24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Quantum Weighted Model Counting

Fabrizio Riguzzi[1]

Abstract. In Weighted Model Counting (WMC) we assign weights to Boolean literals and we want to compute the sum of the weights of the models of a Boolean function where the weight of a model is the product of the weights of its literals.

WMC was shown to be particularly effective for performing inference in graphical models, with a complexity of $O(n2^w)$ where n is the number of variables and w is the treewidth.

In this paper, we propose a quantum algorithm for performing WMC, Quantum WMC (QWMC), that modifies the quantum model counting algorithm to take into account the weights. In turn, the model counting algorithm uses the algorithms of quantum search, phase estimation and Fourier transform.

In the black box model of computation, where we can only query an oracle for evaluating the Boolean function given an assignment, QWMC solves the problem approximately with a complexity of $\Theta(2^{\frac{n}{2}})$ oracle calls while classically the best complexity is $\Theta(2^n)$, thus achieving a quadratic speedup.

1 Introduction

Weighted Model Counting (WMC) is the problem of computing the sum of the weights of the models of a propositional formula, where the weight of a model is given by the product of the weights of the literals in it. WMC has been proved effective for performing inference in graphical models [5, 21]. While other graphical model inference algorithms [17, 26, 9, 8] take time $\Theta(n2^w)$ where n is the number of variables and w is the treewidth of the network, WMC takes time $O(n2^w)$, i.e., exponential in the treewidth in the worst case [5]. WMC does so by exploiting structure in the graphical model in the form of context-specific independence and determinism.

In this paper we propose to perform WMC using a quantum computer, i.e., Quantum WMC (QWMC). Quantum computing [19] is the use of quantum mechanics to perform computation. Various algorithms have been proposed for quantum computers that improve over their classical counterpart, the most prominent are: Shor's algorithm [23], that factorizes integers in polynomial time while no classical polynomial algorithm is known, and quantum search, that has a quadratic speedup over classical search [14, 15, 16].

To perform QWMC, we use various quantum algorithms. In particular, we adapt the method of quantum model counting [2, 3] to take into account weights. Quantum model counting in turn is based on quantum search using Grover's algorithm [14, 15, 16] and on quantum phase estimation [6], the latter using the quantum Fourier transform [7].

Here we consider the problem of WMC under a black box computation model where we don't know anything about the propositional formula, we only have the possibility of querying an oracle giving the value of the formula for an assignment of the propositional variables, and we consider the complexity in terms of oracle calls. In this computation model, QWMC solves the problem approximately with a complexity of $\Theta(2^{\frac{n}{2}})$ while classically the best complexity is $\Theta(2^n)$, thus achieving a quadratic speedup.

QWMC may be useful for models with high treewidth: supposing the cost of implementing the oracle is linear in n, if the treewidth is larger than half the number of variables, then QWMC performs better than other inference algorithms.

QWMC can also be used as a subroutine for probabilistic inference system over graphical models. For example, in the junction tree algorithm [22, 17], it can be used after the probabilities are propagated in the tree to compute the marginals of the variables in tree nodes.

The paper is organized as follows. Section 2 presents the WMC problem. Section 3 briefly introduces quantum computing. Then sections 4, 5, 6 and 3 describe the quantum Fourier transform, quantum phase estimation, quantum search and quantum counting algorithms respectively. Section 8 is the main contribution of this paper and presents the quantum weighted model counting algorithm. Section 9 compares the complexity of the algorithm to the one of classical algorithms. Section 10 discusses related work and Section 11 concludes the paper.

2 Weighted Model Counting

Propositional satisfiability (SAT) is the problem of deciding whether a logical formula over Boolean variables evaluates to true for some truth value assignment of the Boolean variables. If an assignment M makes formula ϕ true we write $M \models \phi$. Model counting or #SAT [12] aims at computing the number of satisfying assignments of a propositional sentence.

Weighted model counting (WMC) [5] generalizes model counting by giving each assignment a weight and aiming at computing the sum of the weights of all satisfying assignments.

Definition 1 *Given a formula ϕ in propositional logic over literals L (Boolean variables or their negation), and a weight function $w : L \to R^{\geq 0}$, the weighted model count (WMC) is defined as:*

$$WMC(\phi, w) = \sum_{M \models \phi} weight(M, w)$$

where $weight(M, w) = \prod_{l \in M} w(l)$

Example 1 *Let us consider an example inspired by the sprinkler problem of [20]: we have three Boolean variable, s, r, w representing respectively propositions "the sprinkler was on", 'it rained last night" and "the grass is wet". We know that if the sprinkler was on*

[1] Department of Mathematics and Computer Science, University of Ferrara Via Saragat 1, 44122, Ferrara Italy, Gruppo Nazionale per il Calcolo Scientifico, Istituto Nazionale di Alta Matematica "Francesco Severi", P.le Aldo Moro 5, 00185, Roma Italy, email: fabrizio.riguzzi@unife.it

the grass is wet ($s \rightarrow w$), if it rained last night the grass is wet ($r \rightarrow w$) and that the the sprinkler being on and rain last night cannot be true at the same time ($s, r \rightarrow$). Transforming the formula into conjunctive normal formal we obtain the formula

$$\phi = (\neg s \vee w) \wedge (\neg r \vee w) \wedge (\neg s \vee \neg r)$$

Suppose the weights of literals are $w(s) = 0.3$, $w(\neg s) = 0.7$, $w(r) = 0.2$, $w(\neg r) = 0.8$, $w(w) = 0.5$ and $w(\neg w) = 0.5$, Table 1 shows the worlds of ϕ together with the weight of each world. The WMC of ϕ is thus $WMC(\phi, w) = 0.28+0.28+0.07+0.12 = 0.75$

s	r	w	ϕ	W
0	0	0	1	$0.7 \cdot 0.8 \cdot 0.5 = 0.28$
0	0	1	1	$0.7 \cdot 0.8 \cdot 0.5 = 0.28$
0	1	0	0	$0.7 \cdot 0.2 \cdot 0.5 = 0.07$
0	1	1	1	$0.7 \cdot 0.2 \cdot 0.5 = 0.07$
1	0	0	0	$0.3 \cdot 0.8 \cdot 0.5 = 0.12$
1	0	1	1	$0.3 \cdot 0.8 \cdot 0.5 = 0.12$
1	1	0	0	$0.3 \cdot 0.2 \cdot 0.5 = 0.03$
1	1	1	0	$0.3 \cdot 0.2 \cdot 0.5 = 0.03$

Table 1. Worlds for formula ϕ of Example 1.

