Journal club:

“Simulation of Quantum measurements with a gate-by-gate strategy”

Michael de Oliveira

Group: QLOC

February 23, 2022

INTERNATIONAL IBERIAN NANOTECHNOLOGY LABORATORY

Outline

- Simulation techniques
  - Qubit-by-qubit sampling
  - Gate-by-gate sampling
- Computational performance of simulation techniques
- Efficient simulation of surface code states on MBQC

How to simulate quantum measurement without computing marginals

Sergey Bravyi[1], David Gosset[2,3,4], and Yinchen Liu[2,3]

[1] IBM Quantum, IBM T.J. Watson Research Center
[2] Department of Combinatorics and Optimization, University of Waterloo
[3] Institute for Quantum Computing, University of Waterloo and
[4] Perimeter Institute for Theoretical Physics, Waterloo

We describe and analyze algorithms for classically simulating measurement of an n-qubit quantum state ψ in the standard basis, that is, sampling a bit string x from the probability distribution $|\langle x|\psi\rangle|^2$. Our algorithms reduce the sampling task to computing poly(n) amplitudes of n-qubit states; unlike previously known techniques they do not require computation of marginal probabilities. First we consider the case where $|\psi\rangle = U|0^n\rangle$ is the output state of an n-gate quantum circuit U. We propose an exact sampling algorithm which involves computing $O(m)$ amplitudes of n-qubit states generated by subcircuits of U spanned by the first $t = 1, 2, \ldots, m$ gates. We show that our algorithm can significantly accelerate quantum circuit simulations based on tensor network contraction methods or low-rank stabilizer decompositions. As another striking consequence we obtain an efficient classical simulation algorithm for measurement-based quantum computation with the surface code resource state on any planar graph, generalizing a previous algorithm which was known to be efficient only under restrictive topological constraints on the ordering of single-qubit measurements. Second, we consider the case in which ψ is the unique ground state of a local Hamiltonian with a spectral gap that is lower bounded by an inverse polynomial function of n. We prove that a simple Metropolis-Hastings Markov Chain mixes rapidly to the desired probability distribution provided that ψ obeys a certain technical condition, which we show is satisfied for all sign-problem free Hamiltonians. This gives a sampling algorithm which involves computing poly(n) amplitudes of ψ.

There is strong evidence that quantum circuits cannot be simulated efficiently using a classical computer. Likewise, physical properties of locally interacting quantum many-body systems are unlikely to be classically accessible in the general case. Nevertheless, classical simulation techniques are widely used in quantum computation and condensed matter physics. To some extent this is out of necessity, as a means to go beyond the limits of pen-and-paper calculation. But it is also facilitated by the fact that mathematicians, computer scientists, and physicists have identified certain remarkable quantum systems where efficient classical simulation is possible. These include the family of Clifford circuits (simulable using the stabilizer formalism [1]), systems that are equivalent to noninteracting fermionic particles including matchgate circuits [2, 3] and the 2D Ising model [4–6] (via fermionic linear optics [7]), gapped 1D quantum many-body systems [8, 9] or shallow quantum circuits in a 1D geometry [10] (tensor network methods [11–13]), and ferromagnetic spin systems [14–16] (Markov chain Monte Carlo methods). Such examples are rare and insightful: each provides a glimpse of a facet of the quantum-classical boundary and informs our understanding of hard-to-simulate quantum resources. Perhaps more importantly, the above algorithmic techniques can often be extended to more general settings with an increased computational cost. For example, the classical simulation algorithms based on low-rank stabilizer decompositions [17–19] have a runtime which scales exponentially only in the number of non-Clifford gates in a quantum circuit. Tensor-network based simulation methods for quantum circuits [13] have a runtime which scales exponentially only in the treewidth of a graph which describes the connectivity of the circuit. A large body of recent work (see, e.g., [20–26]) has focused on optimizing practical implementations of tensor network methods for the benchmark task of sampling from the output distribution of random quantum circuits, in response to the quantum experiment [27]. We expect classical simulation will continue to be a key technique for validation and verification of near-term quantum devices, and in the study of quantum many-body systems.

In this work we provide new techniques for a fundamental and ubiquitous task: simulating measurement of a quantum state ψ in the standard basis. Throughout we shall assume ψ is a normalized n-qubit quantum state and so our goal is to sample from the output distribution $|\langle x|\psi\rangle|^2$, where $x \in \{0,1\}^n$.

It is well known that this task can be performed given the ability to compute any marginal probability of the form

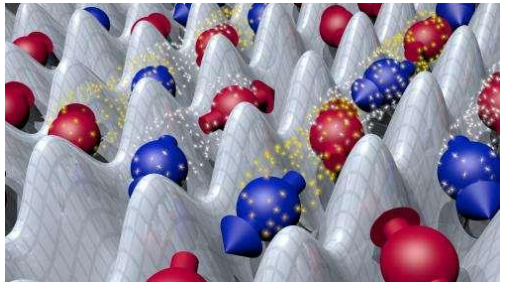
$$\pi_y(\psi) \equiv \langle\psi|(|y\rangle\langle y| \otimes I_{n-r})|\psi\rangle \qquad y \in \{0,1\}^r. \quad (1)$$

The standard *qubit-by-qubit* sampling algorithm uses the chain rule for conditional probabilities to simulate measurement of each qubit $j = 1, 2, \ldots, n$ in sequence. It samples each measurement outcome $x_j \in \{0,1\}$ for $j = 1, 2, \ldots, n$ from its conditional distribution given the values of all previously sampled bits. This qubit-by-qubit algorithm—stated formally as Algorithm 1 below—is the usual way to reduce the task of weak simulation (our sampling task) to strong simulation (computing a given

(Bravyi et al., 2021)

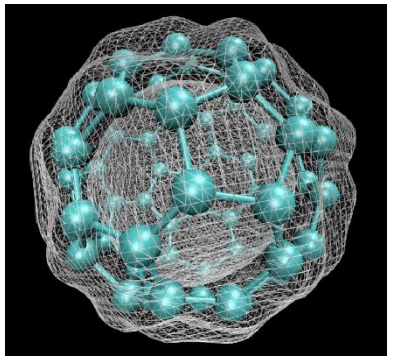
Quantum simulation

- In a simulation process, the objective is to replicate with some precision the quantum system to gain some insight into the object.
- Simulation techniques have provided significant advances, as they allow physicists to keep extending their computations and, therefore, learn more about the physical systems being simulated.



Credit: Thomas Uehlinger, ETH Zurich

Ultra-cold fermionic atoms



Itamblyn/CC BY-NC 4.0 via Wikimedia Commons

Buckminsterfullerene (C_{60})

Quantum simulation

Generally, when the simulation object is quantum computation, the goal is to acquire information about the output state ($|\psi_{out}\rangle$).

Weak simulation

- Samples from the output distribution $\big(P(x_1, x_2, \dots, x_n)\big)$.
- Provides the same information as measurements to the final quantum state.

Strong simulation

- Computes the precise probability associated with a measurement outcome.
- Provides more information as the quantum computation/experiment.

Quantum simulation

Generally, when the simulation object is quantum computation, the goal is to acquire information about the output state ($|\psi_{out}\rangle$).

Weak simulation

- Samples from the output distribution $\left(P(x_1, x_2, \dots, x_n)\right)$.
- Provides the same information as measurements to the final quantum state.

Strong simulation

- Computes the precise probability associated with a measurement outcome.
- Provides more information as the quantum computation/experiment.

Quantum simulation

Generally, when the simulation object is quantum computation, the goal is to acquire information about the output state ($|\psi_{out}\rangle$).

Weak simulation

- Samples from the output distribution $\left(P(x_1, x_2, \dots, x_n)\right)$.
- Provides the same information as measurements to the final quantum state.

Strong simulation

- Computes the precise probability associated with a measurement outcome.
- Provides more information as the quantum computation/experiment.

Research in simulation techniques increases the understanding of which elements/parameters make quantum computation difficult to simulate on a classical device.

Entanglement

(van den Nest et al., 2006)
(Jozsa, Richard; Linden, 2003)

Superposition

(Grover, 1998)

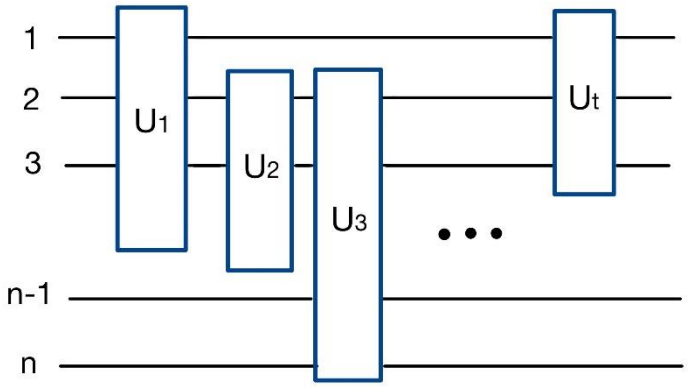
Contextuality

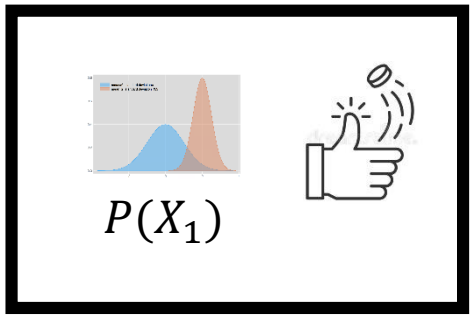
(Bermejo-Vega et al., 2017)

Qubit-by-qubit approach

This weak simulation method is composed of two steps:

- A process computing the probabilities of each qubit based on the previous sampling results.
- Sampling process for the state of a qubit based on the computed probability distributions.

