Quantum Algorithms to Check Resiliency of Boolean Functions

[Extended Abstract]

Kaushik Chakraborty[1]
Subhamoy Maitra[1]

[1] Indian Statistical Institute
Kolkata, India

April, 2013 / WCC , Bergen, Norway

Outline

Qubit (preliminaries)

- Classical bits: $0, 1$.
- Quantum counterpart $|0\rangle, |1\rangle$.
- $|0\rangle$ can be written as $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$
- $|1\rangle$ can be written as $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$.
- Superposition of $|0\rangle, |1\rangle$: $\alpha|0\rangle + \beta|1\rangle$ can be written as $\alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$.
- $\alpha, \beta \in \mathbb{C}$, $|\alpha|^2 + |\beta|^2 = 1$.

Qubit and Measurement

- A qubit:
$$\alpha|0\rangle + \beta|1\rangle,$$
$\alpha, \beta \in \mathbb{C}$, $|\alpha|^2 + |\beta|^2 = 1$.
- Measurement in $\{|0\rangle, |1\rangle\}$ basis: we will get $|0\rangle$ with probability $|\alpha|^2$, $|1\rangle$ with probability $|\beta|^2$. The original state gets destroyed.
- Example:
$$\frac{1+i}{2}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle.$$
After measurement: we will get
$|0\rangle$ with probability $\frac{1}{2}$,
$|1\rangle$ with probability $\frac{1}{2}$.

Multi-Qubit System

- Tensor products among the qubits are used to represent a system of multiple qubits. For example, two qubits $|0\rangle$ and $|0\rangle$ together can be represented as

$$|00\rangle \equiv \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} \equiv \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$

- n qubit quantum state can be written as $|\psi_n\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle + ... + \alpha_{2^n-1}|2^n - 1\rangle$, where $|\alpha_0|^2 + |\alpha_1|^2 + ...|\alpha_{2^n-1}|^2 = 1$, in vector form it can be written as $\begin{pmatrix}\alpha_0 & \alpha_1 & \alpha_2 & . & . & . & \alpha_{2^n-1}\end{pmatrix}^T$

Operations on Qubits

- All operators are unitary operators.
- A quantum operator which can operate on n qubits is a $2^n \times 2^n$ unitary matrix having real or imaginary entries.
- **Example:** Hadamard operator $H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$,

  It is a single qubit operator.

U_f Operator

- For n bit Boolean function f, U_f will be an $n+1$ qubit operator.
- First n qubits will be called control bits, and the last bit will be called a target bit.
- It works in the following fashion,
  $U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$
- if $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, then
  $U_f|x\rangle|y\rangle = (-1)^{f(x)}|x\rangle|y\rangle$

Outline

Deutsch-Jozsa(DJ) Algorithm

Problem Statement :
Given a Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ with a promise that f is either balanced or constant. Decide which one it is.

Input : A Boolean function f on n variables is available in the form of the transformation U_f.

Output : If the state $|00...0\rangle$ is observed then conclude f is constant.
Otherwise conclude f is balanced

Deutsch-Jozsa(DJ) Algorithm

Problem Statement :
Given a Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ with a promise that f is either balanced or constant. Decide which one it is.

Input : A Boolean function f on n variables is available in the form of the transformation U_f.

Output : If the state $|00...0\rangle$ is observed then conclude f is constant.
Otherwise conclude f is balanced

Deutsch-Jozsa(DJ) Algorithm

Problem Statement :
Given a Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ with a promise that f is either balanced or constant. Decide which one it is.

Input : A Boolean function f on n variables is available in the form of the transformation U_f.

Output : If the state $|00...0\rangle$ is observed then conclude f is constant.
Otherwise conclude f is balanced

