中山大學
SUN YAT-SEN UNIVERSITY

Characterizations of symmetric partial Boolean functions and time-space complexity of quantum computing

Daowen Qiu (邱 道 文)

Institute of Computer Science Theory

School of Data and Computer Science

Sun Yat-sen University, China

Joint work with Shenggen Zheng and Jozef Gruska

2018Mathematical & Physical Aspects of Information Sciences

January 5-7, 2018, 哈尔滨工业大学数学研究院，哈尔滨

Why do we study quantum computing?

Computing speed up： **Shor’s factorization algorithm、Grover’s search algorithm**

Security： **Quantum cryptography (BB84—Bennett and Brassard in 1984)**

Physical realization： Ion trap, optics, superconductivity, cavity quantum electrodynamics, nuclear magnetic resonance, etc.

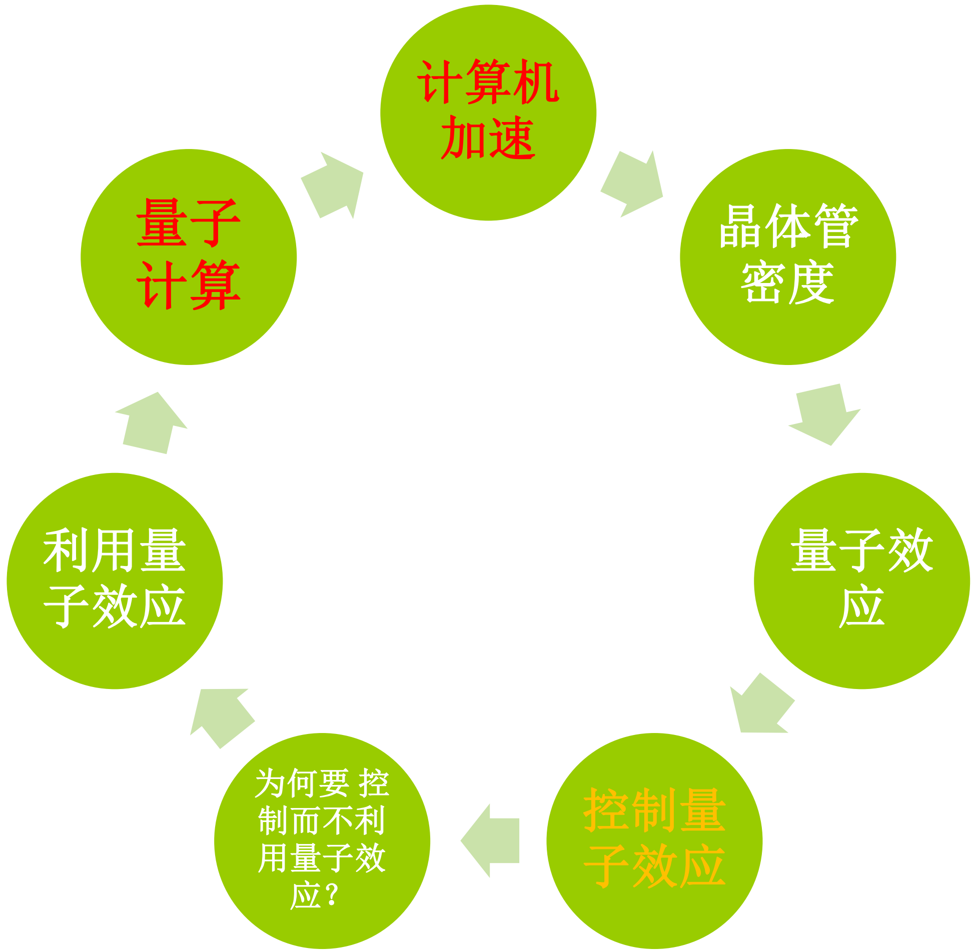
Development of Computers

Microcosmic world ——

——Quantum effect

2018MPAIS

微观世界——量子效应



Basic background

- Discover more problems to show that quantum computing is more powerful than classical computing
- These problems have the potential of applications in other areas such as cryptography.