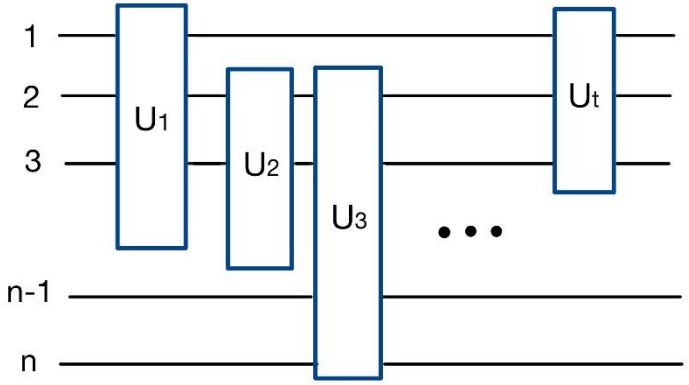


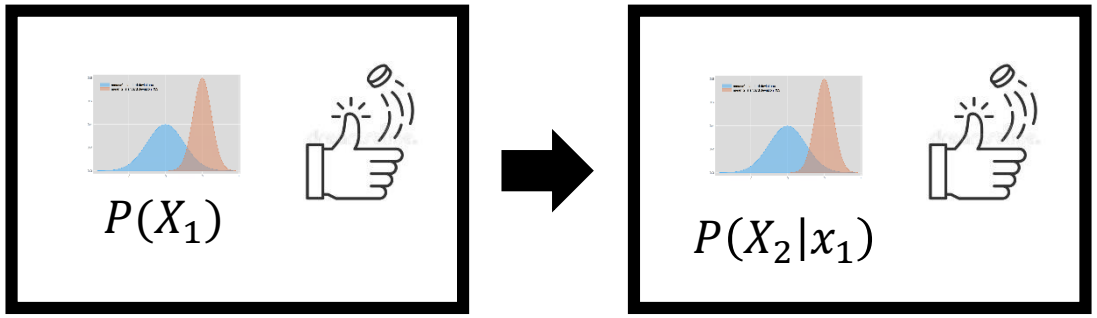


Qubit-by-qubit approach

This weak simulation method is composed of two steps:

- A process computing the probabilities of each qubit based on the previous sampling results.
- Sampling process for the state of a qubit based on the computed probability distributions.

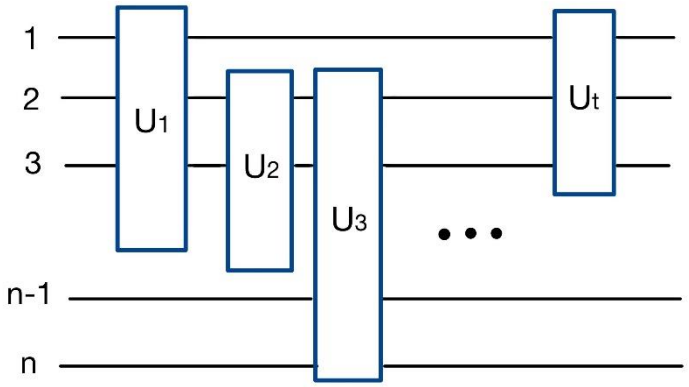


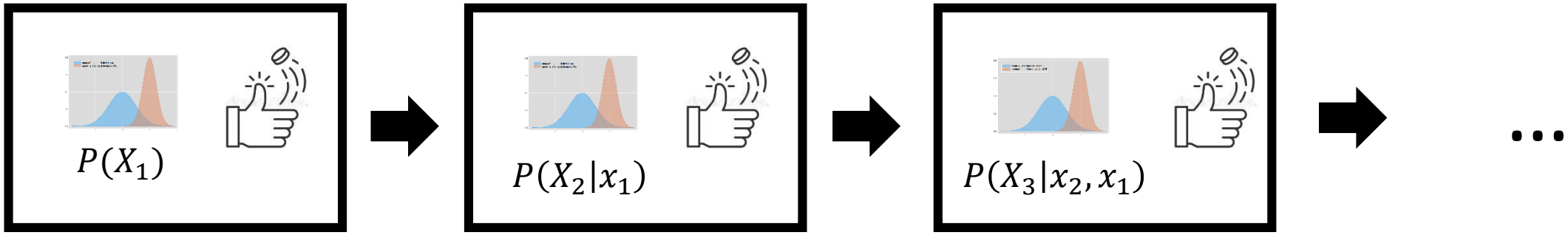


Qubit-by-qubit approach

This weak simulation method is composed of two steps:

- A process computing the probabilities of each qubit based on the previous sampling results.
- Sampling process for the state of a qubit based on the computed probability distributions.





Qubit-by-qubit approach

This process delivers each sample based on the correct probability distribution :

$$|\langle x|\psi\rangle|^2, x \in \{0,1\}^n$$

Algorithm 1 Qubit-by-qubit sampling

Input: An n-qubit quantum state ψ.
Output: $x \in \{0,1\}^n$ with probability $|\langle x|\psi\rangle|^2$.

1: Sample $x_1 \in \{0,1\}$ from the probability distribution $\pi_1(x_1)$.
2: **for** $j = 2$ to n **do**
3: Sample $x_j \in \{0,1\}$ from the probability distribution $\pi_j(x_1 \dots x_{j-1}x_j)/\pi_{j-1}(x_1 \dots x_{j-1})$.
4: **end for**
5: **return** $x = x_1 x_2 \dots x_n$

(Bravyi et al., 2021)

Qubit-by-qubit approach

This process delivers each sample based on the correct probability distribution :

$$|\langle x|\psi\rangle|^2, x \in \{0,1\}^n$$

Algorithm 1 Qubit-by-qubit sampling

Input: An n-qubit quantum state ψ.
Output: $x \in \{0,1\}^n$ with probability $|\langle x|\psi\rangle|^2$.

1: Sample $x_1 \in \{0,1\}$ from the probability distribution $\pi_1(x_1)$.
2: **for** $j = 2$ to n **do**
3: Sample $x_j \in \{0,1\}$ from the probability distribution $\pi_j(x_1 \dots x_{j-1}x_j)/\pi_{j-1}(x_1 \dots x_{j-1})$.
4: **end for**
5: **return** $x = x_1 x_2 \dots x_n$

(Bravyi et al., 2021)

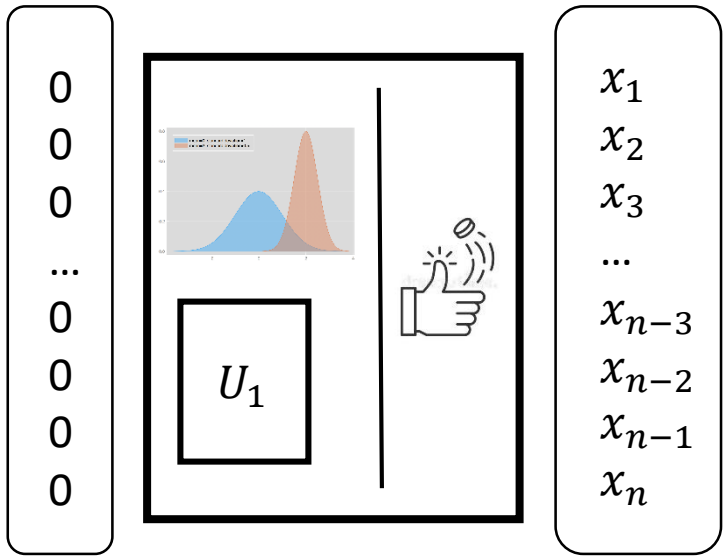
However, each of the marginal probabilities can be very computational expensive as the marginalization process has to be performed for all remaining variables:

$$P(X_j) = \sum_{\{1,2,\dots,n\}\backslash \mathrm{j}} P(X_1, X_2, \dots, X_n)$$

For some circuits, computing the marginals can be a **#P-hard problem** in the worst cases.