DJ Circuit

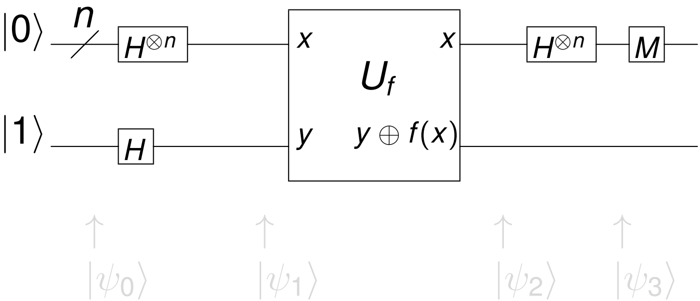

Figure : Quantum circuit to implement Deutsch-Jozsa Algorithm

DJ Circuit

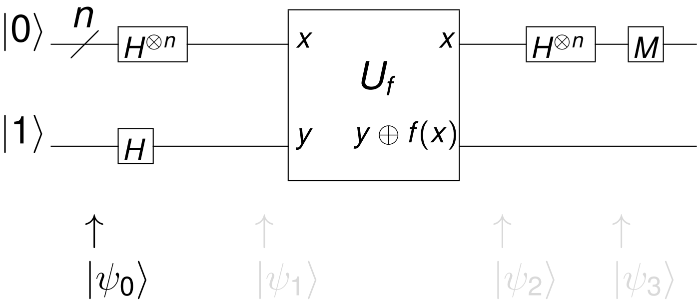

Figure : Quantum circuit to implement Deutsch-Jozsa Algorithm

DJ Circuit

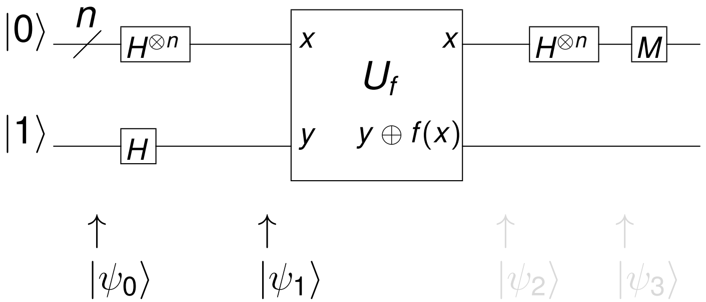

Figure : Quantum circuit to implement Deutsch-Jozsa Algorithm

DJ Circuit

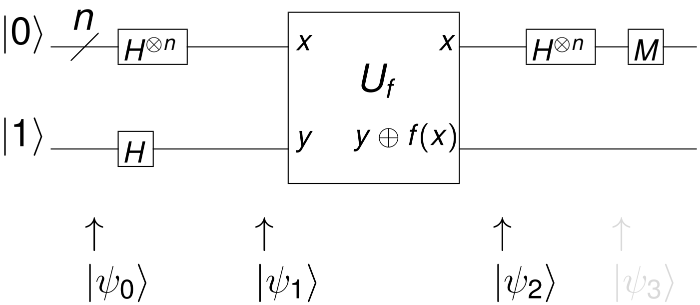

Figure : Quantum circuit to implement Deutsch-Jozsa Algorithm

DJ Circuit

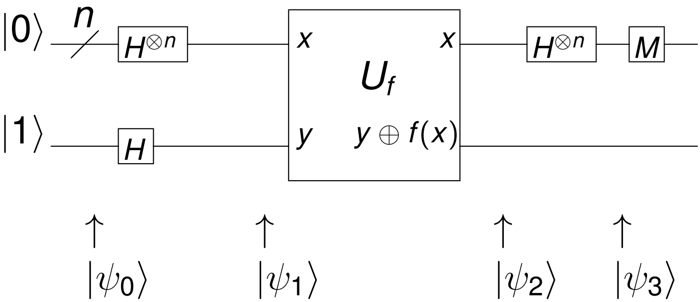

Figure : Quantum circuit to implement Deutsch-Jozsa Algorithm

Outline

1. Basics of Quantum Computation
2. Basic Quantum Algorithm and Resiliency Checking
   - Deutsch-Jozsa Algorithm
   - Resiliency Checking
3. Our Approach Towards Resiliency Checking
   - Improvement Using Grover Algorithm
   - Query Complexity
   - Exponential Speedup for Special Class of Boolean Functions
4. Conclusion
   - Potential Advantages
   - Future Work

From DJ to Walsh Spectrum *[Maitra et. al, IJQI, 2005]*

- Let $\mathcal{D}_f$ be the DJ operator, where, $\mathcal{D}_f = H^{\otimes n} U_f H^{\otimes n}$
- $\mathcal{D}_f$ operator converts the input state $|0^n\rangle$ to $|\psi_3\rangle$

  $\mathcal{D}_f|00\ldots0\rangle = |\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)}|z\rangle$
- $\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)} =$ Walsh spectrum value of the function f at point $z = W_f(z)$.
- So, we can write $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$

From DJ to Walsh Spectrum*[Maitra et. al, IJQI, 2005]*

- Let $\mathcal{D}_f$ be the DJ operator, where, $\mathcal{D}_f = H^{\otimes n} U_f H^{\otimes n}$

- $\mathcal{D}_f$ operator converts the input state $|0^n\rangle$ to $|\psi_3\rangle$

  $\mathcal{D}_f|00\ldots0\rangle = |\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)}|z\rangle$

- $\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)} =$ Walsh spectrum value of the function f at point $z = W_f(z)$.