3 Quantum Computing

Here we provide a brief introduction to quantum computing following [19]. As the bit is at the basis of classical computing, the *quantum bit* or *qubit* is at the basis of quantum computing. A qubit is a mathematical object that can have various physical implementations. Mathematically it is a unit vector in the $\mathbf{C}^2$ space where $\mathbf{C}$ is the set of complex numbers. A bit can be in one of two states, similarly a qubit has a state which is its vector in $\mathbf{C}^2$. Usually qubits are represented using the Dirac notation where $|\psi\rangle$ is a two dimensional column vector representing the state of a qubit while $\langle\psi|$ is a two dimensional row vector also representing the state. The special states $|0\rangle$ and $|1\rangle$ are identified: they are called *computational basis states* and form the orthonormal basis

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

for $\mathbf{C}^2$. Any qubit state $|\psi\rangle$ can thus be expressed as a linear combination of the computational basis states:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

where α and β are complex number such that $|\alpha|^2 + |\beta|^2 = 1$. In this case we say that $|\psi\rangle$ is in a superposition of states $|0\rangle$ and $|1\rangle$.

In this paper we follow the quantum circuit model of computation where each qubit corresponds to a wire and quantum gates are applied to sets of wires.

Quantum gates are represented by matrices with complex elements. The *adjoint* or *Hermitian conjugate* of a matrix M, denoted by $M^\dagger$, is the conjugate and transpose matrix $M^\dagger = (M^*)^T$. A matrix is *unitary* if $M^\dagger M = I$. Quantum gates are represented by unitary matrices. The simplest gates are those operating on a single qubit and belong to $\mathbf{C}^{2\times 2}$. For example, the counterpart of the NOT Boolean gate for classical bits is X defined as

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

and represented as in Figure 1 top left. Another important gate is the *Hadamard gate* (see Figure 1 top center)

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

A gate that we will use in the following is:

$$R_y(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$$

that applies a rotation of $\theta/2$ radians, with θ user defined, see Figure 1 top right.

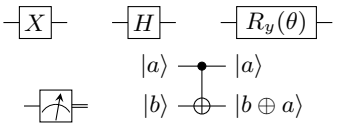

Figure 1. Examples of quantum gates.

Another operation we can apply to a qubit is *measurement*. There are various types of measurements, here we consider only the one with respect to the computational basis that, given a qubit $\alpha\,|0\rangle + \beta\,|1\rangle$, returns a classical bit, namely 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$. Since qubits are unit vectors, this operation is well-defined. Measurement is represented as in Figure 1 bottom left.

Whenever we have more than one bit, we have a composite physical system and the state space expands accordingly: for n qubits, there are 2^n computational basis states, e.g., if $n = 2$ the basis states are $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$ and the state of the qubits can be written as

$$|\psi\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle + \alpha_{10}\,|10\rangle + \alpha_{11}\,|11\rangle$$

Moreover, the state space of a composite physical system is the tensor product of the state spaces of the component physical systems.

The tensor product of two column vectors a and b is ab^T. So the tensor product of two qubits

$$|a\rangle = a_0\,|0\rangle + a_1\,|1\rangle = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$$

$$|b\rangle = b_0\,|0\rangle + b_1\,|1\rangle = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}$$

is

$$|a\rangle \otimes |b\rangle = \begin{bmatrix} a_0 b_0 \\ a_0 b_1 \\ a_1 b_0 \\ a_1 b_1 \end{bmatrix} =$$

$$a_0 b_0\,|00\rangle + a_0 b_1\,|01\rangle + a_1 b_0\,|10\rangle + a_1 b_1\,|11\rangle$$

For two qubits, the most important gate is the *controlled-NOT* or *CNOT* gate that has two inputs, the control and the target qubits, and acts by flipping the target qubit if the control bit is set to 1 and does nothing if the control bit is set to 0. It can also be defined as a gate that operates as $|ab\rangle \rightarrow |a, b \oplus a\rangle$ where $\oplus$ is the XOR operation, see Figure 1 bottom right.

Any multiple qubit logic gate may be composed from CNOT and single qubit gates.

CNOT may be generalized to the case of more than two bits: in this case, the extra qubits act as controls and the target is flipped if all controls are 1. Moreover, given an operator U, it is possible to define a control-U operator defined as $|ab\rangle \rightarrow |a, U^a b\rangle$: if $a = 0$ it does nothing, otherwise it applies operator U to b.

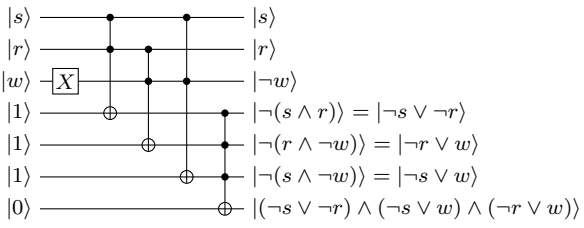

Figure 2. Quantum circuit for computing ϕ

Example 2 *The quantum circuit for computing the value of formula ϕ from Example 1 is shown in Figure 2.*

Quantum circuits should be read from left to right. Each line or wire correspond to a qubit and starts in a computational basis state, usually $|0\rangle$ unless otherwise indicated. The circuit in Figure 2 contains one wire for each Boolean variable of Example 1 plus four other wires that represent the so called *ancilla qubits*. Ancilla qubits are used in order to make the circuit reversible. The bottom ancilla qubit contains the truth value of function ϕ.

4 Quantum Fourier Transform

The discrete Fourier transform computes a vector of complex numbers $y_0, \ldots y_{N-1}$ given a vector of complex numbers $x_0, \ldots, x_{N-1}$ as follows

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi ijk/N}$$

The *quantum Fourier transform* [7] is similar, it takes an orthonormal basis $|0\rangle, \ldots, |N-1\rangle$ and transforms it as:

$$|j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |k\rangle$$

It is a Fourier transform because the action on an arbitrary state is

$$\sum_{j=0}^{N-1} x_j |j\rangle \rightarrow \sum_{k=0}^{N-1} y_k |k\rangle$$

with y_k as in the discrete Fourier transform.

