time, the use of TN for simulating quantum circuits grows. It was known that the absence of increasing entanglement in a quantum algorithm is enough to guarantee efficient simulations [10]. Then, the power of TN was used to identify classes of circuits that generate entanglement but still can be classically simulated. This was done by Markov and Shi [11], using the concept of treewidth, and Yoran and Short [12], using matrix product states representations. Finally, Jozsa [13] puts these results together and extends the families of efficiently simulable circuits. From these initial proposals, many efficient contraction strategies and approximated methods have been developed and improved [6, 14, 15].

1.3. Objectives

This project aims to study different types of quantum circuit geometries to determine which ones are more difficult to simulate. This is made considering different sizes (or number of qubits and depth) for the circuit geometries and also using different contraction methods implemented in the Python library `opt_einusm` [7]. The information extracted is the size of the largest intermediate tensor for each circuit size (presented in terms of the number of qubits and depth) and path finding method rather than performing the contraction itself.

2. Methodology

To determine the difficulty of different contractions, one wants to obtain the size of the largest intermediate tensor in the contraction of a TN for different geometries and contracting methods. The the size of the largest intermediate tensor is directly related with the cost of a contraction, so this will become a solid parameter to compare among different cases. The aim is to minimise the cost, so in the end one wants to obtain the lowest intermediate tensor sizes possible.

2.1. Opt_einsum library

To obtain the size of the largest intermediate tensor, it has been used the Python library `opt_einsum` [7], which has been created in order to optimise the execution time of einsum expressions. This library is specialised in the tensor contraction using different path finding algorithms. It also provides further information that can be obtained without the need of performing the contraction itself, like the contracting path or the size of the largest intermediate tensor which is the relevant information in this work.

For this, it has been used the function `contract_path()` which returns two variables: the `path` and the `PathInfo` which includes the size of the largest intermediate tensor. This function doesn't perform any contraction but as input still requires the full information of the TN: a string with all the indexes of all the arrays separated by a comma and the list of all the arrays to be contracted (using the same order as in the string with the indexes). Also, one can specify the path finding method.

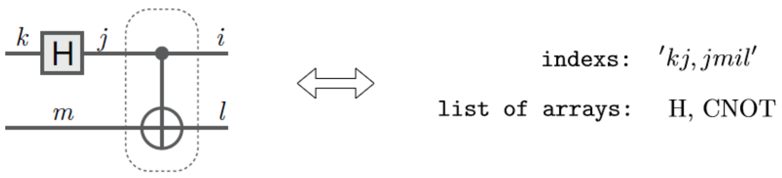

Figure 3. Illustration of the information that has to be provided to opt_einsum provided the circuit. Image in the left extracted from [4].

The optimisers used in this work are the following: `greedy`, `random-greedy`, `branch-1` and `branch-2`. Nevertheless, the two most accurate optimisers in the library are `optimal` and `branch-all`, which consist in an exhaustive search.

- `greedy`: this method is very efficient for contractions with a large number of tensors. It chooses one path at a time based on an heuristic cost function. The cost heuristic in a single contraction step is the `reduced_size`, which is the size of the pair of possible tensors to contract minus the size of the output tensor. This method consists in three steps:
  - (i) Compute any element-wise products, which requires the two matrices chosen to be the same dimensions.
  - (ii) Contract pairs of remaining tensors in such a way that at each step it is chosen the pair that maximises the `reduced_size`.
  - (iii) Compute any pairwise outer products choosing the pair that maximises the sum of input sizes.

This method is the one that scales better with a large number of tensors to contract and is the one used by default when there is a large number of inputs.

- `random_greedy`: this method is one of the bests for large and complex expressions since, on the one hand, the exhaustive methods (like `optimal` and `branch-all`) will become too slow, and on the other hand, the `greedy` path output can become very different from the optimal. This method randomly selects the contraction at each step by adding a Boltzman factor at each possible contraction taking as reference the one with the lowest cost. Proceeding like this, the `random-greedy` approach manages to explore a large amount of possible paths but in a guided way, since the paths with the higher cost will be the least likely to be explored.
- `branch`: the branching path adds an heuristic cost to the `optimal` path (which explores every possible way to contract the tensors) in order to sort the possible contractions and only saving the best one at each step. The branching path can have different types depending on the number of paths explored:
  (i) `branch-all`: explore all the possibilities starting with the ones that seem best according to the heuristic cost.
  (ii) `branch-2`: explore at each step the two best possibilities according to the cost heuristic. This explores 2^N paths (being N the number of tensors).
  (iii) `branch-1`: his method only explores the best looking path according to the heuristic cost. It doesn't really "branch", so it doesn't differ much from the greedy path.