- So, we can write $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$

From DJ to Walsh Spectrum*[Maitra et. al, IJQI, 2005]*

- Let $\mathcal{D}_f$ be the DJ operator, where, $\mathcal{D}_f = H^{\otimes n} U_f H^{\otimes n}$
- $\mathcal{D}_f$ operator converts the input state $|0^n\rangle$ to $|\psi_3\rangle$

$\mathcal{D}_f|00\ldots0\rangle = |\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)}|z\rangle$

- $\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)} =$ Walsh spectrum value of the function f at point $z = W_f(z)$.
- So, we can write $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$

From DJ to Walsh Spectrum *[Maitra et. al, IJQI, 2005]*

- Let $\mathcal{D}_f$ be the DJ operator, where, $\mathcal{D}_f = H^{\otimes n} U_f H^{\otimes n}$
- $\mathcal{D}_f$ operator converts the input state $|0^n\rangle$ to $|\psi_3\rangle$

$$\mathcal{D}_f|00\ldots0\rangle = |\psi_3\rangle = \frac{1}{2^n} \sum_{z\in\{0,1\}^n} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot z \oplus f(x)} |z\rangle$$

- $\sum_{x\in\{0,1\}^n} (-1)^{x\cdot z\oplus f(x)}$ = Walsh spectrum value of the function f at point $z = W_f(z)$.
- So, we can write $|\psi_3\rangle = \frac{1}{2^n} \sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$

From DJ to Walsh Spectrum*[Maitra et. al, IJQI, 2005]*

- Let $\mathcal{D}_f$ be the DJ operator, where, $\mathcal{D}_f = H^{\otimes n} U_f H^{\otimes n}$
- $\mathcal{D}_f$ operator converts the input state $|0^n\rangle$ to $|\psi_3\rangle$

$$\mathcal{D}_f|00\ldots0\rangle = |\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)}|z\rangle$$

- $\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)} =$ Walsh spectrum value of the function f at point $z = W_f(z)$.
- So, we can write $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$

From DJ to Walsh Spectrum *[Maitra et. al, IJQI, 2005]*

- Let $\mathcal{D}_f$ be the DJ operator, where, $\mathcal{D}_f = H^{\otimes n} U_f H^{\otimes n}$
- $\mathcal{D}_f$ operator converts the input state $|0^n\rangle$ to $|\psi_3\rangle$

$$\mathcal{D}_f|00\ldots0\rangle = |\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)}|z\rangle$$

- $\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)} =$ Walsh spectrum value of the function f at point $z = W_f(z)$.
- So, we can write $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$

From DJ to Walsh Spectrum *[Maitra et. al, IJQI, 2005]*

- Let $\mathcal{D}_f$ be the DJ operator, where, $\mathcal{D}_f = H^{\otimes n} U_f H^{\otimes n}$
- $\mathcal{D}_f$ operator converts the input state $|0^n\rangle$ to $|\psi_3\rangle$

  $\mathcal{D}_f|00\ldots0\rangle = |\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)}|z\rangle$

- $\sum_{x\in\{0,1\}^n}(-1)^{x\cdot z\oplus f(x)} =$ Walsh spectrum value of the function f at point $z = W_f(z)$.
- So, we can write $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$

Resiliency Checking

- a function $f \in \mathcal{B}_n$ is m-resilient iff its Walsh transform satisfies $W_f(z) = 0$, for $0 \leq wt(z) \leq m$
- **Goal :** Find some z, where $0 \leq wt(z) \leq m$, for which $W_f(z) \neq 0$

  If found such z, conclude f is not m-resilient

  Otherwise conclude f is m resilient
- $S_m = \{x \in \{0,1\}^n | wt(x) \leq m\}$
- $\overline{S}_m = \{x \in \{0,1\}^n | wt(x) > m\}$.