Assuming $N = 2^n$, the quantum Fourier transform can be given a *product representation* [6, 13]:

$$\begin{array}{l} |j_1, \ldots, j_n\rangle \rightarrow \\ \frac{1}{2^{n/2}} \sum_{k=0}^{2^n-1} e^{2\pi ijk/2^n} |k\rangle \\ = \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} e^{2\pi ij\left(\sum_{l=1}^{n} k_l 2^{-l}\right)} |k_1 \ldots k_n\rangle \\ = \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \otimes_{l=1}^{n} e^{2\pi ijk_l 2^{-l}} |k_l\rangle \\ = \frac{1}{2^{n/2}} \otimes_{l=1}^{n} \left[\sum_{k_l=0}^{1} e^{2\pi ijk_l 2^{-l}} |k_l\rangle\right] \\ = \frac{1}{2^{n/2}} \otimes_{l=1}^{n} \left[|0\rangle + e^{2\pi ij2^{-l}} |1\rangle\right] \\ = \frac{\left(|0\rangle + e^{2\pi 0.j_n} |1\rangle\right) \otimes \left(|0\rangle + e^{2\pi 0.j_{n-1}j_n} |1\rangle\right) \otimes \cdots \left(|0\rangle + e^{2\pi 0.j_1 j_2 \cdots j_n} |1\rangle\right)}{2^{n/2}} \end{array} \quad (1)$$

where the state $|j\rangle$ is written using the binary representation $j = j_1 j_2 \ldots j_n$ and $0.j_l j_{l+1} \ldots j_m$ represents the number $j_l/2 + j_{l+1}/4 + \ldots + j_m/2^{m-l+1}$. The quantum Fourier transform requires $\Theta(n^2)$ gates.

5 Quantum Phase Estimation

In the problem of *quantum phase estimation* [6], we are given an operator U and one of its eigenvectors $|u\rangle$ with eigenvalue $e^{2\pi i\varphi}$ and we want to find the value of φ. We assume that that we have black boxes that can prepare the state $|u\rangle$ and perform controlled-U^{2^j} operations for non negative integers j.

Phase estimation uses two registers, one with t qubits initially in state $|0\rangle$ and the other with as many qubits as are necessary to store $|u\rangle$ that is also its initial state.

The first stage of phase estimation is shown in Figure 3. A controlled-U^{2^j} operation on control qubit b and target register in an eigenvector state $|u\rangle$ of U acts as follows. If b is $|0\rangle$, U^{2^j} is not applied and the output is $|0\rangle |u\rangle$. If b is $|1\rangle$, then U^{2^j} is applied to $|u\rangle$. Since $|u\rangle$ is an eigenvector of U, $|u\rangle$ is brought to $e^{2\pi i 2^j \varphi} |u\rangle$ and $|1\rangle |u\rangle$ becomes $e^{2\pi i 2^j \varphi} |1\rangle |u\rangle$.

The result of the controlled-U^{2^j} operation on $(H|0\rangle)|u\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}} |u\rangle$ is

$$\left(\frac{|0\rangle + e^{2\pi i 2^j \varphi} |1\rangle}{\sqrt{2}}\right) |u\rangle$$

Thus the final state of the first register after the first phase of phase estimation is

$$\begin{array}{l} \frac{1}{2^{t/2}} \left(|0\rangle + e^{2\pi i 2^{t-1} \varphi} |1\rangle\right) \otimes \left(|0\rangle + e^{2\pi i 2^{t-2} \varphi} |1\rangle\right) \cdots \left(|0\rangle + e^{2\pi i 2^0 \varphi} |1\rangle\right) \\ = \frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} e^{2\pi i \varphi k} |k\rangle \end{array} \quad (2)$$

If the phase can be represented with exactly t bits as $\varphi = 0.\varphi_1 \ldots \varphi_t$, Equation (2) can be rewritten as

$$\frac{\left(|0\rangle + e^{2\pi 0.\varphi_t} |1\rangle\right) \otimes \left(|0\rangle + e^{2\pi 0.\varphi_{t-1}\varphi_t} |1\rangle\right) \otimes \cdots \left(|0\rangle + e^{2\pi 0.\varphi_1 \cdots \varphi_t} |1\rangle\right)}{2^{n/2}}$$

This form is exactly the same as that of Equation (1) so, if we apply the inverse of the Fourier transform, we obtain $|\varphi_1, \ldots, \varphi_t\rangle$. The inverse of an operator is its adjoint so the overall phase estimation circuit is shown in Figure 4.

If φ cannot be represented exactly with t bits, the algorithm provides approximation guarantees: if we want to approximate φ to m bits with probability of success at least $1 - \epsilon$ we must choose $t = m + \lceil \log_2 \left(2 + \frac{1}{2\epsilon}\right) \rceil$ [19].

6 Quantum Search

The problem of *quantum search* is, given a Boolean function $\phi : \{0,1\}^n \rightarrow \{0,1\}$, return a configuration of bits x such that $\phi(x) = 1$ [14, 15, 16]. We assume we have a black box that evaluates ϕ, we call it an *oracle* O, that is such that

$$|x\rangle \rightarrow^O (-1)^{\phi(x)} |x\rangle$$

i.e., the oracle marks solutions to the search problems by changing their sign. The oracle may use extra ancilla bits to do so. For the case of the function of Example 1, the oracle will use a circuit such as the one of Figure 2 in its internals. Figure 5 shows the circuit performing quantum search operating on an n-qubit register r and the oracle workspace o. All qubits of register r start in state $|0\rangle$. The circuit includes a gate G that is called the *Grover operator* and is implemented as show in Figure 6. The first gate of the search circuit applies the H gate to each qubit in register r. Since all qubits in register r start as $|0\rangle$ and the effect of H is to bring $|0\rangle$ to the state $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$, then register r is brought to

$$\begin{array}{l} |\psi\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \cdots \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \\ = \frac{|00\rangle+|01\rangle+|10\rangle+|11\rangle}{\sqrt{2^2}} \cdots \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \\ = \frac{|000\rangle+|001\rangle+|010\rangle+|011\rangle+|100\rangle+|101\rangle+|110\rangle+|111\rangle}{\sqrt{2^3}} \cdots \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} \\ = \frac{1}{N^{1/2}} \sum_{x=0}^{N-1} |x\rangle \end{array}$$

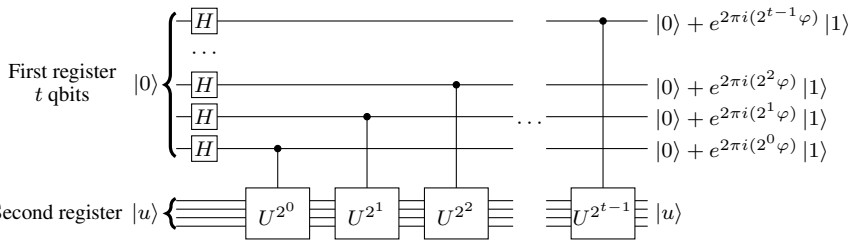

Figure 3. First stage of phase estimation. On the right we have omitted normalization factors of $\frac{1}{\sqrt{2}}$.