The optimizers `optimal` and `branch-all` are not used in this work even though they are the most accurate due to the bad scaling with the number of tensors (like $N!$ being N the number of tensors to contract). With this scaling it soon becomes impossible to compute the contraction path, so here there are used methods that first try the "good" path and don't explore the "worst".

2.2. Circuit geometries

To build the circuits, it has been used the Python library `quimb` [16] which contains predefined gates. The circuits have been modelled by hand as defined previously using the library's gates.

There have been considered four geometries and five scenarios for this study. The first circuit considers two scenarios and the rest, one. The layout of the circuits has been constructed as follows: first apply a layer of single-qubit gates (one at each qubit) and then apply at least one two-qubit gate between some of them (this will vary according to the geometry). This pattern is repeated for each layer until reaching a given depth. The single-qubit gates used are: Hadamard (H), S and T. When a single-qubit gate is applied, the one used is selected randomly from this set. The two-qubit gate needs to be non-diagonal in order to create entanglement, and in this case it is chosen the CNOT.

At this point, one should consider only the layout of the circuits to be presented and not the number of qubits (N) or the depth (D) depicted in the figures. This has to be this way since the results have been computed using different sizes of the circuits. In other words, there have been taken different values of N and their corresponding D values for each geometry. This has been done taking into account that the size of a circuit is also a matter of interest.

Circuit 1

This is the only case where there are multiple two-qubit gates applied at the same time. They only act on neighbouring qubits and alternating the two neighbours they act on as depicted in Fig.4. For this geometry, there have been considered two scenarios where the geometry remains the same but the depth of the circuit is different. In one case $\mathrm{D} = O(\mathrm{N})$ and in the other, $\mathrm{D} = O(\log_2 \mathrm{N})$ following the main result of [13].

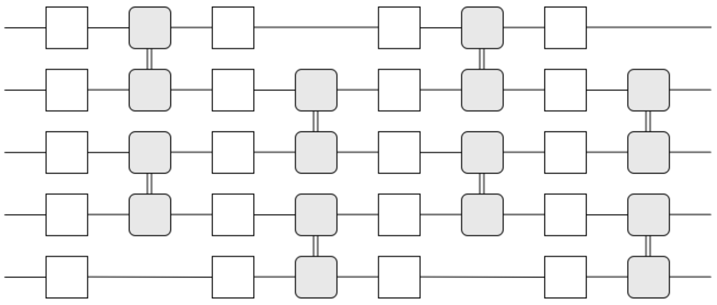

Figure 4. Circuit 1 geometry for $N = 5$ qubits. White squares represent single-qubit gates and grey linked squares represent two-qubit gates.

Circuit 2

The geometry of this circuit has one two-qubit gate for a given depth and these are placed in such a way that all qubits are connected through one of these gates. In this case the depth is not arbitrary and depends directly on the number of qubits of the circuits as $\mathrm{D} = \sum_{n=1}^{N-1} n = O(\mathrm{N}^2)$.

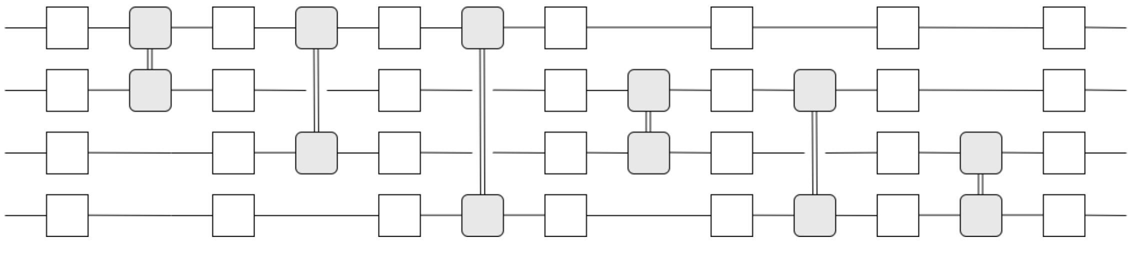

Figure 5. Circuit 2 geometry for $N = 4$ qubits. White squares represent single-qubit gates and grey linked squares represent two-qubit gates.