Resiliency Checking

- a function $f \in \mathcal{B}_n$ is m-resilient iff its Walsh transform satisfies $W_f(z) = 0$, for $0 \leq wt(z) \leq m$
- **Goal :** Find some z, where $0 \leq wt(z) \leq m$, for which $W_f(z) \neq 0$

  If found such z, conclude f is not m-resilient

  Otherwise conclude f is m resilient
- $S_m = \{x \in \{0,1\}^n | wt(x) \leq m\}$
- $\overline{S}_m = \{x \in \{0,1\}^n | wt(x) > m\}$.

From DJ to Resiliency Checking

- In DJ algorithm, $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$
- o, we can write $|\psi_3\rangle$ as
  $|\psi_3\rangle = \sum_{s\in S_m}\frac{W_f(s)}{2^n}|s\rangle + \sum_{s\in\overline{S}_m}\frac{W_f(s)}{2^n}|s\rangle$.
- Equivalently $|\psi_3\rangle = a|X\rangle + b|Y\rangle$, where,

$$a^2 = \sum_{s\in S_m}\frac{W_f^2(s)}{2^{2n}} \text{ and } b^2 = \sum_{s\in\overline{S}_m}\frac{W_f^2(s)}{2^{2n}}.$$

From DJ to Resiliency Checking

- In DJ algorithm, $|\psi_3\rangle = \frac{1}{2^n} \sum_{z \in \{0,1\}^n} W_f(z)|z\rangle$
- o, we can write $|\psi_3\rangle$ as
  $|\psi_3\rangle = \sum_{s \in S_m} \frac{W_f(s)}{2^n}|s\rangle + \sum_{s \in \overline{S}_m} \frac{W_f(s)}{2^n}|s\rangle$.
- Equivalently $|\psi_3\rangle = a|X\rangle + b|Y\rangle$, where,

$$a^2 = \sum_{s \in S_m} \frac{W_f^2(s)}{2^{2n}} \text{ and } b^2 = \sum_{s \in \overline{S}_m} \frac{W_f^2(s)}{2^{2n}}.$$

From DJ to Resiliency Checking

- In DJ algorithm, $|\psi_3\rangle = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} W_f(z)|z\rangle$
- o, we can write $|\psi_3\rangle$ as $|\psi_3\rangle = \sum_{s\in S_m}\frac{W_f(s)}{2^n}|s\rangle + \sum_{s\in \overline{S}_m}\frac{W_f(s)}{2^n}|s\rangle$.
- Equivalently $|\psi_3\rangle = a|X\rangle + b|Y\rangle$, where,

$$a^2 = \sum_{s\in S_m}\frac{W_f^2(s)}{2^{2n}} \text{ and } b^2 = \sum_{s\in \overline{S}_m}\frac{W_f^2(s)}{2^{2n}}.$$

Simple Algorithm to Check Resiliency

Take an $(n+1)$ qubit state $|\psi_0\rangle = |0\rangle^{\otimes n}|1\rangle$;
for $i = 1$ *to* r **do**
    Apply $\mathcal{D}_f \otimes H$ on $|\psi_0\rangle$ to get $|\psi_3\rangle = a|X\rangle + b|Y\rangle$;
    measure the first n qubits of $|\psi_3\rangle$ and let u be the output of the measurement;
    if $u \in S_m$ **then**
        Report that the function is not m-resilient (NO) and terminate;
    end
end
Report that the function is m-resilient (YES);

Algorithm 1: Resiliency Checking Using DJ algorithm

Query Complexity for *Algorithm* 1

Theorem

Let c be a predefined constant. Algorithm 1 correctly answers NO, but answers YES with success probability greater that or equal to c, in r many steps, where r is $O(\frac{1}{a^2})$ *and*
$a^2 = \sum_{s \in S_m} \frac{W_f^2(s)}{2^{2n}}$.

Proof.