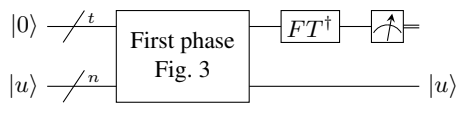

Figure 4. The complete phase estimation circuit.

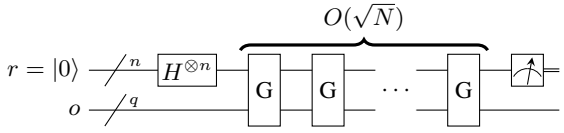

Figure 5. Quantum search algorithm.

where $N = 2^n$. This state is also called the *uniform superposition state*.

The Grover operator can be written as

$$G = (2\,|\psi\rangle\,\langle\psi| - I)O$$

We now show that the Grover operator is a rotation. Consider the two states

$$|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum_{x:\phi(x)=0} |x\rangle$$

$$|\beta\rangle = \frac{1}{\sqrt{M}} \sum_{x:\phi(x)=1} |x\rangle$$

where M is the number of solutions to $\phi(x) = 1$. These two states are orthonormal. The uniform superposition state $|\psi\rangle$ can be written as a linear combination of $|\alpha\rangle$ and $|\beta\rangle$:

$$|\psi\rangle = \sqrt{\frac{N-M}{N}}\,|\alpha\rangle + \sqrt{\frac{M}{N}}\,|\beta\rangle$$

so $|\psi\rangle$ belongs to the plane defined by $|\alpha\rangle$ and $|\beta\rangle$. In this plane, the effect of the oracle operation O is to perform a reflection about the vector α because $O(|\alpha\rangle + |\beta\rangle) = |\alpha\rangle - |\beta\rangle$, see Figure 7.

The other component of Grover operator, $2\,|\psi\rangle\,\langle\psi| - I$, also performs a reflection in the plane defined by $|\alpha\rangle$ and $|\beta\rangle$, about the vector $|\psi\rangle$. The overall effect is that of a rotation [1]. Define $\cos\theta/2 = \sqrt{(N-M)/N}$, then $|\psi\rangle = \cos\theta/2\,|\alpha\rangle + \sin\theta/2\,|\beta\rangle$.

From Figure 7 we can see that the rotation applied by G is exactly θ so

$$G\,|\psi\rangle = \cos\frac{3\theta}{2}\,|\alpha\rangle + \sin\frac{3\theta}{2}\,|\beta\rangle$$

Repeated applications of G take the state to

$$G^k\,|\psi\rangle = \cos\left(\frac{2k+1}{2}\theta\right)|\alpha\rangle + \sin\left(\frac{2k+1}{2}\theta\right)|\beta\rangle\,.$$

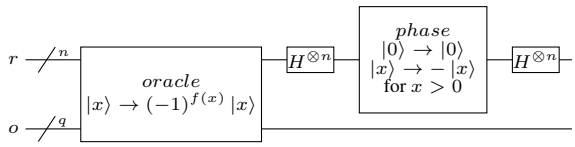

Figure 6. Grover operator.

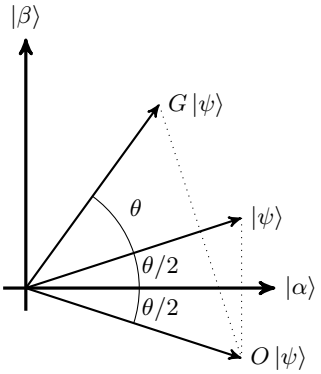

Figure 7. Visualization of the effect of Grover operator.

These rotations bring $|\psi\rangle$ closer and closer to $|\beta\rangle$. If we perform the right number of rotations, an observation in the computational basis produces with high probability one of the outcomes superposed in $|\beta\rangle$, i.e., a solution to the search problem. It turns out that the number of applications of G (and thus of oracle calls) required to maximise the probability of measuring one of the solutions to the search problem is $O(\sqrt{N/M})$, while classically by treating ϕ as a black box the number of oracle calls would be $O(N/M)$.

The algorithm works if $M \leq N/2$. If this is not true, it is enough to consider an extra qubit e, defining a new function $\phi'(x)$ that is true only if e is true, i.e., $\phi'(x) = \phi(x) \wedge e$. This leaves M unchanged but multiplies N by 2.

7 Quantum Counting

With quantum counting we want to count the number of solutions to the equation $\phi(x) = 1$ where ϕ is a Boolean function as above. In the notation of the previous section, it means computing M.

Suppose $|a\rangle$ and $|b\rangle$ are the two eigenvectors of the Grover operator G in the space spanned by $|\alpha\rangle$ and $|\beta\rangle$. Since G is a rotation of angle θ in such a space, the eigenvalues of $|a\rangle$ and $|b\rangle$ are $e^{i\theta}$ and $e^{i(2\pi-\theta)}$. If we know θ, we can compute M from $\sin^2(\theta/2) = M/2N$ (supposing the oracle has been augmented with an extra qubit). Since $\sin(\theta/2) = \sin(\pi - \theta/2)$, it does not matter which eigenvalue is estimated.

So quantum counting is performed by using quantum phase esti-

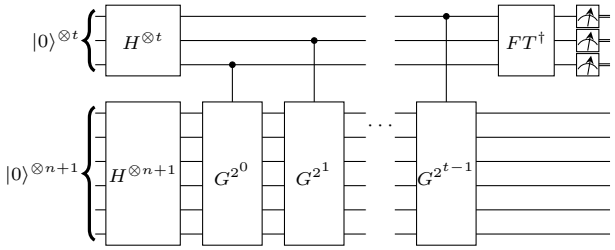

Figure 8. Circuit for quantum counting.

mation to compute the eigenvalues of the Grover operator G. The circuit for quantum counting is shown in Figure 8 [2, 3].