Gate-by-gate approach

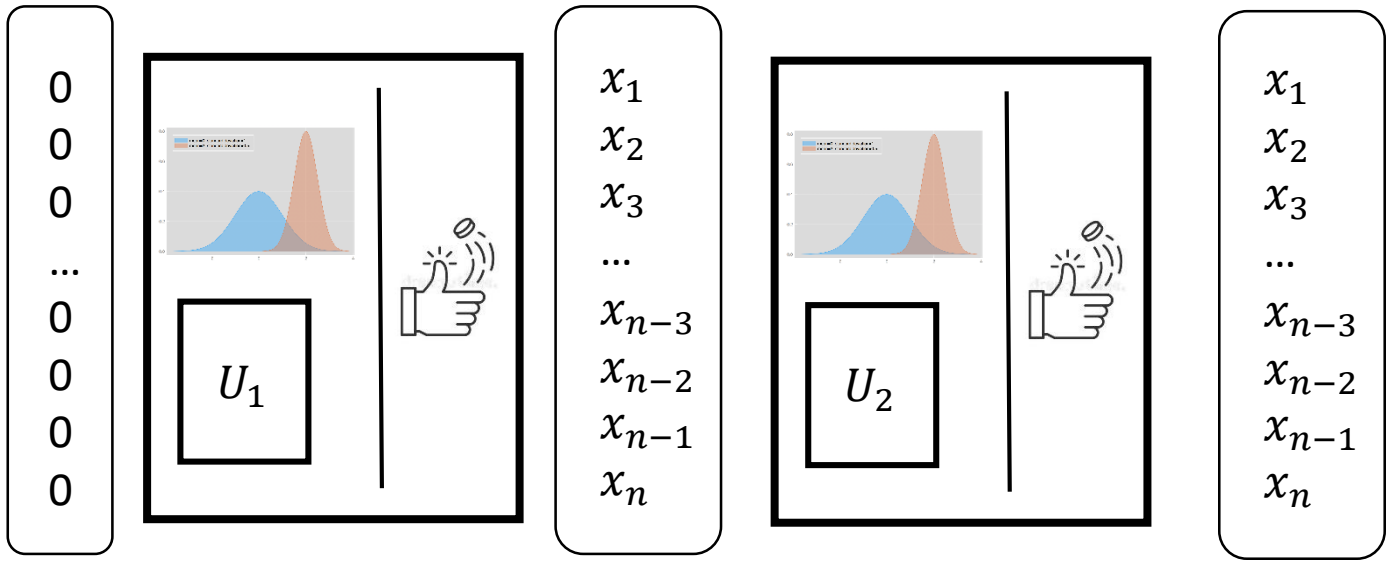
This approach takes advantage of the quantum circuit that generates the quantum state intended to be measured.



It computes the probability distribution for the bit-string after each operation and samples a new instance for the bit-string, so each step starts from a fixed bit-string.

Gate-by-gate approach

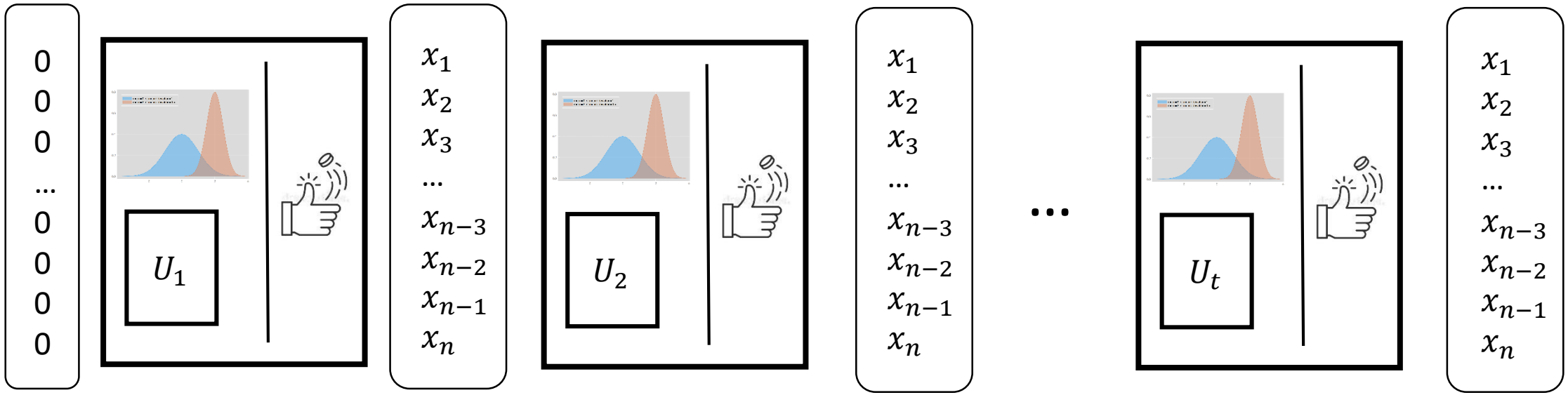
This approach takes advantage of the quantum circuit that generates the quantum state intended to be measured.



It computes the probability distribution for the bit-string after each operation and samples a new instance for the bit-string, so each step starts from a fixed bit-string.

Gate-by-gate approach

This approach takes advantage of the quantum circuit that generates the quantum state intended to be measured.



It computes the probability distribution for the bit-string after each operation and samples a new instance for the bit-string, so each step starts from a fixed bit-string.

Gate-by-gate approach

This method was proven to generate a bit-string with by correct probability distribution, obtaining the same results as the qubit-by-qubit simulation.

$$|\langle x|\psi\rangle|^2, x \in \{0,1\}^n$$

Algorithm 2 Gate-by-gate sampling

Input: An n-qubit quantum circuit $U = U_m \cdots U_2 U_1$.
Output: $x \in \{0,1\}^n$ with probability $|\langle x|U|0^n\rangle|^2$.

1: $x \leftarrow 0^n$
2: **for** $t = 1$ to m **do**
3: $\quad A \leftarrow \{1, 2, \ldots, n\} \setminus \mathrm{supp}(U_t)$
4: $\quad S \leftarrow \{y \in \{0,1\}^n : y_A = x_A\}$
5: $\quad$ Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y \in S} P_t(y)$
6: **end for**
7: **return** x

(Bravyi et al., 2021)

Gate-by-gate approach

This method was proven to generate a bit-string with by correct probability distribution, obtaining the same results as the qubit-by-qubit simulation.

$$|\langle x|\psi\rangle|^2, x \in \{0,1\}^n$$

Algorithm 2 Gate-by-gate sampling

Input: An n-qubit quantum circuit $U = U_m \cdots U_2 U_1$.
Output: $x \in \{0,1\}^n$ with probability $|\langle x|U|0^n\rangle|^2$.

1: $x \leftarrow 0^n$
2: **for** $t = 1$ to m **do**
3: $\quad A \leftarrow \{1, 2, \ldots, n\} \setminus \mathrm{supp}(U_t)$
4: $\quad S \leftarrow \{y \in \{0,1\}^n : y_A = x_A\}$
5: $\quad$ Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y \in S} P_t(y)$
6: **end for**
7: **return** x

(Bravyi et al., 2021)

During the computation of the probability distribution, some small tricks can simplify this process. However, it can be necessary to compute $m * 2^k$ operations to recover the amplitudes .

m: #Gates

k: #Interacting qubits

Gate-by-gate approach

This method was proven to generate a bit-string with by correct probability distribution, obtaining the same results as the qubit-by-qubit simulation.

$$|\langle x|\psi\rangle|^2, x \in \{0,1\}^n$$

Algorithm 2 Gate-by-gate sampling

Input: An n-qubit quantum circuit $U = U_m \cdots U_2 U_1$.
Output: $x \in \{0,1\}^n$ with probability $|\langle x|U|0^n\rangle|^2$.

1: $x \leftarrow 0^n$
2: **for** $t = 1$ to m **do**
3: $\quad A \leftarrow \{1, 2, \ldots, n\} \setminus \mathrm{supp}(U_t)$
4: $\quad S \leftarrow \{y \in \{0,1\}^n : y_A = x_A\}$
5: $\quad$ Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y \in S} P_t(y)$
6: **end for**
7: **return** x

(Bravyi et al., 2021)

During the computation of the probability distribution, some small tricks can simplify this process. However, it can be necessary to compute $m * 2^k$ operations to recover the amplitudes .

m: #Gates

k: #Interacting qubits

"In other words we give alternative efficient reductions from weak to strong simulation for these families of states" , Bravyi et al., 2021

Gate-by-gate approach (Robustness to error)

The authors have proven that errors do not scale with a multiplications rule!

Lemma 1 (Robustness to errors). *Let Q be the probability distribution describing the output of a modified version of Algorithm 2 in which the approximation R_t is used in place of P_t in line 5. Then*

$$\|Q - P_m\|_1 := \sum_{x \in \{0,1\}^n} |Q(x) - P_m(x)| \leq 16 \sum_{t=1}^{m-1} \epsilon_t. \quad (3)$$

The errors from each iteration are proportional to their sum by a small constant, supporting with this some imprecision from the computing device.

Gate-by-gate approach (Robustness to error)

The authors have proven that errors do not scale with a multiplications rule!

Lemma 1 (Robustness to errors). *Let Q be the probability distribution describing the output of a modified version of Algorithm 2 in which the approximation R_t is used in place of P_t in line 5. Then*

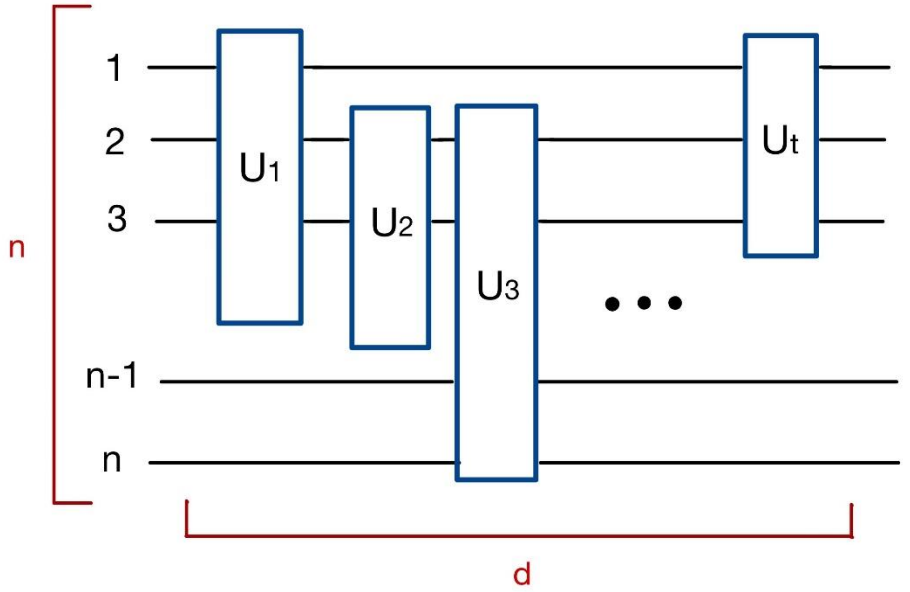
$$\|Q - P_m\|_1 := \sum_{x \in \{0,1\}^n} |Q(x) - P_m(x)| \leq 16 \sum_{t=1}^{m-1} \epsilon_t. \quad (3)$$

Important for practical implementations!