According to *Algorithm* 1, one can observe that for each iteration, the success probability is a^2. At i-th step, the success probability will be $1 - (1 - a^2)^i$. So, at $i = r$ the success probability will become $1 - (1 - a^2)^r = c$. Now solving this equation we get r is $O(\frac{1}{a^2})$. $\square$

Outline

Amplitude Amplification using Grover Operator

Theorem

Let $|\Psi\rangle = \sum_{s \in S_m} \frac{W_f(s)}{2^n}|s\rangle + \sum_{s \in \overline{S}_m} \frac{W_f(s)}{2^n}|s\rangle = a|X\rangle + b|Y\rangle$*, where* $a = \sin\theta$*,* $b = \cos\theta$*. The application of* $[(2|\Psi\rangle\langle\Psi| - I)\mathcal{O}_g]^t$ *operator on* $|\Psi\rangle$ *produces* $|\Psi_t\rangle$*, in which the probability amplitude of* $|X\rangle$ *is* $\sin(2t+1)\theta$*.*

Resiliency Checking using Grover Algorithm

1 $S_m = \{x \in \{0,1\}^n | wt(x) \leq m\}$;
2 **for** $i = 0$ *to* r **do**
3   Apply Deutsch-Jozsa algorithm till the step before measurement to obtain $|\Psi\rangle = \sum_{s \in S_m} \frac{W_f(s)}{2^n} |s\rangle + \sum_{s \in \overline{S}_m} \frac{W_f(s)}{2^n} |s\rangle$;
4   By applying Grover iteration, obtain $|\Psi_{t_i}\rangle = [(2|\Psi\rangle\langle\Psi| - I)\mathcal{O}_g]^{t_i}|\Psi\rangle$;
5   Measure $|\Psi_{t_i}\rangle$ in computational basis to obtain n-bit string u;
6   **if** $u \in S_m$ **then**
        Report that the function is not m-resilient (NO) and terminate;
    end
end
7 Report that the function is m-resilient (YES);

Outline

When To Stop

- *Theorem* - 2 implies that at the first iteration the probability of success goes from $\sin^2\theta$ to $\sin^2 3\theta$
- In t iteration the success probability will become $\sin^2(2t+1)\theta$
- So, too much big value of t may lead to bad success probability

When To Stop

- *Theorem* - 2 implies that at the first iteration the probability of success goes from $\sin^2\theta$ to $\sin^2 3\theta$
- In t iteration the success probability will become $\sin^2(2t+1)\theta$
- So, too much big value of t may lead to bad success probability

When to Stop (Contd..)

- Value of $a = \sin\ \theta$ in $|\psi\rangle$ is not known as priori
- According to *Theorem* 2 the value of t_i depend upon the value of a or θ
- without loss of generality assume that $0 \leq |\theta| \leq \frac{\pi}{2}$
- Desired success probability is some predefined constant $c = \sin^2\ \theta_c$.
- Assume at **STEP 4** of *Algorithm* 2 $|\psi_{t_i}\rangle = \sin\theta_i|X\rangle + \cos\theta_i|Y\rangle$
- **GOAL :** to put θ_i into the region $[\theta_c, \pi - \theta_c]$

When to Stop (Contd..)

- Value of $a = \sin\ \theta$ in $|\psi\rangle$ is not known as priori
- According to *Theorem* 2 the value of t_i depend upon the value of a or θ
- without loss of generality assume that $0 \leq |\theta| \leq \frac{\pi}{2}$
- Desired success probability is some predefined constant $c = \sin^2\ \theta_c$.
- Assume at **STEP 4** of *Algorithm* 2 $|\psi_{t_i}\rangle = \sin\theta_i|X\rangle + \cos\theta_i|Y\rangle$
- **GOAL :** to put θ_i into the region $[\theta_c, \pi - \theta_c]$

When to Stop (Contd..)

- Value of $a = \sin\ \theta$ in $|\psi\rangle$ is not known as priori
- According to *Theorem* 2 the value of t_i depend upon the value of a or θ
- without loss of generality assume that $0 \leq |\theta| \leq \frac{\pi}{2}$
- Desired success probability is some predefined constant $c = \sin^2\ \theta_c$.
- Assume at **STEP 4** of *Algorithm* 2
  $|\psi_{t_i}\rangle = \sin\theta_i|X\rangle + \cos\theta_i|Y\rangle$
- **GOAL :** to put θ_i into the region $[\theta_c, \pi - \theta_c]$

Our Approach

- Divide the region $[0, \frac{\pi}{2}]$ into $r+1$ many parts, $\alpha_r \geq \theta, \alpha_r, \alpha_{r-1}, \ldots, \alpha_1 = \theta_c$ (in ascending order)
- So, $\exists i \in [1, r]$ such that $\alpha_{i+1} \leq \theta < \alpha_i$ or for $i = 0$, may be $\frac{\pi}{2} \leq \theta \leq \theta_c$
- At i-th step, $i \geq 1$ in *Algorithm* 2 we assume that $\alpha_{i+1} \leq \theta \leq \alpha_i$
- So, the value of t_i will be such that $\theta_c = (2t_i + 1)\alpha_{i+1} \leq (2t_i + 1)\theta \leq (2t_i + 1)\alpha_i = \pi - \theta_c$