Shor’s Algorithm

Polynomial- Time

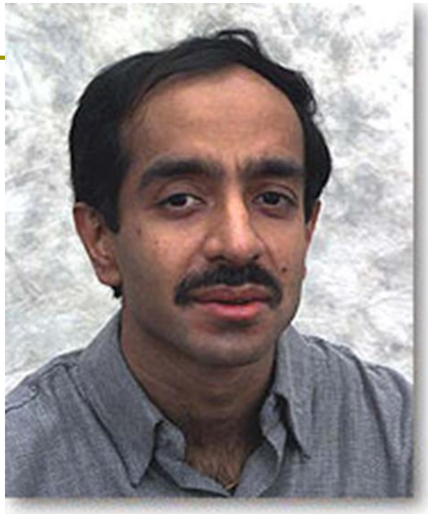
Peter Shor’s factorization algorithm

$$n = p \times q$$

Is it better?

However, we can not prove that it is strictly better than any classical algorithm, since we still do not know the classical lower bound of factorization.

Grover's Algorithm



Quadratic speed up

Search an unstructured database

It is exactly better

Quantum: $O(\sqrt{N})$ queries

Classical: $\Omega(N)$ queries

This Talk: TWO Problems

Deutsch-Jozsa Problem

Time-Space Complexity

Abstract

In this talk, I would like to report a recent work regarding:

1. An optimal exact quantum query algorithm for generalized Deutsch-Jozsa problem
2. The characterization of all symmetric partial Boolean functions with exact quantum 1-query complexity
3. A number of time-space complexity results concerning quantum finite automata

Outline

- 1. Motivations, Problems, Results
- 2. Preliminaries
- 3. Main Results
- 4. Methods of Proofs
- 5. Conclusions & Further Problems

1. Motivations, Problems, Results

- **(1) Generalized Deutsch-Jozsa**
- **(2) Problems with exact quantum 1-query complexity**
- **(3) Time-space complexity problems**

Motivation I

- The Deutsch-Jozsa promise problem [DJ'92]:

$x \in \{0,1\}^n$, $|\mathrm{x}|$ is the Hamming weight of x,

$$DJ(x) = \begin{cases} 0 & if\, |\mathrm{x}| = 0 \ or\ |\mathrm{x}| = n \\ 1 & if\ |\mathrm{x}| = n/2 \end{cases}$$

$$Q_E(DJ) = 1, D(DJ) = \frac{n}{2} + 1$$

- $DJ_n^1 = \begin{cases} 0 & if\, |\mathrm{x}| \leq 1\ \ or |\mathrm{x}| \geq n-1 \\ 1 & if\, |\mathrm{x}| = n/2 \end{cases}$ [MJM'15]

$$Q_E(DJ_n^1) \leq 2$$

[DJ'92] D. Deutsch, R. Jozsa, Rapid solution of problems by quantum computation, In Proceedings of the Royal Society of London, 439A (1992): 553—558.
[MJM'15] A. Montanaro, R. Jozsa, G. Mitchison, On exact quantum query complexity, Algorithmica**419** (2015) 775--796.

Problem I

- $DJ_n^k = \begin{cases} 0 & if\,|\mathrm{x}| \le k \;\; or\; |\mathrm{x}| \ge n-k \\ 1 & if\,|\mathrm{x}| = n/2 \end{cases}$ **?**

Our result:

Theorem 1 $Q_E(DJ_n^k) = k+1$ and $D(DJ_n^k) = n/2+k+1$.

Motivation II

Deutsch-Jozsa problem that is a symmetric partial Boolean function can be solved by DJ algorithm (exact quantum 1-query algorithm).

Then how to characterize the other symmetric partial Boolean functions with exact quantum 1-query complexity? Can such functions be solved by DJ algorithm?

Problem II

- **What can be solved with exact quantum 1-query complexity?**

- **Our result:**

- **Theorem 2 Any symmetric partial Boolean function f has $Q_E(f) = 1$ if and only if f can be computed by the Deutsch-Jozsa algorithm.**

Motivation III (BPP and BQP)

Advantages of Quantum Turing Machines

It is very hard to prove strictly that quantum Turing machines have advantages in time complexity over classical ones.