The upper register in Figure 8 has t qubits while the lower register $n+1$. θ is estimated to m bits of accuray with probability at least $1-\epsilon$ if $t = m + \lceil \log_2(2 + 1/2\epsilon) \rceil$. The error on the estimate of the count M is given by [19]:

$$\frac{|\Delta M|}{2N} = \left|\sin^2\left(\frac{\theta+\Delta\theta}{2}\right) - \sin^2\left(\frac{\theta}{2}\right)\right| = \left(\sin\left(\frac{\theta+\Delta\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\right)\left|\sin\left(\frac{\theta+\Delta\theta}{2}\right) - \sin\left(\frac{\theta}{2}\right)\right|$$

Since $|\sin((\theta+\Delta\theta)/2) - \sin(\theta/2)| \leq |\Delta\theta|/2$ and $|\sin((\theta+\Delta\theta)/2)| < \sin(\theta/2) + |\Delta\theta|/2$ from calculus and trigonometry respectively, we get

$$\frac{|\Delta M|}{2N} < \left(2\sin\left(\frac{\theta}{2}\right) + \frac{|\Delta\theta|}{2}\right)\frac{|\Delta\theta|}{2}$$

Using $\sin^2(\theta/2) = M/2N$ and $|\Delta\theta| \leq 2^{-m}$ we obtain

$$|\Delta M| < \left(\sqrt{2MN} + \frac{N}{2^{m+1}}\right)2^{-m}$$

Consider this case: let $m = \lceil n/2 \rceil + 2$ and $\epsilon = 1/12$. Then $t = \lceil n/2 \rceil + 5$. The number of applications of the Grover operator is $\Theta(\sqrt{N})$ and so is the number of oracle calls. The error is $|\Delta M| < \sqrt{M/8} + 1/32 = O(\sqrt{M})$.

8 Quantum Weighted Model Counting

For the moment suppose that the literal weights sum to 1, i.e., that $w(x_i) + w(\neg x_i) = 1$ for all bits x_i.

The circuit for performing quantum weighted model counting is shown in Figure 9 and differs from the one in Figure 8 because the Hadamard operations applied to the lower register are replaced by rotations $R_y(\theta_i)$ where i is the qubit index except for the extra qubit for which the Hadamard operator is kept. θ_i is computed as

$$\theta_i = 2\arccos\sqrt{1-w_i}$$

where $w_i = w(x_i)$. So

$$\cos\theta_i/2 = \cos\arccos\sqrt{1-w_i} = \sqrt{1-w_i}$$

and

$$\sin\theta_i/2 = \sqrt{1-(\cos\theta_i/2)^2} = \sqrt{w_i}$$

The effect of the rotation on the ith bit is

$$R_y(\theta_i)|0\rangle = \begin{bmatrix} \cos\frac{\theta_i}{2} & -\sin\frac{\theta_i}{2} \\ \sin\frac{\theta_i}{2} & \cos\frac{\theta_i}{2} \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \cos\frac{\theta_i}{2} \\ \sin\frac{\theta_i}{2} \end{bmatrix} = \begin{bmatrix} \sqrt{1-w_i} \\ \sqrt{w_i} \end{bmatrix} = \sqrt{1-w_i}\,|0\rangle + \sqrt{w_i}\,|1\rangle$$

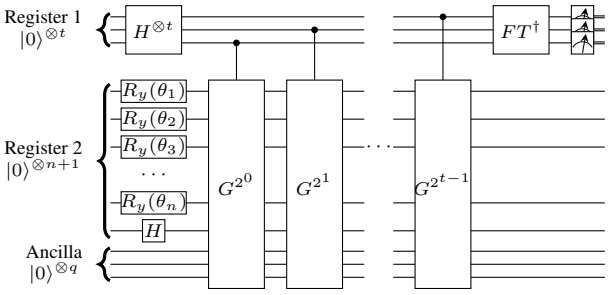

Figure 9. Circuit for quantum weighted model counting.

Therefore the rotations prepare the state

$$\psi = \bigotimes_{i=1}^{n}(\sqrt{1-w_i}\,|0\rangle + \sqrt{w_i}\,|1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \sum_{b_{n+1}b_n\ldots b_1=0}^{2^{n+1}-1}\sqrt{\frac{w'_n\ldots w'_1}{2}}\,|b_{n+1}b_n\ldots b_1\rangle$$

where w'_i is

$$w'_i = \begin{cases} w_i & \text{if } b_i = 1 \\ 1-w_i & \text{if } b_i = 0 \end{cases}$$

Define $W_{b_n b_{n-1}\ldots b_1}$ as $w'_n w'_{n-1}\ldots w'_1$ and normalized states

$$|\alpha\rangle = \frac{1}{\sqrt{\frac{\sum_{x;\phi(x)=0} W_x}{2}}}\sum_{x;\phi(x)=0}\sqrt{\frac{W_x}{2}}\,|x\rangle$$

$$|\beta\rangle = \frac{1}{\sqrt{\frac{\sum_{x;\phi(x)=1} W_x}{2}}}\sum_{x;\phi(x)=1}\sqrt{\frac{W_x}{2}}\,|x\rangle,$$

then $|\psi\rangle$ can be expressed as

$$|\psi\rangle = \left(\sqrt{\frac{\sum_{x;\phi(x)=0} W_x}{2}}\right)|\alpha\rangle + \left(\sqrt{\frac{\sum_{x;\phi(x)=1} W_x}{2}}\right)|\beta\rangle$$

so the initial state of the quantum computer is in the space spanned by $|\alpha\rangle$ and $|\beta\rangle$