Circuit 3

This geometry is directly extracted from the work in [17] which uses the ordering ABCDCDAB. In this specific case, the number of qubits is fixed ($\mathrm{N} = 5$) and the only variable here is the depth, which is increased by repeating the same pattern.

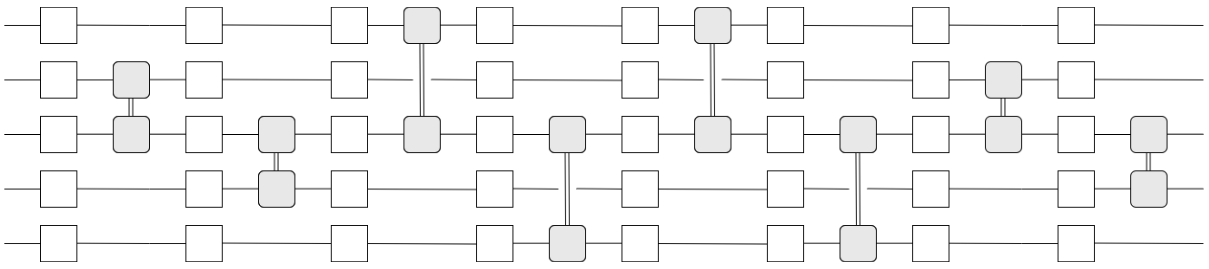

Figure 6. Circuit 3 geometry for $N = 5$ qubits. White squares represent single-qubit gates and grey linked squares represent two-qubit gates.

Circuit 4

This is the Quantum Fourier Transform circuit. In order to build this circuit there have to be used some specific gates which will be: H, $\mathrm{CR_k}$ and SWAP. Note that the $\mathrm{CR_k}$ gates are Control (the dots in Fig.7) and

$$\mathrm{R}_k = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{2\pi i/2^k} \end{pmatrix} , \tag{1}$$

which corresponds to the squares connected to the dots in Fig7. The SWAP gates are represented as two crosses linked by a line placed at the end of the circuit. The depth of this circuit is also directly known provided the number of qubits: $D = \sum_{n=1}^{N} n = O(N^2)$.

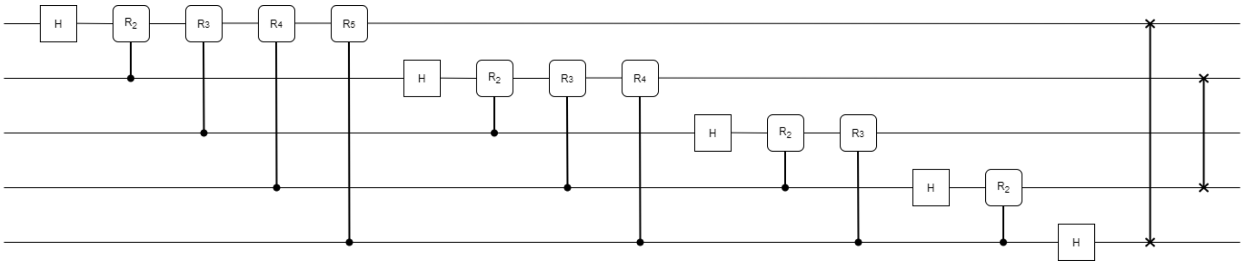

Figure 7. Circuit 4 geometry for $N = 5$ qubits. Single-qubit gates are Hadamard gates and the two-qubit gates are $\mathrm{CR_k}$, at the end there are SWAP gates.

3. Results

Following the methodology described in the previous section, here the results are presented for each method and circuit geometry. In each plot, one can see the the size of the largest intermediate tensor as a function of the product of the number of qubits N and depth D for each method. It has been chosen the product of N and D to give an idea of the size of the circuit and because not only the number of qubits needs to be taken into account but also the depth. The number of qubits of the circuits (except in the circuit 3 which is always N = 5) are N = $\{4, 9, 16, 25, 36\}$.