Lower Bounds

Indeed, it is very hard to find out lower bounds of time complexity in classical Turing machines.

Problem: Another way to show the advantages of quantum computing---Time-Space Complexity

For classical models

Some scholars have contributed it, e.g., Borodin, Cook, Babai, Nisan, Szegedy,etc.

For quantum models

Klauck, Spalek, de Wolf, SIAM J. Comput. **36** (2007) 1472–1493

Problem III

Time-Space complexity:

quantum finite automata

vs

probabilistic Turing machines

Our Results

For **recognizing some languages**, concerning the **time-space complexity:**

- Two-way probabilistic finite automata (2PFA) are strictly better than deterministic Turing machines (DTM)
- Two-way finite automata with quantum and classical states (2QCFA) are strictly better than probabilistic Turing machines (PTM)

Preliminaries

- Symmetric partial Boolean functions
- Classical query complexity
- Quantum query complexity
- Multilinear polynomials
- Quantum finite automata
- Communication complexity

Symmetric partial Boolean functions

- Let f be a Boolean function from $D \subseteq \{0,1\}^n$ to $\{0,1\}$. If $D = \{0,1\}^n$, then f is called a total Boolean function. Otherwise, f is called a partial Boolean function or a promise problem.
- A Boolean function f is called symmetric if $f(x)$ only depends on the Hamming weight (i.e. $|x|$) of x, that is, if $|x| = |y|$, then $f(x) = f(y)$.
- Given a partial Boolean function f with its domain of definition $D \subseteq \{0,1\}^n$, if for any $x \in D$, and $y \in \{0,1\}^n$, with $|x| = |y|$, we have $y \in D$, and $f(x) = f(y)$, then f is called a symmetrical partial Boolean function.

Representation of symmetric partial Boolean functions

- Given a partial symmetric function $f:\{0,1\}^n \to \{0,1\}$, with the domain D of definition, it can be equivalently described by a vector $(b_0, b_1, \dots, b_n) \in \{0,1,*\}^{n+1}$, where $f(x) = b_{|x|}$, i.e. b_k is the value of $f(x)$ when $|x| = k$, and $f(x)$ is `undefined' for $b_{|x|} = *$.

- Example

$$f(x) = x_1 \vee x_2 \qquad b = (b_0, b_1, b_2) = (0,1,1)$$
$$f(x) = x_1 \wedge x_2 \qquad b = (b_0, b_1, b_2) = (0,0,1)$$

Isomorphism of symmetric partial Boolean functions

- Two symmetric partial functions f and g over $\{0,1\}^n$ are isomorphic if they are equal up to negations and permutations of the input variables, and negation of the output variable.
- Concerning the n-bit symmetric partial functions, it is clear that the following four functions are isomorphic to each other:

$$(b_0, b_1, \ldots, b_n)\ ; \qquad (b_n, b_{n-1}, \ldots, b_0);$$
$$(\bar{b}_0, \bar{b}_1, \ldots, \bar{b}_n); \qquad (\bar{b}_n, \bar{b}_{n-1}, \ldots, \bar{b}_0).$$

Another simple example:

$$f(x) = x_1 \vee x_2 \qquad b = (b_0, b_1, b_2) = (0,1,1)$$
$$g(x) = x_1 \wedge x_2 \qquad b = (b_0, b_1, b_2) = (0,0,1)$$

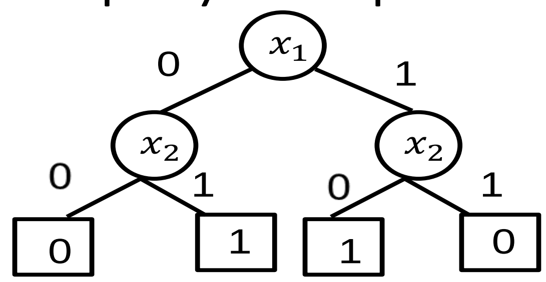
Classical query complexity

- An exact classical (deterministic) query algorithm to compute a Boolean function $f: \{0,1\}^n \rightarrow \{0,1\}$ can be described by a decision tree.