The errors from each iteration are proportional to their sum by a small constant, supporting with this some imprecision from the computing device.

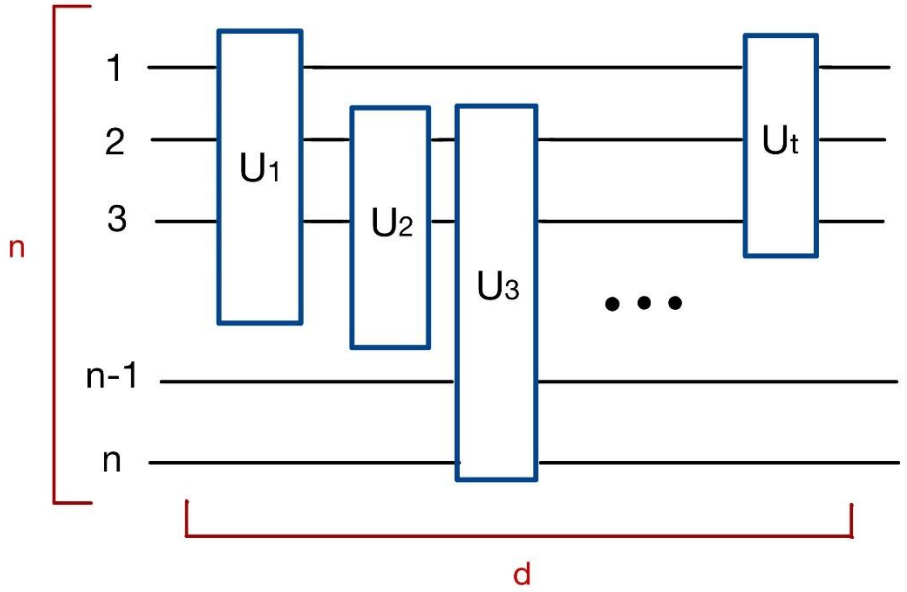
Computation performance

The computational cost of computing the amplitudes, in a strong simulation method, of a n-qubit circuit of depth d has a cost parametrized by those values equal to some $f(n, d)$.



Computation performance

The computational cost of computing the amplitudes, in a strong simulation method, of a n-qubit circuit of depth d has a cost parametrized by those values equal to some $f(n, d)$.



Notice that computing the marginals:

$$\langle 0^n | U^*(|y\rangle\langle y| \otimes I) U | 0^n \rangle$$

Does relate to a computational cost with the double of the depth, $f(n, 2d)$.

Computation performance

View this paper on arXiv.wiki:
https://arxiv.wiki/abs/2112.08499

3 comments

Michal Oszmaniec about a month ago
Magnificent paper!

I have one small question. On page 3, when comparing gate-by-gate and qubit-by-qubit simulations you state that the estimated cost of computing the marginal probability is f(n,2d), where f(n,d) is the cost of computing an amplitude of depth d circuit acting on n qubits. Why is the "operator depth" associated to the marginal probability upper bounded by 2d+1?

David Gosset about a month ago (2 points)
Thanks, Michal! To answer your question, say we are interested in the marginal probability for obtaining $x \in 0,1^{|A|}$ when measuring a subset $A \subseteq [n]$ of the qubits in the computational basis, starting from a state $U|0^n\rangle$ where U is a depth-d circuit. We can write this marginal as

$$\langle 0^n|U^\dagger(|x\rangle\langle x|_A \otimes I)U|0^n\rangle$$

Here U and $U^\dagger$ are each depth d by assumption and the projector $|x\rangle\langle x|$ is depth-1 since it is a tensor product of 1-qubit operators $|x_i\rangle\langle x_i|$ for $i \in A$. This is how we arrive at an upper bound on depth of $2d+1$.

Michal Oszmaniec in reply to David Gosset about a month ago
Thanks for clarification David! In my thinking I was "fixated" on density matrices and channels and it was harder to see this.

Computation performance

By this description of the relation between the parameters of the **strong simulation technique** and each one of the methods, we obtain that:

Qubit-by-qubit algorithm	→	$f(n, 2d)$
Gate-by-gate algorithm	→	$f(n, d)$

Computation performance

By this description of the relation between the parameters of the **strong simulation technique** and each one of the methods, we obtain that:

Qubit-by-qubit algorithm	→	$f(n, 2d)$
Gate-by-gate algorithm	→	$f(n, d)$

This creates the intuitive notion that there **can be a significant advantage** from the **gate-by-gate** to the **qubit-by-qubit** method related to the difference on the depth parameter d:

$$\frac{f(n, 2d)}{f(n, d)}^{*}$$

*Ratio is completely dependent on the strong simulation method.

Computation performance (Schrodinger)

If a Schrodinger simulation technique is used, the **amplitudes are stored in memory** with a $\mathbf{2^n}$ **size vector**, and a sparse matrix multiplication process updates these vectors.

$$\begin{pmatrix} c_1 \\ c_2 \\ \cdots \\ c_{2^n-1} \\ c_{2^n} \end{pmatrix} * \begin{pmatrix} u_{1,1} & \cdots & u_{1,2^n} \\ u_{1,1} & & u_{2,2^n} \\ \vdots & \ddots & \vdots \\ u_{2^n-1,1} & & u_{2^n-1,2^n} \\ u_{2^n,1} & \cdots & u_{2^n,2^n} \end{pmatrix} = \begin{pmatrix} c_1{}' \\ c_2{}' \\ \cdots \\ c_{2^n-1}{}' \\ c_{2^n}{}' \end{pmatrix}$$

Computation performance (Schrodinger)

If a Schrodinger simulation technique is used, the **amplitudes are stored in memory** with a $\mathbf{2^n}$ **size vector**, and a sparse matrix multiplication process updates these vectors.

$$\begin{pmatrix} c_1 \\ c_2 \\ \cdots \\ c_{2^n-1} \\ c_{2^n} \end{pmatrix} * \begin{pmatrix} u_{1,1} & \cdots & u_{1,2^n} \\ u_{1,1} & & u_{2,2^n} \\ \vdots & \ddots & \vdots \\ u_{2^n-1,1} & & u_{2^n-1,2^n} \\ u_{2^n,1} & \cdots & u_{2^n,2^n} \end{pmatrix} = \begin{pmatrix} c_1{}' \\ c_2{}' \\ \cdots \\ c_{2^n-1}{}' \\ c_{2^n}{}' \end{pmatrix}$$

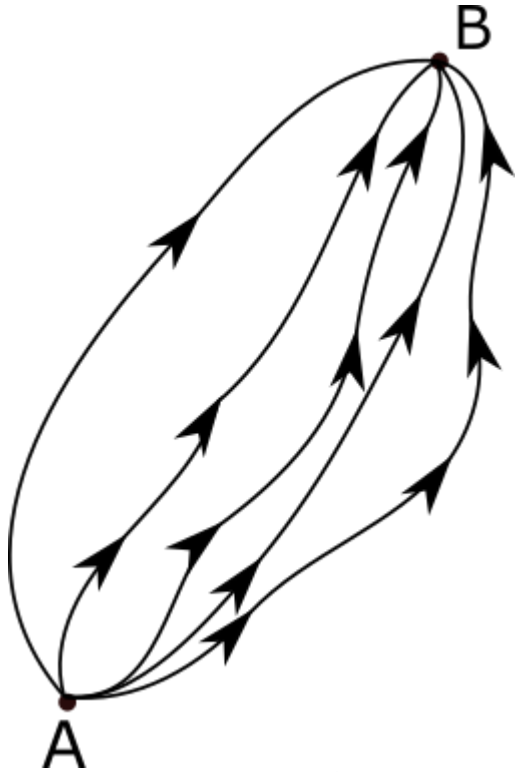
In those cases, the difference in the depth d between the qubit-by-qubit algorithm and gate-by-gate algorithm only translates to a **difference of constant size** on the computational effort.

$$\frac{f(n,2d)}{f(n,d)} = \frac{O(n2d2^n)}{O(nd2^n)} \approx \boxed{constant}$$

Computation performance (Feynman sum-over-paths)

If a computational method as the Feynman-sum-over-paths method* is used, which has **polynomial access to a memory resource**, then:

- This reduced memory access does increase the computational run-times with an exponential relation to the parameter d.