Let $\cos\theta/2 = \sqrt{\frac{\sum_{x;\phi(x)=0} W_x}{2}}$ and $\sin\theta/2 = \sqrt{\frac{\sum_{x;\phi(x)=1} W_x}{2}}$ so that

$$|\psi\rangle = \cos\theta/2\,|\alpha\rangle + \sin\theta/2\,|\beta\rangle$$

From this point we can repeat the reasoning used for quantum counting: the application of the Grover operator rotates $|\psi\rangle$ in the space spanned by $|\alpha\rangle$ and $|\beta\rangle$ by angle θ and $e^{i\theta}$ and $e^{i(2\pi-\theta)}$ are the eigenvalues of G. θ can be found by quantum phase estimation. From $\sin^2(\theta/2) = \frac{\sum_{x;\phi(x)=1} W_x}{2}$ we obtain

$$WMC(\phi, w) = \sum_{x:\phi(x)=1} W_x = 2\sin^2(\theta/2)$$

If the literal weights do not sum to 1, i.e., $w(x_i) + w(\neg x_i) \neq 1$, consider the normalized weights, i.e., the new weights $\hat{w}(x_i) = \frac{w(x_i)}{w(x_i)+w(\neg x_i)}$ and $\hat{w}(\neg x_i) = \frac{w(\neg x_i)}{w(x_i)+w(\neg x_i)}$. Let V_i be $w(x_i) + w(\neg x_i)$ for $i = 1, \ldots, n$. Then we perform QWMC with $\hat{w}$ replacing w. We get a normalized WMC $\widehat{WMC}(\phi, w)$

$$\widehat{WMC}(\phi, w) = \sum_{x:\phi(x)=1} \hat{W}_x$$

where $\hat{W}_{b_n b_{n-1} \dots b_1}$ is $\hat{w}'_n \hat{w}'_{n-1} \dots \hat{w}'_1$ and

$$\hat{w}'_i = \begin{cases} \hat{w}(x_i) & \text{if } b_i = 1 \\ 1 - \hat{w}(x_i) & \text{if } b_i = 0 \end{cases}$$

Then

$$\begin{aligned} &\widehat{WMC}(\phi, w) = \\ &\sum_{x:\phi(x)=1} \hat{W}_x = \\ &\sum_{b_n \dots b_1 : \phi(b_n \dots b_1)=1} \hat{W}_{b_n \dots b_1} = \\ &\sum_{b_n \dots b_1 : \phi(b_n \dots b_1)=1} \hat{w}'_n \dots \hat{w}'_1 = \\ &\sum_{b_n \dots b_1 : \phi(b_n \dots b_1)=1} \frac{w(b_n)}{V_n} \dots \frac{w(b_1)}{V_1} = \\ &\sum_{b_n \dots b_1 : \phi(b_n \dots b_1)=1} \frac{1}{\prod_{i=1}^n V_i} w(b_n) \dots w(b_1) = \\ &\frac{1}{\prod_{i=1}^n V_i} \sum_{b_n \dots b_1 : \phi(b_n \dots b_1)=1} w(b_n) \dots w(b_1) = \\ &\frac{1}{\prod_{i=1}^n V_i} WMC(\phi, w) \end{aligned}$$

where $w(b_i) = w(x_i)$ if $b_i = 1$ and $w(b_i) = w(\neg x_i)$ if $b_i = 0$. So if we multiply $\widehat{WMC}(\phi, w)$ by $\prod_{i=1}^n V_i$ we obtain $WMC(\phi, w)$ also when the weights do not sum to 1.

Let us consider the complexity of the algorithm.

Theorem 1 *QWMC on n bits requires $\Theta(\sqrt{N})$ oracle calls to bound the error to $2^{-\frac{n+1}{2}}$ with probability $1/12$ using $t = \lceil n/2 \rceil + 5$ bits.*

Proof: We can repeat the derivation of the previous section where M is replaced by $N \times \widehat{WMC}(\phi, w)$. We get

$$\frac{|\Delta \widehat{WMC}(\phi, w)|}{2} < \left(2 \sin\left(\frac{\theta}{2}\right) + \frac{|\Delta\theta|}{2} \right) \frac{|\Delta\theta|}{2}$$

Using $\sin^2(\theta/2) = \widehat{WMC}(\phi, w)/2$ and $|\Delta\theta| \leq 2^{-m}$ we obtain

$$|\Delta \widehat{WMC}(\phi, w)| < \left(\sqrt{2\widehat{WMC}(\phi, w)} + 2^{-m-1} \right) 2^{-m}$$

Since $\widehat{WMC}(\phi, w) \leq 1$ we have

$$|\Delta \widehat{WMC}(\phi, w)| < \left(\sqrt{2} + 2^{-m-1} \right) 2^{-m} < 2^{-m+\frac{1}{2}} + 2^{-2m-1}$$

If we choose $m = \lceil n/2 \rceil + 2$ and $\epsilon = 1/12$, then $t = \lceil n/2 \rceil + 5$ and the algorithm requires $\Theta(\sqrt{N})$ oracle calls. The error becomes (for n even, for n odd the result is similar):

$$\begin{aligned} &|\Delta \widehat{WMC}(\phi, w)| < 2^{-\frac{n}{2}-2+\frac{1}{2}} + 2^{-n-5} < \\ &2^{-\frac{n}{2}-\frac{3}{2}} + 2^{-\frac{n}{2}-\frac{3}{2}} < 2^{-\frac{n}{2}-\frac{1}{2}} \end{aligned}$$

so the error is bounded by $2^{-\frac{n+1}{2}}$ □

9 Complexity of Classical Algorithms

Let us now discuss the advantages fo QWMC with respect to WMC. We consider a black box model of computation [19], where the only knowledge we have on the Boolean function ϕ is the possibility of evaluating it given an assignment of the Boolean variables, i.e., we have an oracle that answers queries over ϕ. We want to know what is the minimum number of evaluations that are needed to solve counting problems.

Consider first an unweighted counting problem. A classical algorithm for probabilistically solving it proceeds by taking k samples uniformly from the search space. This can be performed by sampling each Boolean variable uniformly and combining the bit samples obtaining an assignment sample. For each assignment sample, we query the oracle and we obtain a value X_i with $i = 1, \dots, k$, where X_i is 1 if ϕ evaluates to true for the sample and X_i is 0 if ϕ evaluates to false. Then we can estimate the count as

$$S = \frac{N}{k} \times \sum_{i=1}^{k} X_k = \frac{N\overline{X}}{k}$$

where $\overline{X} = \sum_{i=1}^{k} X_k$. Variable $\overline{X} = Sk/N$ is binomially distributed with k the number of trials and probability of success M/N where M is the model count of ϕ. Therefore the mean of $\overline{X}$ is kM/N and the mean of S is $(N/k)k(M/N) = M$, so S is an unbiased estimate of M.