- If the output of a decision tree is $f(x)$, for all $x \in \{0,1\}^n$, the decision tree is said to "compute" f. The depthof a tree is the maximum number of queries that can happen before a leaf is reached and a result obtained.
- $D(f)$, the deterministic decision tree complexity of f is **the smallest depth** among all deterministic decision trees that compute f.

Example

- Deterministic query complexity
  (how many times we need to query the input bits)
- Example:
  $f(x_1, x_2) = x_1 \oplus x_2$



- Decision tree

The minimal depth over all decision trees computing f is the exact classical query complexity (deterministic query complexity, decision tree complexity) $D(f)$.

Quantum query algorithms

- Quantum T -query algorithm (its complexity is T)

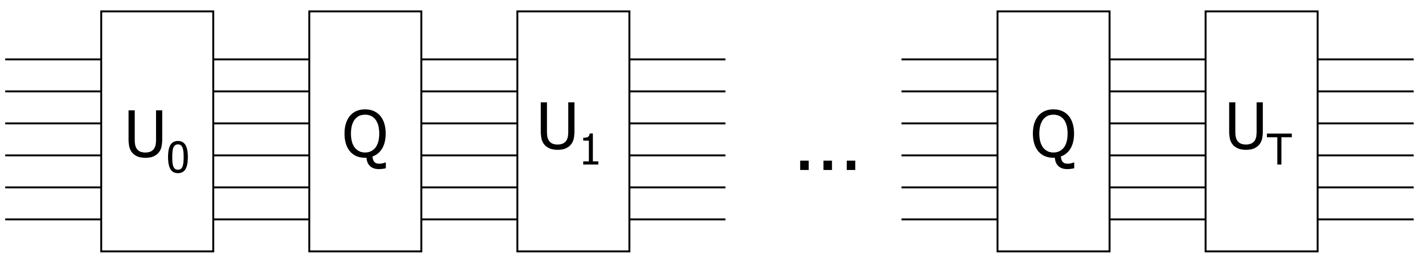
$f: \{0,1\}^n \to \{0,1\}$, input bit string $x = x_1 \cdots x_n$

We consider a Hilbert space $\mathcal{H}$ with basis state $|i, j\rangle$ for

$i \in \{0,1, \dots, n\}, j \in \{1, \dots, m\}$ (m can be chosen arbitrarily)

A T-query quantum algorithm:

$$|\psi_f\rangle = U_T Q_x U_{T-1} Q_x \cdots Q_x U_1 Q_x U_0 \, |\psi_s\rangle,$$



Black box

A T-query quantum algorithm:

$$|\psi_f\rangle = U_T Q_x U_{T-1} Q_x \cdots Q_x U_1 Q_x U_0 \, |\psi_s\rangle,$$

and then the algorithm performs a measurement,

where

$$Q_x|i,j\rangle = (-1)^{x_i}|i,j\rangle \text{ for } i \in \{1, \dots, n\}$$

$$Q_x|0,j\rangle = |0,j\rangle$$

$$|i,j\rangle \rightarrow \boxed{Q_x} \longrightarrow (-1)^{x_i}|i,j\rangle$$

- Deutsch-Jozsa's query box, Grover's query box

$$|i\rangle \rightarrow \boxed{Q_x} \longrightarrow (-1)^{x_i}|i\rangle$$

Quantum query complexity

The final state is then measured with a measurement $\{M_0, M_1\}$. For an input $x \in \{0,1\}^n$, we denote $A(x)$ the output of the quantum query algorithm A.

- We say that the quantum query algorithm A computes f within an error ε if for every input $x \in \{0,1\}^n$ it holds that $\Pr[A(x) = f(x)] \geq 1 - \varepsilon$.
- If ε=0, we says that the quantum algorithm is **exact**.
- $Q_\varepsilon(f), Q(f), Q_E(f)$ are **the smallest** T among all quantum query algorithms that compute f (with error ε, bounded-error, exact, respectively).