Our Approach

- Divide the region $[0, \frac{\pi}{2}]$ into $r+1$ many parts, $\alpha_r \geq \theta, \alpha_r, \alpha_{r-1}, \ldots, \alpha_1 = \theta_c$ (in ascending order)
- So, $\exists i \in [1, r]$ such that $\alpha_{i+1} \leq \theta < \alpha_i$ or for $i = 0$, may be $\frac{\pi}{2} \leq \theta \leq \theta_c$
- At i-th step, $i \geq 1$ in *Algorithm* 2 we assume that $\alpha_{i+1} \leq \theta \leq \alpha_i$
- So, the value of t_i will be such that $\theta_c = (2t_i + 1)\alpha_{i+1} \leq (2t_i + 1)\theta \leq (2t_i + 1)\alpha_i = \pi - \theta_c$

Our Approach

- Divide the region $[0, \frac{\pi}{2}]$ into $r + 1$ many parts, $\alpha_r \geq \theta, \alpha_r, \alpha_{r-1}, \ldots, \alpha_1 = \theta_c$ (in ascending order)
- So, $\exists i \in [1, r]$ such that $\alpha_{i+1} \leq \theta < \alpha_i$ or for $i = 0$, may be $\frac{\pi}{2} \leq \theta \leq \theta_c$
- At i-th step, $i \geq 1$ in *Algorithm* 2 we assume that $\alpha_{i+1} \leq \theta \leq \alpha_i$
- So, the value of t_i will be such that $\theta_c = (2t_i + 1)\alpha_{i+1} \leq (2t_i + 1)\theta \leq (2t_i + 1)\alpha_i = \pi - \theta_c$

Our Approach

- Divide the region $[0, \frac{\pi}{2}]$ into $r+1$ many parts, $\alpha_r \geq \theta, \alpha_r, \alpha_{r-1}, \ldots, \alpha_1 = \theta_c$ (in ascending order)
- So, $\exists i \in [1, r]$ such that $\alpha_{i+1} \leq \theta < \alpha_i$ or for $i = 0$, may be $\frac{\pi}{2} \leq \theta \leq \theta_c$
- At i-th step, $i \geq 1$ in *Algorithm* 2 we assume that $\alpha_{i+1} \leq \theta \leq \alpha_i$
- So, the value of t_i will be such that $\theta_c = (2t_i + 1)\alpha_{i+1} \leq (2t_i + 1)\theta \leq (2t_i + 1)\alpha_i = \pi - \theta_c$

Our Approach (Contd..)

So, we can write,

$$(2t_i + 1)\alpha_i = \pi - \theta_c, \quad (1)$$

$$(2t_i + 1)\alpha_{i+1} = \theta_c. \quad (2)$$

Similarly, we can write

$$(2t_{i-1} + 1)\alpha_{i-1} = \pi - \theta_c, \quad (3)$$

$$(2t_{i-1} + 1)\alpha_i = \theta_c. \quad (4)$$

Thus, from (1), (4), and solving them by taking initial condition $t_1 = 0$ we get,

$$(2t_i + 1) = \frac{(\pi - \theta_c)^{(i-1)}}{\theta_c^{(i-1)}} \quad (5)$$

Our Approach (Contd..)

So, we can write,

$$(2t_i + 1)\alpha_i = \pi - \theta_c, \tag{1}$$

$$(2t_i + 1)\alpha_{i+1} = \theta_c. \tag{2}$$

Similarly, we can write

$$(2t_{i-1} + 1)\alpha_{i-1} = \pi - \theta_c, \tag{3}$$

$$(2t_{i-1} + 1)\alpha_i = \theta_c. \tag{4}$$

Thus, from (1), (4), and solving them by taking initial condition $t_1 = 0$ we get,

$$(2t_i + 1) = \frac{(\pi - \theta_c)^{(i-1)}}{\theta_c^{(i-1)}} \tag{5}$$

Our Approach (Contd..)