Theorem 2 *[19, Exercise 6.13]. The complexity of the classical algorithm for estimating M with a probability of at least $3/4$ within an accuracy of $\sqrt{M}$ is $\Omega(N)$ oracle calls.*

Proof: We can use the normal approximation of the binomial proportion confidence interval according to which the true success probability of the binomial variable lies in the interval

$$\hat{p} \pm z \sqrt{\frac{\hat{p}\,(1-\hat{p})}{k}}$$

where $\hat{p}$ is the estimated probability and z is the quantile of a standard normal distribution that depends on the confidence (in our case the confidence is 75% and so $z = 0.6744898$). The size of the interval where the true probability lies is therefore

$$2z \sqrt{\frac{S/N(1-S/N)}{k}}$$

and the the size of the interval of the number of solutions is

$$2zN \sqrt{\frac{S/N(1-S/N)}{k}}.$$

We replace the estimated probability with the true one to get a better estimate:

$$2zN \sqrt{\frac{M/N(1-M/N)}{k}}.$$

We want this to be smaller than $\sqrt{M}$ so

$$\begin{aligned} \sqrt{M} &\geq 2zN \sqrt{\frac{M/N(1-M/N)}{k}} \\ M &\geq 4z^2N^2 \frac{M/N(1-M/N)}{k} \\ k &\geq 4z^2N^2 \frac{M/N(1-M/N)}{M} \\ k &\geq 4z^2N(1-M/N) \end{aligned}$$

so $k = \Omega(N)$ □

It turns out that this is the best bound, in the sense that any classical counting algorithm with a probability at least 3/4 for estimating M correctly to within an accuracy $c\sqrt{M}$ for some constant c must make $\Omega(N)$ oracle calls [19, Exercise 6.14], [18, Table 2.5]. So quantum computing gives us a quadratic speedup.

For QWMC, consider the following classical algorithm: take k assignment samples by sampling each bit according to its normalized weight. For each assignment sample, query the oracle obtaining value X_i with $i = 1, \ldots, k$ and estimate the WMC as for the unweighted case: $S = \frac{N}{k}\sum_{i=1}^{k} X_i$ Variable Sk/N is again binomially distributed with k the number of trials and probability of success $\widehat{WMC}(\phi, w)$. In fact, the probability $P(X_i = 1)$ is given by $P(X_i) = \sum_x P(X_i, x) = \sum_x P(X_i|x)P(x)$ where $P(X_i|x)$ is 1 if x is a model of ϕ and 0 otherwise. So

$$\begin{aligned} P(X_i) &= \sum_{x:\phi(x)=1} P(x) = \\ &\sum_{b_n \ldots b_1 : \phi(b_n \ldots b_1)=1} P(b_n \ldots b_1) = \\ &\sum_{b_n \ldots b_1 : \phi(b_n \ldots b_1)=1} P(b_n) \ldots P(b_1) = \\ &\sum_{b_n \ldots b_1 : \phi(b_n \ldots b_1)=1} \prod_{i=1}^{n} \hat{w}_i' = \\ &\sum_{b_n \ldots b_1 : \phi(b_n \ldots b_1)=1} \prod_{i=1}^{n} \frac{w(b_i)}{V_i} = \\ &\frac{WMC(\phi, w)}{\prod_{i=1}^{n} V_i} = \widehat{WMC}(\phi, w) \end{aligned}$$

Theorem 3 *The complexity of the classical algorithm for estimating $\widehat{WMC}(\phi, w)$ with a probability of at least $3/4$ within an accuracy of $2^{-\lceil \frac{n}{2} \rceil}$ is $\Omega(N)$. oracle calls and this is the best bound for a classical algorithm.*

Proof: We can repeat the reasoning performed with counting: the size of the interval where the true value of $\widehat{WMC}(\phi, w)$ lies is

$$2z\sqrt{\frac{S/N(1 - S/N)}{k}}$$

Let us replace S/N by its true value $\widehat{WMC}(\phi, w)$ obtaining

$$2z\sqrt{\frac{\widehat{WMC}(\phi, w)(1 - \widehat{WMC}(\phi, w))}{k}}$$

Suppose we want the error below $2^{-\lceil \frac{n}{2} \rceil}$ so

$$2^{-\lceil \frac{n}{2} \rceil} \geq 2z\sqrt{\frac{\widehat{WMC}(\phi, w)(1 - \widehat{WMC}(\phi, w))}{k}}$$

Squaring both members we get (if n is even, if it is odd the result is similar)

$$2^{-n} \geq 4z^2 \frac{\widehat{WMC}(\phi, w)(1 - \widehat{WMC}(\phi, w))}{k}$$

and

$$k \geq 4z^2 2^n \widehat{WMC}(\phi, w)(1 - \widehat{WMC}(\phi, w))$$

We want the bound to work for all valules of $\widehat{WMC}(\phi, w)$ and $\widehat{WMC}(\phi, w)(1 - \widehat{WMC}(\phi, w)) \leq 1/4$ so we have

$$k \geq z^2 2^n$$

Therefore $k = \Omega(N)$ This is also the best bound for a classical algorithm, as otherwise we could solve model counting with a better bound than $\Omega(N)$ by setting all weights to 0.5 □

Therefore QWMC offers a quadratic speedup over classical computation in the black box model.

10 Related Work

Recently the problem of computing the weighted count of eigenstates of Hamiltonians was tackled by [24]. The authors proposed mixed quantum-Monte Carlo algorithms that repeat several times quantum computations and then use statistics from the results to approximate the weighted count. In particular, they propose applications of Adiabatic quantum optimization (AQO), quantum approximate optimization algorithm (QAOA) and Grover's algorithm to find the eigenstates and provide individual samples. The authors prove (numerically for QAOA) that the number of samples needed to achieve a certain relative error is lower than that of classical optimal Monte Carlo simulation. This work can be used to perform weighted model counting, we differ from it because we rely on quantum phase estimation rather than sampling.

An approximate algorithm for counting was also proposed by [4]: the algorithm uses a logarithmic number of calls to a SAT oracle. We differ from this work because in our case the oracle is the evaluation of the Boolean function which is much cheaper than a SAT call.

Another work on approximate weighted model counting is [11] that exploits a dynamic programming algorithm on tree decomposition and can be parallelized using GPUs. However, the method require the treewidth of the formula to be small, while we have no such requirement.