Multilinear polynomials

Every Boolean function $f: \{0,1\}^n \rightarrow \{0,1\}$ has a unique representation as an n-variate multilinear polynomial over the reals, i.e., there exist real coefficients a_S such that

$$f(x_1, \ldots, x_n) = \sum_{S \subseteq [n]} a_S \prod_{i \in S} x_i$$

The degree of f is the degree of its largest monomial:

$\deg(f) = \max\{|S|: a_S \neq 0\}$.

For example, $AND_2(x_1, x_2) = x_1 \cdot x_2$ and

$$OR_2(x_1, x_2) = x_1 + x_2 - x_1 \cdot x_2.$$

Multilinear polynomials representing symmetric partial Boolean functions

- Let f be a partial function with a domain of definition $D \subseteq \{0,1\}^n$. For $0 \leq \varepsilon < 1/2$, we say a real multilinear polynomial p approximates f with error ε if:

(1) $|p(x) - f(x)| \leq \varepsilon$ for all $x \in D$;
(2) $0 \leq p(x) \leq 1$ for all $x \in \{0,1\}^n$.

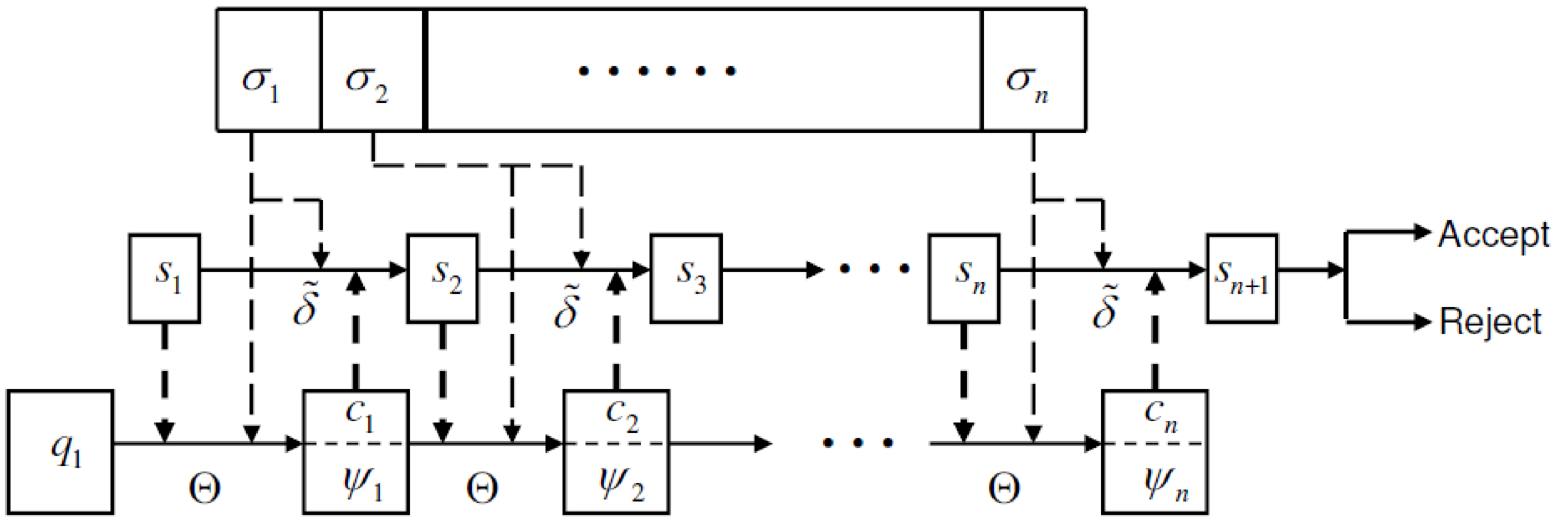
- The approximate degree of f with error ε, denoted by $\widetilde{deg}_{\varepsilon}(f)$, is the minimum degree among all real multilinear polynomials that approximate f with error ε. In particular,

$$\widetilde{deg}_0(f) \triangleq \deg(f)$$

Quantum Finite Automata (QFA)