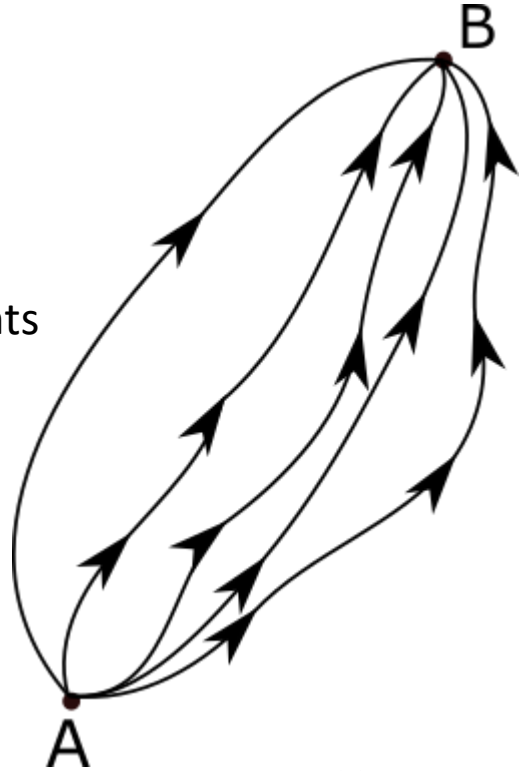
* (Malgieri et al., 2014)

Computation performance (Feynman sum-over-paths)

If a computational method as the Feynman-sum-over-paths method* is used, which has **polynomial access to a memory resource**, then:

- This reduced memory access does increase the computational run-times with an exponential relation to the parameter d.
- Then the distinction between the **qubit-by-qubit** and the **gate-by-gate** approach presents an **exponential difference** in the computational workload.

$$\frac{f(n,2d)}{f(n,d)} = \frac{O(n(4d)^{n+1})}{O(n(2d)^{n+1})} \approx \boxed{2^{n+1}}$$



* (Malgieri et al., 2014)

Computation performance (Experimental results)

- These **results** are based on **asymptotical growth rates**; however, in practical examples, these **divergences can be smaller** due to:
  - The specific characteristics of the circuits
  - Application of efficient heuristics

Computation performance (Experimental results)

- These **results** are based on **asymptotical growth rates**; however, in practical examples, these **divergences can be smaller** due to:
  - The specific characteristics of the circuits
  - Application of efficient heuristics
- The conducted **experiments verified** that for various memory restrictions the gate-by-gate approach displays an **authentic advantage**!

	$\log_2$ of max intermediate tensor size			
	29	31	33	35
$\log_2(F_G)$	58.4433	58.3197	58.1232	58.1339
$\log_2(F_Q)$	75.4501	73.1768	71.0325	68.9512
F_Q/F_G	131690	29677	7693	1804

TABLE I. FLOP count comparison for memory-limited tensor network simulation of a 2D depth-16 circuit on a 7×7 grid of qubits. Here F_G, F_Q are, respectively, the FLOP counts of the gate-by-gate and qubit-by-qubit algorithms.

(Bravyi et al., 2021)

Computation performance (Experimental results)

- These **results** are based on **asymptotical growth rates**; however, in practical examples, these **divergences can be smaller** due to:
  - The specific characteristics of the circuits
  - Application of efficient heuristics
- The conducted **experiments verified** that for various memory restrictions the gate-by-gate approach displays an **authentic advantage**!

*

	$\log_2$ of max intermediate tensor size			
	29	31	33	35
$\log_2(F_G)$	58.4433	58.3197	58.1232	58.1339
$\log_2(F_Q)$	75.4501	73.1768	71.0325	68.9512
F_Q/F_G	131690	29677	7693	1804

TABLE I. FLOP count comparison for memory-limited tensor network simulation of a 2D depth-16 circuit on a 7×7 grid of qubits. Here F_G, F_Q are, respectively, the FLOP counts of the gate-by-gate and qubit-by-qubit algorithms.

(Bravyi et al., 2021)

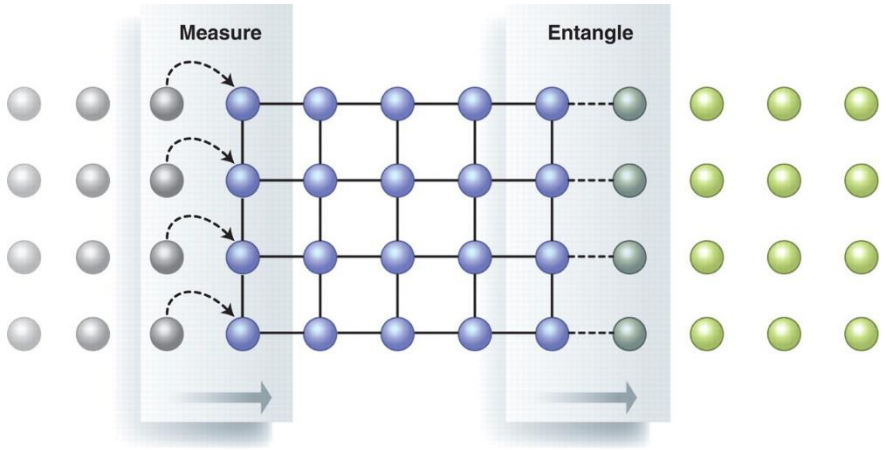
* The authors used various memory restrictions to test the hypothesis.

Applications

Additional to the discussion about the efficiency of the gate-by-gate approach, the authors identified two direct simulation tasks for which the approach provides notable advantages.

Applications

Additional to the discussion about the efficiency of the gate-by-gate approach, the authors identified two direct simulation tasks for which the approach provides notable advantages.

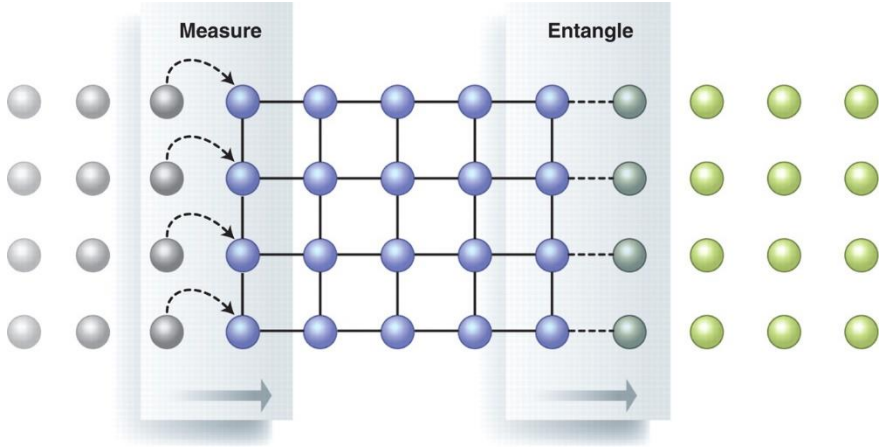


(L, 2007)

Measurement-based Quantum computing with surface code states

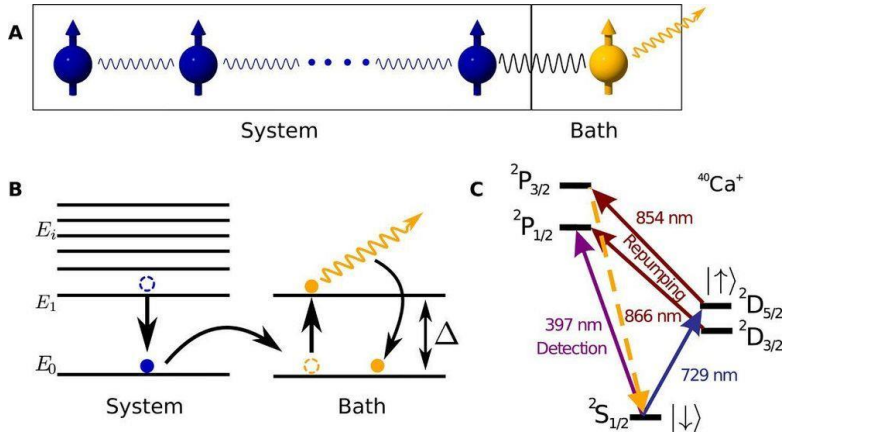
Applications

Additional to the discussion about the efficiency of the gate-by-gate approach, the authors identified two direct simulation tasks for which the approach provides notable advantages.



(L, 2007)

Measurement-based Quantum computing with surface code states

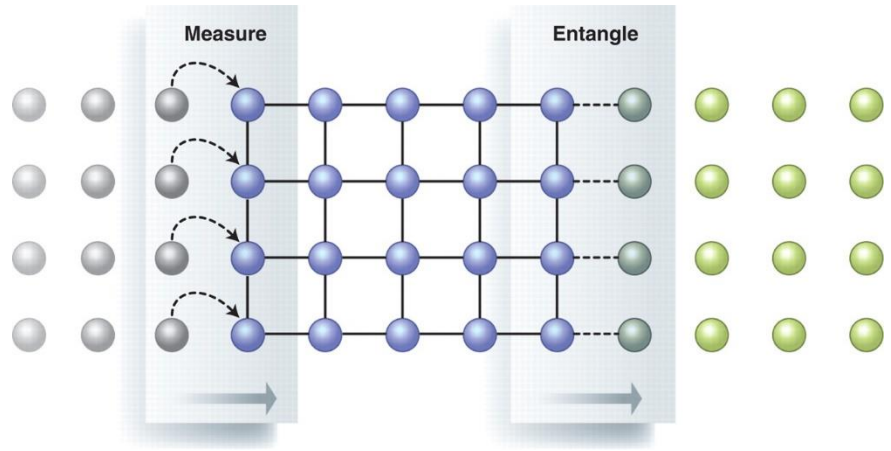


(Raghunandan et al., 2020)

Specific Hamiltonian ground state simulations

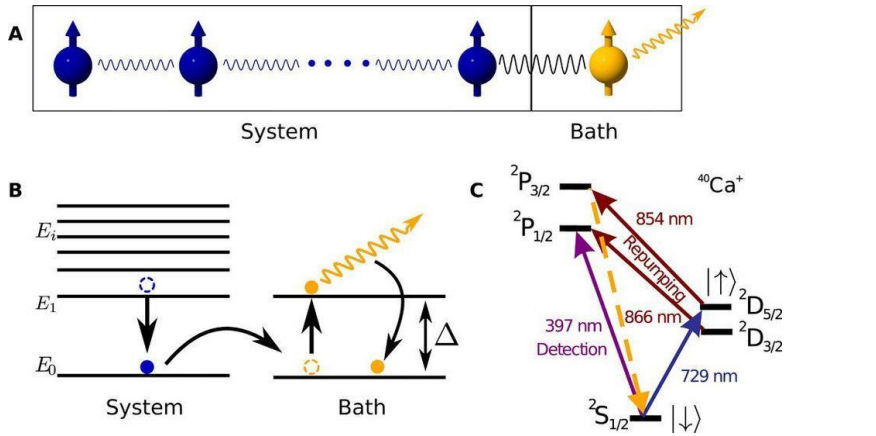
Applications

Additional to the discussion about the efficiency of the gate-by-gate approach, the authors identified two direct simulation tasks for which the approach provides notable advantages.