11 Conclusion

We have proposed an algorithm for performing quantum weighted model counting. The algorithm minimally modifies the quantum counting algorithm by just changing the preparation of the state of the second register. In turn QWMC also uses quantum search, phase estimation and the Fourier transform.

Using the black box model of computation, QWMC makes $\Theta(\sqrt{N})$ oracle calls to return a result whose errors is bounded by $2^{-\frac{n+1}{2}}$ with probability 11/12. By contrast, the best classical algorithm requires $\Theta(N)$ calls to the oracle. Thus QWMC offers a quadratic speedup that may be useful for example for computing marginals for the variables of a tree node in the junction tree algorithm

We have implemented the algorithm in two versions, one for Microsoft's Q# [25] and one for IBM's Qiskit [10], two programming languages for quantum computers. The code is available from `https://bitbucket.org/machinelearningunife/qwmc`. Performing ten QWMC operations on the sprinkler example takes 605 ms on a PC with Intel Core i7 @ 3.20 Ghz using the Q# simulator and 11.99 seconds for the Qiskit version using the `qasm_simulator` on the IBM Q Experience web application.

REFERENCES

[1] Dorit Aharonov, 'Quantum computation', in *Annual Reviews of Computational Physics VI*, 259–346, World Scientific, (1999).
[2] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp, 'Tight bounds on quantum searching', *Fortschritte der Physik: Progress of Physics*, **46**(4-5), 493–505, (1998).
[3] Gilles Brassard, Peter Høyer, and Alain Tapp, 'Quantum counting', in *25th International Colloquium on Automata, Languages and Programming (ICALP 1998)*, eds., Kim Guldstrand Larsen, Sven Skyum, and Glynn Winskel, volume 1443 of *Lecture Notes in Computer Science*, pp. 820–831. Springer, (1998).
[4] Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi, 'Algorithmic improvements in approximate counting for probabilistic inference: From linear to logarithmic SAT calls', in *25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*, ed., Subbarao Kambhampati, pp. 3569–3576. AAAI Press/IJCAI, (2016).
[5] Mark Chavira and Adnan Darwiche, 'On probabilistic inference by weighted model counting', *Artificial Intelligence*, **172**(6-7), 772–799, (2008).
[6] Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca, 'Quantum algorithms revisited', *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, **454**(1969), 339–354, (1998).
[7] Don Coppersmith, 'An approximate fourier transform useful in quantum factoring', *arXiv preprint quant-ph/0201067*, (2002).
[8] Adnan Darwiche, 'Recursive conditioning', *Artificial Intelligence*, **126**(1-2), 5–41, (2001).
[9] Rina Dechter, 'Bucket elimination: A unifying framework for reasoning', *Artificial Intelligence*, **113**(1-2), 41–85, (1999).
[10] Héctor Abraham et al. Qiskit: An open-source framework for quantum computing, 2019.
[11] Johannes Klaus Fichte, Markus Hecher, Stefan Woltran, and Markus Zisser, 'Weighted model counting on the GPU by exploiting small treewidth', in *26th Annual European Symposium on Algorithms (ESA 2018)*, eds., Yossi Azar, Hannah Bast, and Grzegorz Herman, pp. 28:1–28:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, (2018).
[12] Carla P. Gomes, Ashish Sabharwal, and Bart Selman, 'Model counting', in *Handbook of Satisfiability*, eds., Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, volume 185, 633–654, IOS Press, (2009).
[13] Robert B Griffiths and Chi-Sheng Niu, 'Semiclassical fourier transform for quantum computation', *Physical Review Letters*, **76**(17), 3228, (1996).
[14] Lov K. Grover, 'A fast quantum mechanical algorithm for database search', in *28th Annual ACM Symposium on Theory of Computing (STOC 1996)*, pp. 212–219, New York, NY, USA, (1996). ACM Press.
[15] Lov K Grover, 'A fast quantum mechanical algorithm for database search', *arXiv preprint quant-ph/9605043*, (1996).
[16] Lov K Grover, 'Quantum mechanics helps in searching for a needle in a haystack', *Physical review letters*, **79**(2), 325, (1997).
[17] Steffen L Lauritzen and David J Spiegelhalter, 'Local computations with probabilities on graphical structures and their application to expert systems', *Journal of the Royal Statistical Society: Series B (Methodological)*, **50**(2), 157–194, (1988).
[18] Michele Mosca, *Quantum computer algorithms*, Ph.D. dissertation, University of Oxford. 1999., 1999.
[19] M.A. Nielsen and I.L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, Cambridge University Press, 2010.
[20] Judea Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1988.
[21] Tian Sang, Paul Beame, and Henry A. Kautz, 'Performing bayesian inference by weighted model counting', in *20th National Conference on Artificial Intelligence (AAAI 2005)*, pp. 475–482, Palo Alto, California USA, (2005). AAAI Press.
[22] Prakash P. Shenoy and Glenn Shafer, 'Axioms for probability and belief-function proagation', in *4th Conference Conference on Uncertainty in Artificial Intelligence (UAI 1988)*, eds., Ross D. Shachter, Tod S. Levitt, Laveen N. Kanal, and John F. Lemmer, pp. 169–198. North-Holland, (1990).
[23] P. W. Shor, 'Algorithms for quantum computation: discrete logarithms and factoring', in *35th Annual Symposium on Foundations of Computer Science (FOCS 1994)*, pp. 124–134. IEEE Press, (1994).
[24] Bhuvanesh Sundar, Roger Paredes, David T Damanik, Leonardo Duenas-Osorio, and Kaden RA Hazzard, 'A quantum algorithm to count weighted ground states of classical spin hamiltonians', *arXiv preprint arXiv:1908.01745*, (2019).
[25] Krysta Svore, Alan Geller, Matthias Troyer, John Azariah, Christopher Granade, Bettina Heim, Vadym Kliuchnikov, Mariia Mykhailova, Andres Paz, and Martin Roetteler, 'Q#: Enabling scalable quantum computing and development with a high-level dsl', in *Real World Domain Specific Languages Workshop (RWDSL 2018)*, pp. 7:1–7:10, New York, NY, USA, (2018). ACM.
[26] Nevin Lianwen Zhang and David L. Poole, 'Exploiting causal independence in Bayesian network inference', *Journal of Artificial Intelligence Research*, **5**, 301–328, (1996).