- QFA—simpler models
- Here we employ **two-way finite automata with quantum and classical states (2QCFA)** first proposed by Ambainis and Watrous
- We also appropriately compare with classical Turing machines

2QCFA---Semi-quantum finite automata



2QCFA---simpler than quantum Turing machines
2QCFA---more complicated than one-way quantum finite automata

Time and space complexity of QFA

- Time complexity T(|x|): For input x, T(|x|) is the steps of the machines operating, where |x| denotes the length of x with Binary coding mode.
- Space complexity (state complexity) S: The number of (qu)bits required to represent the automaton states.

Time-Space complexity of QFA

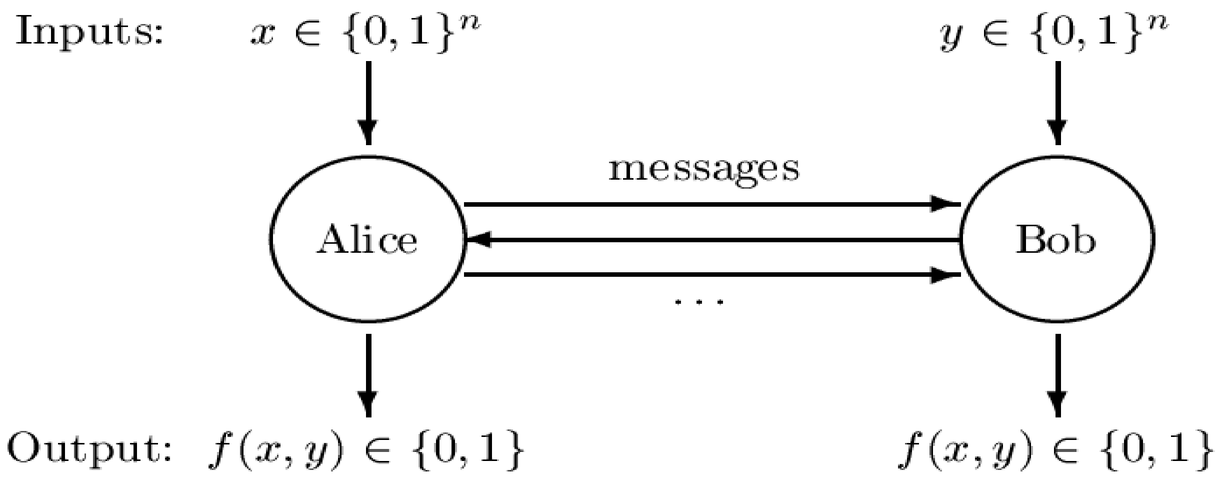
Time-Space complexity---The product of Time and Space:

$$\mathrm{T(Time)} \cdot S\mathrm{(Space)}$$

Similarly, Time-Space complexity for Turing machines, but the space complexity depends on the amount of memory consumed by the computation.

Communication complexity

- (Two-way) Communication complexity model



- There are three kinds of communication complexities according to the models (or protocols) used by Alice and Bob
  - Deterministic
  - Probabilistic
  - Quantum

Two most studied problems

$x, y \in \{0,1\}^n$

- **Equality**

EQ(x,y)=1 if x=y and 0 otherwise.

- **Intersection**

INT(x,y)=1 if there is an index *i* such that $x_i = y_i = 1$ and 0 otherwise.

Methods of Proofs

We would like to outline the basic ideas and methods for the proofs of main results.

$$DJ_n^k = \begin{cases} 0 & if\,|x| \le k \;\; or\; |x| \ge n-k \\ 1 & if\,|x| = n/2 \end{cases}$$

Theorem 1 $Q_E\left(DJ_n^k\right) = k+1$ and $D\left(DJ_n^k\right) = \frac{n}{2} + k + 1.$

Proof method:

- Using the exact quantum query algorithms for computing $EXACT_n^k$ and due to Ambainis *et al.* (TQC'13), we can give an exact quantum $(k+1)$-query algorithm for computing DJ_n^k
- On the other hand, we will prove that $\deg\left(DJ_n^k\right) \ge 2k+2$, and therefore

$$Q_E\left(DJ_n^k\right) \ge \deg(DJ_n^k)/2 = k+1$$

$Q_E\left(DJ_n^k\right) \leq k+1$

Subroutine: Xquery(m, x) 【from Ambainis *et al.* (TQC'13)】

Input: $x = x_1, x_2, \dots, x_m$.