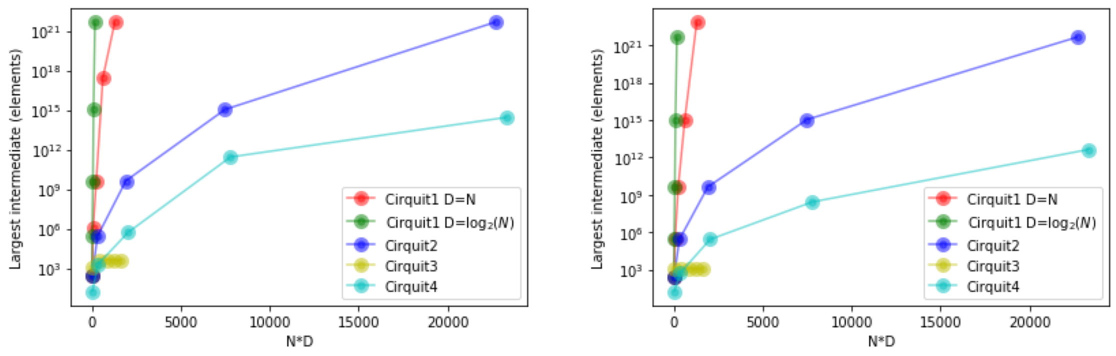

Figure 8. Size of the largest intermediate tensor as a function of the product of the number of qubits and the depth of the circuit. In the left panel there are the results for the greedy path and in the right panel there are the results for the random-greedy path.

In this figure the results for the `greedy` (left) and `random-greedy` (right) methods can be seen. The plots are similar and show the largest sizes for the geometry of the circuit 1. It can also be seen that the results are the same for low values of ND. For the large values, the `random-greedy` path improves the results for the geometry of the circuit 4 and some parts of the circuit 1 with the version D = N. In addition, there are no visible changes between the results of the circuit 1 with the version D = $\log_2$(N), the circuit 2 and the circuit 3. This last geometry occupies a small part of the plot since the number of qubits is fixed to N = 5 and the sizes become much lower than in the rest of cases.

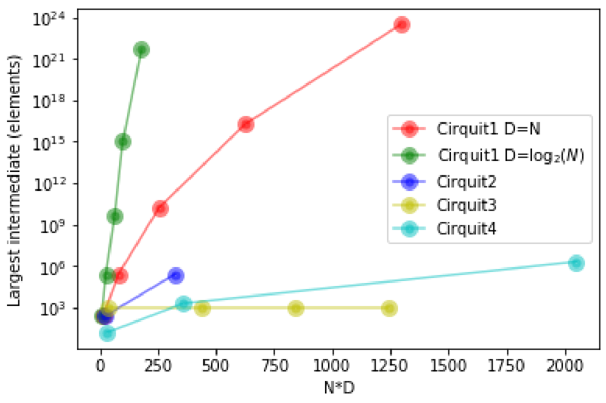

Figure 9. Size of the largest intermediate tensor as a function of the product of the number of qubits and the depth of the circuit using the branch-1 method.

In this figure there are the results of the method `branch-1`. Unfortunately, some points in this plot are not available because during the path calculations the recursion depth is exceeded before reaching any result and the method stops without any output. All the results for both cases of circuit 1 are available but in the rest of geometries there is a lack of information for the greatest values of the product ND. In the case of the `branch-2` path, something similar to the `branch-1` happens and only the few points can be calculated. With this scarce data it is impossible to generate a solid plot where information can be extracted.

Even though not having all the data in the `branch-1` plot, both Fig.8 and Fig.9 can be compared. The results for small circuit sizes are similar if not the same, so the impact of the path finding method used in a contraction becomes only relevant for large circuits: the `random-greedy` method improves the `greedy` for certain geometries and the `branch-1` is worse than the previous.

Regarding the depth, there are two remarks: on one hand, it has been observed that the depth does affect the result. For the largest values of ND, the largest intermediate tensor size is greater for circuit 1 with D $= N$ than for its shorter version. On the other hand, in the case of the geometry 3 with the `branch-1`, the increase of depth by repeating the same pattern (ABCDCDAB) multiple times doesn't impact on the intermediate tensor size. This might mean that the method is not a very good approximation since the largest intermediate tensor size is expected to grow when the depth is increased.