So, we can write,

$$(2t_i + 1)\alpha_i = \pi - \theta_c, \tag{1}$$

$$(2t_i + 1)\alpha_{i+1} = \theta_c. \tag{2}$$

Similarly, we can write

$$(2t_{i-1} + 1)\alpha_{i-1} = \pi - \theta_c, \tag{3}$$

$$(2t_{i-1} + 1)\alpha_i = \theta_c. \tag{4}$$

Thus, from (1), (4), and solving them by taking initial condition $t_1 = 0$ we get,

$$(2t_i + 1) = \frac{(\pi - \theta_c)^{(i-1)}}{\theta_c^{(i-1)}} \tag{5}$$

Determine the Value of r

- Assume that $a = \sin\theta$, if $\theta \to 0$, then $a \approx \theta$
- Implies in worst case $(2t_r + 1)\theta = \theta_c$
- So, $(2t_r + 1) \approx \frac{1}{a}$ and

$$r \approx \log_{\frac{\pi - \theta_c}{\theta_c}}\left(\frac{1}{a}\right)$$

- r is $O(\log(\frac{1}{a}))$

Determine the Value of r

- Assume that $a = \sin\theta$, if $\theta \to 0$, then $a \approx \theta$
- Implies in worst case $(2t_r + 1)\theta = \theta_c$
- So, $(2t_r + 1) \approx \frac{1}{a}$ and

$$r \approx \log_{\frac{\pi-\theta_c}{\theta_c}}\left(\frac{1}{a}\right)$$

- r is $O(\log(\frac{1}{a}))$

Determine the Value of r

- Assume that $a = \sin\theta$, if $\theta \to 0$, then $a \approx \theta$
- Implies in worst case $(2t_r + 1)\theta = \theta_c$
- So, $(2t_r + 1) \approx \frac{1}{a}$ and

$$r \approx \log_{\frac{\pi-\theta_c}{\theta_c}}(\frac{1}{a})$$

- r is $O(\log(\frac{1}{a}))$

Determine the Value of r

- Assume that $a = \sin\theta$, if $\theta \to 0$, then $a \approx \theta$
- Implies in worst case $(2t_r + 1)\theta = \theta_c$
- So, $(2t_r + 1) \approx \frac{1}{a}$ and

$$r \approx \log_{\frac{\pi-\theta_c}{\theta_c}}(\frac{1}{a})$$

- r is $O(\log(\frac{1}{a}))$

Example

- Let $f : \{0,1\}^3 \rightarrow \{0,1\}$, such that $f(x_2, x_1, x_0) = x_0 x_1 \oplus x_1 x_2$
- **GOAL :** to check whether f is 0 resilient or not.
- Assume that $c = \frac{1}{2}$, so, $\theta_c = \frac{\pi}{4}$
- Here $S_m = \{000\}$
- After applying the DJ algorithm the state will be $|\psi\rangle = \frac{1}{2}[|000\rangle + |010\rangle + |101\rangle - |111\rangle]$
- $a = \sin\theta = \sqrt{(\frac{1}{2})^2} = \frac{1}{2}, \theta = \frac{\pi}{6}$
- Now according to the algorithm, assume that $\frac{\pi}{4} \leq \theta \leq \frac{\pi}{2}$ and $t_0 = 0$, so at **STEP 5** we have to measure $|\psi\rangle$, probability of success will be $\frac{1}{4}$ which is less than c
- If measurement outcome is 000 then conclude that f is not 0 resilient. else go to next step and conclude with probability $\frac{1}{2}$ that $\theta < \frac{\pi}{4}$

Example (Contd..)

- In next iteration, $i = 1$, $t_1 = 1$ and assume that $\frac{\pi}{4} = (2t_1 + 1)\alpha_2 \leq (2t_1 + 1)\theta \leq (2t_1 + 1)\alpha_1 = \frac{3\pi}{4}$. So, apply the Grover operator on $|\psi\rangle$ t_i many times
- Now if we measure the state $|\psi_{t_1}\rangle$, then θ will become $3\theta = \frac{\pi}{2} \geq \theta_c$, so, the success probability will become greater than c
- Now if $|000\rangle$ is observed then conclude that f is not 0 resilient
  Otherwise resilient

Overall Query Complexity

Theorem

Let c be a predefined constant. Algorithm 2 correctly answers NO, but answers YES with success probability greater than or equal to c, in r, i.e., $O(\log \frac{1}{a})$ *many steps and the number of times the Grover operator is executed is* $O(\frac{1}{a})$ *where*

$$a^2 = \sum_{s \in S_m} \frac{W_f^2(s)}{2^{2n}}.$$

Overall Query Complexity (Contd..)