(L, 2007)

Measurement-based Quantum computing with surface code states

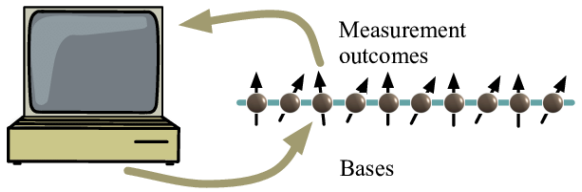


(Raghunandan et al., 2020)

Specific Hamiltonian ground state simulations

MBQC with surface code states

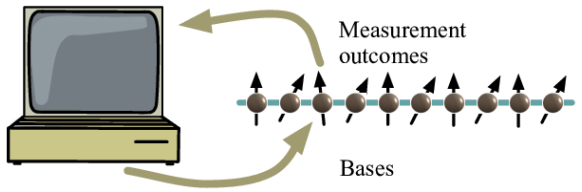
Remember that in an MBQC protocol, a sequence of measurements is applied to the resource state, and these measurements can depend on previous measurement results.



(Gross et al., 2009)

MBQC with surface code states

Remember that in an MBQC protocol, a sequence of measurements is applied to the resource state, and these measurements can depend on previous measurement results.



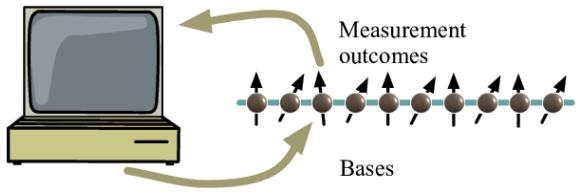
(Gross et al., 2009)

Therefore the probability of sampling a certain bit-string relates to the following expression,

$$P_t(x) = |\langle x|\langle(U_1 \otimes U_2(x_1) \otimes \cdots \otimes U_t(x_1, \ldots, x_{t-1}))|\psi_G\rangle|^2$$

MBQC with surface code states

Remember that in an MBQC protocol, a sequence of measurements is applied to the resource state, and these measurements can depend on previous measurement results.



(Gross et al., 2009)

Therefore the probability of sampling a certain bit-string relates to the following expression,

$$P_t(x) = |\langle x|\langle((U_1 \otimes U_2(x_1) \otimes \cdots \otimes U_t(x_1, \ldots, x_{t-1}))|\psi_G\rangle|^2$$

This set of operations is not previously defined, such that the first algorithm presented for the gate-by-gate simulation technique does not work.

MBQC with surface code states (Gate-by-gate)

The gate-by-gate simulation approach can be adjusted for an MBQC protocol to handle the adaptive selection process of measurement operators.

Algorithm 3 Simulate MBQC with the surface code state ψ_G

1: Sample x from $P_0(x) = |\langle x|\psi_G\rangle|^2$
2: **for** $t = 1$ to n **do**
3: $\quad S \leftarrow \{x, x \oplus e^t\}$
4: $\quad$ Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y\in S} P_t(y)$
5: **end for**
6: **return** x

The algorithm does not start from the 0^n string, but with a sample of the resource state.

(Bravyi et al., 2021)

MBQC with surface code states (Gate-by-gate)

The gate-by-gate simulation approach can be adjusted for an MBQC protocol to handle the adaptive selection process of measurement operators.

Algorithm 3 Simulate MBQC with the surface code state ψ_G

1: Sample x from $P_0(x) = |\langle x|\psi_G\rangle|^2$
2: **for** $t = 1$ to n **do**
3: $S \leftarrow \{x, x \oplus e^t\}$
4: Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y\in S} P_t(y)$
5: **end for**
6: **return** x

(Bravyi et al., 2021)

The algorithm does not start from the 0^n string, but with a sample of the resource state.

Each measurement operator will be simulated sequentially.

MBQC with surface code states (Gate-by-gate)

The gate-by-gate simulation approach can be adjusted for an MBQC protocol to handle the adaptive selection process of measurement operators.

Algorithm 3 Simulate MBQC with the surface code state ψ_G

1: Sample x from $P_0(x) = |\langle x|\psi_G\rangle|^2$
2: **for** $t = 1$ to n **do**
3: $S \leftarrow \{x, x \oplus e^t\}$
4: Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y\in S} P_t(y)$
5: **end for**
6: **return** x

The algorithm does not start from the 0^n string, but with a sample of the resource state.

(Bravyi et al., 2021)

Each measurement operator will be simulated sequentially.

Since the values for the previous measurements are permanently fixed, the measurement operators are regular (non-adaptive) single-qubit gates.

MBQC with surface code states (Gate-by-gate)

The gate-by-gate simulation approach can be adjusted for an MBQC protocol to handle the adaptive selection process of measurement operators.

Algorithm 3 Simulate MBQC with the surface code state ψ_G

1: Sample x from $P_0(x) = |\langle x|\psi_G\rangle|^2$
2: **for** $t = 1$ to n **do**
3: $S \leftarrow \{x, x \oplus e^t\}$
4: Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y\in S} P_t(y)$
5: **end for**
6: **return** x

(Bravyi et al., 2021)

The algorithm does not start from the 0^n string, but with a sample of the resource state.

Each measurement operator will be simulated sequentially.

Since the values for the previous measurements are permanently fixed, the measurement operators are regular (non-adaptive) single-qubit gates.

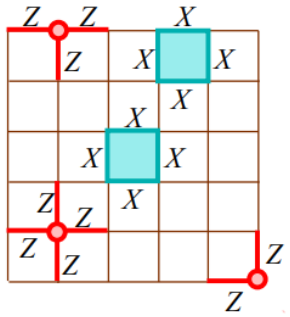
MBQC with surface code states (Gate-by-gate)

Why has this algorithm fixed the **surface code states** as the resource state?

$$A_s = \prod_{e \in \delta_s} Z_e \qquad B_p = \prod_{e \in \partial_p} X_e$$

Edges incident to vertex s

Edges of the boundary of the plaquette p



2D square lattice surface state (Bravyi & Raussendorf, 2007)

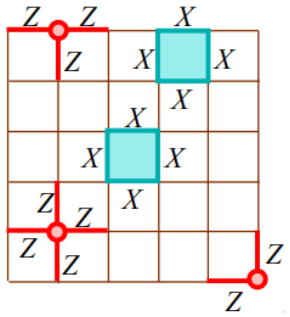
MBQC with surface code states (Gate-by-gate)

Why has this algorithm fixed the **surface code states** as the resource state?

$$A_s = \prod_{e \in \delta_s} Z_e \qquad B_p = \prod_{e \in \partial_p} X_e$$

Edges incident to vertex s

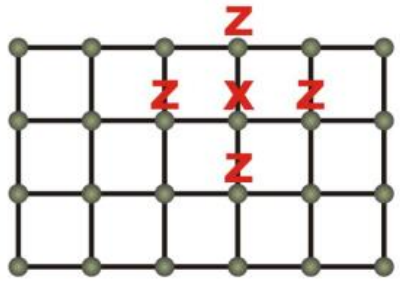
Edges of the boundary of the plaquette p



2D square lattice surface state (Bravyi & Raussendorf, 2007)

And not **2D cluster states**, which are universal,

$$K_a = X_a \bigotimes_{b|(a,b)\in E(C)} Z_b \quad , \forall_a \in V(C_L)$$



2D cluster state (Briegel et al., 2009)

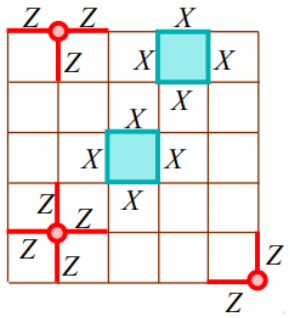
MBQC with surface code states (Gate-by-gate)

Why has this algorithm fixed the **surface code states** as the resource state?

$$A_s = \prod_{e \in \delta_s} Z_e \qquad B_p = \prod_{e \in \partial_p} X_e$$

Edges incident to vertex s

Edges of the boundary of the plaquette p

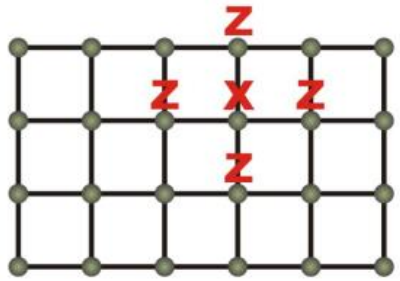


2D square lattice surface state (Bravyi & Raussendorf, 2007)