Another clear result is that the hardest geometry to contract is the circuit 1: even having the smallest ND values, the largest intermediate tensor sizes have the greatest values. Also, it is becomes clear that this geometry scales much more rapidly in contrast with the other layouts: the geometry of circuit 3 becomes stable very fast, and the geometry of circuits 2 and 4 still scale at much lower rates. This can be due to the fact that the geometry 1 is the only one that has more than one two-qubit gate in a layer. Specifically, there are about $N/2$ two-qubit gates applied each layer while in the rest there is only one. Following this same reasoning, the fact that the circuit 4 scales slower than the circuit 2 can be because of the difference in the quantity of single-qubit gates. Both geometries have a similar amount of two-qubit gates, but the number of single-qubit gates in the circuit 4 is much lower. Circuit 4 doesn't have single-qubit gates applied to all qubits each layer like in circuit 2.

4. Conclusions

In this work, it has been seen how the selection of a good contraction path along with the geometry of a TN affects its contraction cost. Particularly, it has been studied the contraction of quantum circuits with different geometries and circuit sizes (expressed as the product of the number of qubits N and the depth D) and also using different contraction methods. The information extracted is the size of the largest intermediate tensor of the contraction in terms of the product ND for different geometries and contracting methods.

One of the main results has been the fact that the largest intermediate tensor size scales very rapidly with ND. Also, it can be observed how the geometry has a great influence on the contraction cost, being the number of two-qubit gates per layer the variable that has the largest impact.

One outcome that is unexpected is the incapability of the `branch` methods to yield some of the results. For the largest ND values it becomes impossible to compute a contraction path even when the number of qubits in a circuit is not that high (N = 36 being the largest number of qubits). This stands with other works done in the same direction, where there is no "universal" path finding method to compute all the contractions and different methods are better for different types of TN.

The simulation of quantum computers is an area that offers a lot of potential but is still growing. The current results are encouraging despite the lack of generality and in the future will challenge quantum computers to be more perfect in order to reach the quantum supremacy.

Acknowledgements. I would like to thank my advisor Dr. Artur García for his help on the development of this project, his tracing on my work and his patience. Also, I would like to thank my family and friends for their support in this important stage in my personal and academic path.

References

[1] M. A. Nielsen and I. L. Chuang 2010 *Quantum computation and quantum information. 10th anniversary* (New York: Cambridge University Press).

[2] N. S. Yanofsky, An Introduction to Quantum Computing, arXiv:0708.0261 (2007).

[3] I. L. Markov, A. Fatima, S. V. Isakov and S. Boixo, Quantum Supremacy Is Both Closer and Farther than It Appears, arXiv:1807.10749 (2018).

[4] J. Biamonte and V. Bergholm, Quantum Tensor Networks in a Nutshell, arXiv:1708.00006 (2017).

[5] Tensor Network https://tensornetwork.org/diagrams/ Visited on 22/08/2021.

[6] J. Gray and S. Kourtis, *Quantum* **5**, 410 (2021).

[7] D. G. A. Smith and J. Gray, *Journal of Open Source Software* **3**, 753 (2018).

[8] G. G. Guerreschi, J. Hogaboam, F. Baruffa and N. P. D. Sawaya, *Quantum Sci. Technol.* **5**, 034007 (2020).

[9] H. De Raedt, et al. *Computer Physics Communications* **237**, 47 (2019).

[10] R. Jozsa and N. Linden, *Proc. R. Soc. Lond. A.* **459**, 2011 (2003).

[11] I. Markov and Y. Shi, *SIAM J. Comput* **38**, 963 (2009).

[12] N. Yoran and A. J. Short, *Phys. Rev. Lett.* **96**, 170503 (2006).

[13] R. Jozsa, On the simulation of quantum circuits, arXiv:quant-ph/0603163v1 (2006).

[14] Y. Zhou, E. M. Stoudenmire and X. Waintal, *Phys. Rev. X* **10**, 041038 (2020).

[15] S. Boixo, S. V. Isakov, V. N. Smelyanskiy and H. Neven, Simulation of Low-Depth Quantum Circuits as Complex Undirected Graphical Models, arXiv:1712.05384 (2018).

[16] J. Gray, *Journal of Open Source Software*, **3** 819 (2018).

[17] F. Arute, et al. *Nature*, **574** 505 (2019).