Proof.

How we estimate r is explained above. In Algorithm 2, at the i-th step we apply the operator $(2|\psi\rangle\langle\psi| - I)$, t_i times. Here i varies from 1 to r. So, the total number of times the Grover operator is applied is $T = \sum_{i=1}^{r} t_i$. So, $T = \frac{1}{2}[\sum_{i=1}^{r}(\frac{(\pi-\theta_c)^{(i-1)}}{\theta_c^{(i-1)}} - 1)]$.

By solving this equation we get,

$$T \approx \frac{1}{2}[\frac{1/a - 1}{(\pi - \theta_c)/\theta_c - 1} - \frac{1}{2}\{\log_{\frac{\pi-\theta_c}{\theta_c}}(\frac{1}{a})(\log_{\frac{\pi-\theta_c}{\theta_c}}(\frac{1}{a}) + 1)\}]. \quad (6)$$

So, the number of times the Grover operator is executed is $O(\frac{1}{a})$. □

Outline

3-valued walsh spectrum

- For any m-resilient function the walsh spectrum will be divisible by 2^{m+2} (*Sarkar et.al, CRYPTO 2000*)
- Consider the set of Boolean functions $A = \{f \in B_n | W_f(\omega) \equiv 0 mod 2^{m+2}\}$
- If $f \in A$ is m resilient then $a \geq \frac{2^{m+2}}{2^n}$
- For them according to *Algorithm* - 2 the query complexity will be $O(2^{n-m-2})$
- If $m \geq n - O(poly(\log n))$, then required query complexity will be $O(ploy(n))$
- Known classical algorithm will take $O(2^n)$ amount of time for deciding the resiliency of the Boolean function for this kind of Boolean functions. So, exponential speed up is achieved using quantum algorithm

Outline

Potential Advantage Over Existing Quantum Methods

- Using Grover like algorithm query complexity has been reduced from $O(\frac{1}{a^2})$ to $O(\frac{1}{a})$, imply **quadratic speed up** on the number of input bits
- Number of measurement is reduced from $O(\frac{1}{a^2})$ to $O(\log \frac{1}{a})$, imply **exponential reduction** in the measurement

Potential Advantage over Existing Classical Methods

- Achieve exponential speed up in checking the resiliency of some special class of boolean functions
- Also achieve exponential speedup over classical methods in some scenarios where the Boolean function is not m-resilient and the walsh spectrum values at points having weight less than or equal to m is very small

  No known classical method is capable of deciding whether the boolean function m-resilient or not, with lesser than $O(2^n)$ queries

  But if the sum of the squares of those non zero walsh spectrum values will be of $\Omega(2^n)$ then our algorithm will achieve exponential speedup

Potential Advantage over Existing Classical Methods

- Achieve exponential speed up in checking the resiliency of some special class of boolean functions
- Also achieve exponential speedup over classical methods in some scenarios where the Boolean function is not m-resilient and the walsh spectrum values at points having weight less than or equal to m is very small
  No known classical method is capable of deciding whether the boolean function m-resilient or not, with lesser than $O(2^n)$ queries
  But if the sum of the squares of those non zero walsh spectrum values will be of $\Omega(2^n)$ then our algorithm will achieve exponential speedup

Potential Advantage over Existing Classical Methods

- Achieve exponential speed up in checking the resiliency of some special class of boolean functions
- Also achieve exponential speedup over classical methods in some scenarios where the Boolean function is not m-resilient and the walsh spectrum values at points having weight less than or equal to m is very small
  No known classical method is capable of deciding whether the boolean function m-resilient or not, with lesser than $O(2^n)$ queries
  But if the sum of the squares of those non zero walsh spectrum values will be of $\Omega(2^n)$ then our algorithm will achieve exponential speedup

Outline

Application of Our Approach in Other Scenarios

- Other than resiliency checking, this Boolean functions and Grover based approach has some other applications, like the *Dicke state* preparation *Chakraborty et. al, arXiv:1209.5932*
- Using the walsh spectrum property of symmetric boolean functions *Dicke states* can be prepared
- The proposed technique has helped to achieve quadratic speed up over existing quantum methods for *Dicke state* preparation

THANK YOU