Non-universal resource for MBQC

And not **2D cluster states**, which are universal,

$$K_a = X_a \bigotimes_{b|(a,b)\in E(C)} Z_b \quad , \forall_a \in V(C_L)$$



2D cluster state (Briegel et al., 2009)

MBQC with surface code states (Gate-by-gate)

The process associated with computing the probability of each sampling process:

$$P(x_i) = |\langle \Phi | \psi_G \rangle|^2 \ , with \ |\Phi\rangle = \otimes_{j \in E} |\phi_j\rangle$$

Single-qubit states

$$, and \ |\psi_G\rangle = \frac{1}{\sqrt{|Z(G)|}} \sum_{x \in Z(G)} |x\rangle$$

Cycles in G

MBQC with surface code states (Gate-by-gate)

The process associated with computing the probability of each sampling process:

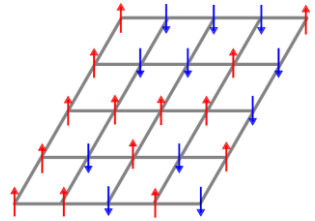
$$P(x_i) = |\langle \Phi | \psi_G \rangle|^2 \ , with \ |\Phi\rangle = \otimes_{j \in E} |\phi_j\rangle$$

Single-qubit states

$$, and \ |\psi_G\rangle = \frac{1}{\sqrt{|Z(G)|}} \sum_{x \in Z(G)} |x\rangle$$

Cycles in G

- If the resource state of the type of a **surface code state**, then it was proven in (Bravyi & Raussendorf, 2007) that this amplitude could be computed with an **equivalent** process to computing the **partition function of an Ising model**.



(Wald, 2017)

MBQC with surface code states (Gate-by-gate)

The process associated with computing the probability of each sampling process:

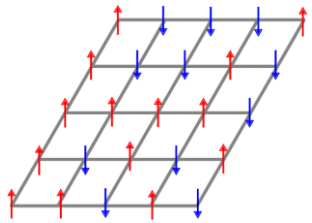
$$P(x_i) = |\langle \Phi | \psi_G \rangle|^2 \ , with \ |\Phi\rangle = \otimes_{j \in E} |\phi_j\rangle$$

Single-qubit states

$$, and \ |\psi_G\rangle = \frac{1}{\sqrt{|Z(G)|}} \sum_{x \in Z(G)} |x\rangle$$

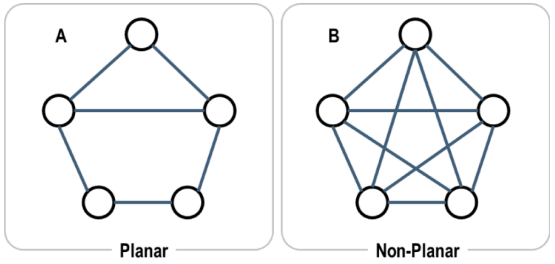
Cycles in G

- If the resource state of the type of a **surface code state**, then it was proven in (Bravyi & Raussendorf, 2007) that this amplitude could be computed with an **equivalent** process to computing the **partition function of an Ising model**.



(Wald, 2017)

- Additional to this equivalence, it was proven that the overlaps of the measurements and surface state with planar graphs are instances of these partition functions, which are **computed in O(n^3) time** (Barahona, 1982).



MBQC with surface code states (Gate-by-gate)

The computation associated with selecting the measurement operator is limited to be polynomial by the MBQC protocol.

Algorithm 3 Simulate MBQC with the surface code state ψ_G

1: Sample x from $P_0(x) = |\langle x|\psi_G\rangle|^2$ → $O(n)$
2: **for** $t = 1$ to n **do**
3: $\quad S \leftarrow \{x, x \oplus e^t\}$
4: $\quad$ Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y\in S} P_t(y)$ → $O(n^3)$
5: **end for**
6: **return** x

(Bravyi et al., 2021)

The number of faces f in the graph is smaller then n.

MBQC with surface code states (Gate-by-gate)

The computation associated with selecting the measurement operator is limited to be polynomial by the MBQC protocol.

Algorithm 3 Simulate MBQC with the surface code state ψ_G

1: Sample x from $P_0(x) = |\langle x|\psi_G\rangle|^2$ → $O(n)$
2: **for** $t = 1$ to n **do**
3: $S \leftarrow \{x, x \oplus e^t\}$
4: Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y\in S} P_t(y)$ → $O(n^3)$
5: **end for**
6: **return** x

(Bravyi et al., 2021)

The number of faces f in the graph is smaller then n.

MBQC with surface code states (Gate-by-gate)

The computation associated with selecting the measurement operator is limited to be polynomial by the MBQC protocol.

Algorithm 3 Simulate MBQC with the surface code state ψ_G

1: Sample x from $P_0(x) = |\langle x|\psi_G\rangle|^2$ → $O(n)$
2: **for** $t = 1$ to n **do**
3: $S \leftarrow \{x, x \oplus e^t\}$
4: Sample $x \in S$ from the probability distribution $P_t(x)/\sum_{y\in S} P_t(y)$ → $O(n^3)$
5: **end for**
6: **return** x

(Bravyi et al., 2021)

The number of faces f in the graph is smaller then n.

Therefore the gate-by-gate technique can simulate any surface code state with a planar graph in time $\boldsymbol{O(tn^4)}$

MBQC with surface code states (Qubit-by-qubit)

For the qubit-by qubit technique, the probabilities of each qubit are associated with:

$$|\langle\Phi|\rho_M|\Phi\rangle|^2 \text{ , with } |\Phi\rangle = \otimes_{j\in M}|\phi_j\rangle$$

$$\text{, and } \rho_M = Tr_{j\notin M}|\psi_G\rangle\langle\psi_G|$$

Subset of qubits to be measured

MBQC with surface code states (Qubit-by-qubit)

For the qubit-by qubit technique, the probabilities of each qubit are associated with:

$$|\langle\Phi|\rho_M|\Phi\rangle|^2 \text{ , with } |\Phi\rangle = \otimes_{j\in M}|\phi_j\rangle$$

$$\text{, and } \rho_M = Tr_{j\notin M}|\psi_G\rangle\langle\psi_G|$$

Subset of qubits to be measured

Complete set of edges

If the set of edges M, of the graph associated with the resource state $|\psi_G\rangle$, are connected, and E/M is connected. Then this probability can be computed in $O(n^3)$ time (Bravyi & Raussendorf, 2007).

MBQC with surface code states (Qubit-by-qubit)

For the qubit-by qubit technique, the probabilities of each qubit are associated with:

$$|\langle\Phi|\rho_M|\Phi\rangle|^2 \text{ , with } |\Phi\rangle = \otimes_{j\in M}|\phi_j\rangle$$

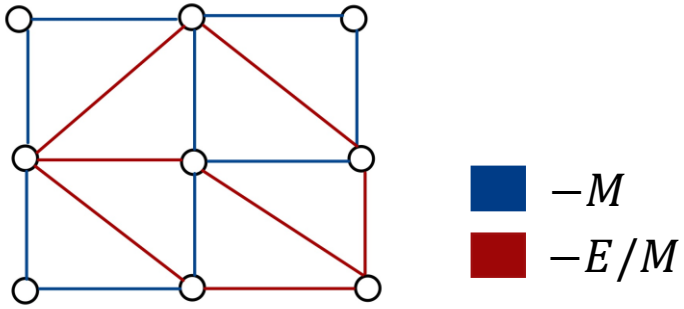
$$\text{, and } \rho_M = Tr_{j\notin M}|\psi_G\rangle\langle\psi_G|$$

Subset of qubits to be measured

Complete set of edges

If the set of edges M, of the graph associated with the resource state $|\psi_G\rangle$, are connected, and E/M is connected. Then this probability can be computed in $O(n^3)$ time (Bravyi & Raussendorf, 2007).

Example:



This edge division represents an efficiently computable overlap

MBQC with surface code states (Qubit-by-qubit)

Although some measurement patterns are efficient to simulate in the qubit-by-qubit approach, it was proven in (Bravyi et al., 2021) that there are some instances that computing $|\langle\Phi|\rho_M|\Phi\rangle|^2$ can be $\#P\text{-}hard$.

MBQC with surface code states (Qubit-by-qubit)

Although some measurement patterns are efficient to simulate in the qubit-by-qubit approach, it was proven in (Bravyi et al., 2021) that there are some instances that computing $|\langle\Phi|\rho_M|\Phi\rangle|^2$ can be $\#P\text{-}hard$.

First, the authors reference a previous work proving that counting perfect matchings in a 3-regular graph is $\#P\text{-}hard$

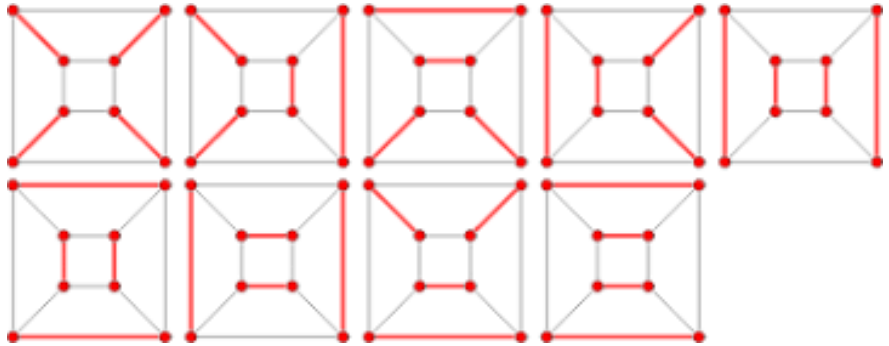
Theorem 1 (Dagum and Luby [49]). *Exact counting of perfect matchings in a 3-regular graph is $\#P$-hard.*

MBQC with surface code states (Qubit-by-qubit)

Although some measurement patterns are efficient to simulate in the qubit-by-qubit approach, it was proven in (Bravyi et al., 2021) that there are some instances that computing $|\langle\Phi|\rho_M|\Phi\rangle|^2$ can be $\#P\text{-}hard$.

First, the authors reference a previous work proving that counting perfect matchings in a 3-regular graph is $\#P\text{-}hard$

Theorem 1 (Dagum and Luby [49]). *Exact counting of perfect matchings in a 3-regular graph is $\#P$-hard.*



Wolfram MathWorld

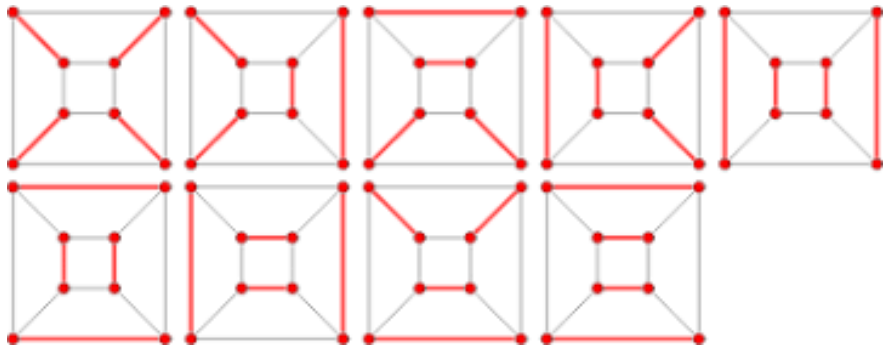
$Perfect\ matchings$

MBQC with surface code states (Qubit-by-qubit)

Although some measurement patterns are efficient to simulate in the qubit-by-qubit approach, it was proven in (Bravyi et al., 2021) that there are some instances that computing $|\langle\Phi|\rho_M|\Phi\rangle|^2$ can be $\#P\text{-}hard$.

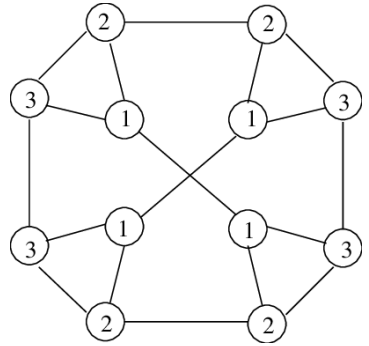
First, the authors reference a previous work proving that counting perfect matchings in a 3-regular graph is $\#P\text{-}hard$

Theorem 1 (Dagum and Luby [49]). *Exact counting of perfect matchings in a 3-regular graph is #P-hard.*



Wolfram MathWorld

$Perfect\ matchings$



$3\text{-}regular\ graph$

(Heggernes & Telle, 2001)

MBQC with surface code states (Qubit-by-qubit)

Then they prove that there is polynomial time algorithm that translates some 3-regular graph to a possible $|\langle\Phi|\rho_M|\Phi\rangle|^2$ instance.

Theorem 2. *There is a polynomial time algorithm that takes as input a 3-regular graph G' and outputs an instance (G, M, Φ) of the Surface Code Marginal problem and a real number C such that the number of perfect matchings in G' coincides with $C\mu(G, M, \Phi)$. The size of G is at most polynomial in the size of G'.*

(Bravyi et al., 2021)

MBQC with surface code states (Qubit-by-qubit)

Then they prove that there is polynomial time algorithm that translates some 3-regular graph to a possible $|\langle\Phi|\rho_M|\Phi\rangle|^2$ instance.

Theorem 2. *There is a polynomial time algorithm that takes as input a 3-regular graph G' and outputs an instance (G, M, Φ) of the Surface Code Marginal problem and a real number C such that the number of perfect matchings in G' coincides with $C\mu(G, M, \Phi)$. The size of G is at most polynomial in the size of G'.*

(Bravyi et al., 2021)

This implies that those instances are at least as difficult as the graph problem.

$$Computing\ |\langle\Phi|\rho_M|\Phi\rangle|^2\ can\ be\ \#P\text{-}hard$$

MBQC with surface code states

Resource/Measurements	Gate-by-gate	Qubit-by-qubit
Surface code state with measurement restrictions	$O(tn^4)$ *	$O(tn^4)$ **

* (Bravyi et al., 2021)

** (Bravyi & Raussendorf, 2007)

MBQC with surface code states

Resource/Measurements	Gate-by-gate	Qubit-by-qubit
Surface code state with measurement restrictions	$O(tn^4)$ *	$O(tn^4)$ **
Surface code state	$O(tn^4)$ *	#P-hard *

* (Bravyi et al., 2021)

** (Bravyi & Raussendorf, 2007)

MBQC with surface code states

Resource/Measurements	Gate-by-gate	Qubit-by-qubit
Surface code state with measurement restrictions	$O(tn^4)$ *	$O(tn^4)$ **
Surface code state	$O(tn^4)$ *	#P-hard *
2D cluster states	#P-hard	#P-hard

* (Bravyi et al., 2021)

** (Bravyi & Raussendorf, 2007)

Conclusions

- The Gate-by-gate approach takes advantage of the quantum circuit creating the quantum state intended to be measured.

Conclusions

- The Gate-by-gate approach takes advantage of the quantum circuit creating the quantum state intended to be measured.
- This approach can be exponentially more efficient in memory restricted simulation processes.

Conclusions

- The Gate-by-gate approach takes advantage of the quantum circuit creating the quantum state intended to be measured.
- This approach can be exponentially more efficient in memory restricted simulation processes.
- Applied to the MBQC protocol, it can simulate in polynomial time any algorithm that uses as a resource a surface code state with a planar graph.

Conclusions

- The Gate-by-gate approach takes advantage of the quantum circuit creating the quantum state intended to be measured.
- This approach can be exponentially more efficient in memory restricted simulation processes.
- Applied to the MBQC protocol, it can simulate in polynomial time any algorithm that uses as a resource a surface code state with a planar graph.
- The same technique can simulate efficiently specific ground states of Hamiltonians.

References I

Barahona, F. (1982). On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, *15*(10), 3241–3253. https://doi.org/10.1088/0305-4470/15/10/028

Bermejo-Vega, J., Delfosse, N., Browne, D. E., Okay, C., & Raussendorf, R. (2017). Contextuality as a Resource for Models of Quantum Computation with Qubits. *prl*, *119*(12), 120505. https://doi.org/10.1103/PhysRevLett.119.120505

Bravyi, S., Gosset, D., & Liu, Y. (2021). How to simulate quantum measurement without computing marginals. *ArXiv E-Prints*, arXiv:2112.08499.

Bravyi, S., & Raussendorf, R. (2007). Measurement-based quantum computation with the toric code states. *Pra*, *76*(2), 22304. https://doi.org/10.1103/PhysRevA.76.022304

Briegel, H. J., Browne, D. E., Dür, W., Raussendorf, R., & Van den Nest, M. (2009). Measurement-based quantum computation. *Nature Physics*, *5*(1), 19–26. https://doi.org/10.1038/nphys1157

Gross, D., Flammia, S. T., & Eisert, J. (2009). Most Quantum States Are Too Entangled To Be Useful As Computational Resources. *Phys. Rev. Lett.*, *102*(19), 190501. https://doi.org/10.1103/PhysRevLett.102.190501

References II

Grover, L. K. (1998). The advantages of superposition. *Science*, *280*, 228. https://link.gale.com/apps/doc/A20520603/AONE?u=anon~b09b35ea&sid=googleScholar&xid=8bbec3a9

Jozsa, Richard; Linden, N. (2003). On the role of entanglement in quantum-computational speed-up. *Proc.R.Soc.Lond.A.*, *459*.

L, Ob. J. (2007). Optical Quantum Computing. *Science*, *318*(5856), 1567–1570. https://doi.org/10.1126/science.1142892

Malgieri, M., Onorato, P., & Ambrosis, A. De. (2014). Teaching quantum physics by the sum over paths approach and {GeoGebra} simulations. *European Journal of Physics*, *35*(5), 55024. https://doi.org/10.1088/0143-0807/35/5/055024

Raghunandan, M., Wolf, F., Ospelkaus, C., Schmidt, P. O., & Weimer, H. (2020). Initialization of quantum simulators by sympathetic cooling. *Science Advances*, *6*(10), eaaw9268. https://doi.org/10.1126/sciadv.aaw9268

van den Nest, M., Miyake, A., Dür, W., & Briegel, H. J. (2006). Universal Resources for Measurement-Based Quantum Computation. *prl*, *97*(15), 150504. https://doi.org/10.1103/PhysRevLett.97.150504

Wald, S. (2017). *Thermalisation and Relaxation of Quantum Systems*. https://doi.org/10.13140/RG.2.2.25169.63842