Output: $(0,0) \Rightarrow |x| \neq \frac{m}{2}$

$(i, j) \Rightarrow x_i \neq x_j$

Algorithm 2 Algorithm for DJ_n^k

```
procedure DJ(integer n, integer k, array x)
    integer l:=1
    while l ≤ k do
        Output ←Xquery(n, x)
        if Output=(0,0) then return 0
        end if
        if Output=(i, j) then
            x ← x \ {x_i, x_j}
            l ← l + 1
            n ← n − 2
        end if
    end while
    Output ←Xquery(n, x)
    if Output=(0,0) then return 0
    end if
    if Output=(i, j) then return 1
    end if
end procedure
```

$\deg\left(DJ_n^k\right) \geq 2k + 2$

Lemma: For any symmetrically partial Boolean function f over $\{0,1\}^n$ with domain of definition D, suppose $\deg_\varepsilon(f) = d$. Then there exists a real multilinear polynomial q approximates f with error ε and q can be written as

$$q(x) = c_0 + c_1V_1 + c_2V_2 + \cdots + c_dV_d,$$

where $c_i \in R, V_1 = x_1 + \cdots + x_n, V_2 = x_1x_2 + x_1x_3 + \cdots + x_{n-1}x_n, \cdots$

Suppose that $\deg\left(DJ_n^k\right) \leq 2k + 1$. Then we can get a contradiction. So, $\deg\left(DJ_n^k\right) \geq 2k + 2$ follows.

Theorem: $Q_E(f) = 1$ *if and only if* f can be computed by DJ algorithm

Lemma 1 Let $n > 1$ and let $f: \{0,1\}^n \to \{0,1\}$ be an n-bit symmetrically partial Boolean function. Then:

(1) $\deg(f) = 1$ iff f is isomorphic to the function $f_{n,n}^{(1)}$

(2) $\deg(f) = 2$ iff f is isomorphic to one of the functions

$$f_{n,k}^{(1)}(x) = \begin{cases} 0 & \text{if } |x| = 0, \\ 1 & \text{if } |x| = k, \end{cases} \qquad f_{n,l}^{(3)}(x) = \begin{cases} 0 & \text{if } |x| = 0 \text{ or } |x| = n, \\ 1 & \text{if } |x| = l, \end{cases}$$

$$f_{n,k}^{(2)}(x) = \begin{cases} 0 & \text{if } |x| = 0, \\ 1 & \text{if } |x| = k \text{ or } |x| = k+1, \end{cases} \qquad f_{n}^{(4)}(x) = \begin{cases} 0 & \text{if } |x| = 0 \text{ or } |x| = n, \\ 1 & \text{if } |x| = \lfloor n/2 \rfloor \text{ or } |x| = \lceil n/2 \rceil, \end{cases}$$

where $n - 1 \geq k \geq \lfloor n/2 \rfloor$, and $\lceil n/2 \rceil \geq l \geq \lfloor n/2 \rfloor$.

Two Lemmas

- Lemma. Let $\boldsymbol{n}$ be even. Then $\boldsymbol{QE(f)=1}$ if and only if $\boldsymbol{f}$ is isomorphic to one of these functions: $\boldsymbol{f_{n,k}^{(1)}}$ and $\boldsymbol{f_{n,n/2}^{(3)}}$, $\boldsymbol{k \geq \frac{n}{2}}$.
- Lemma. Let $\boldsymbol{n}$ be odd. Then $\boldsymbol{QE(f)=1}$ if and only if $\boldsymbol{f}$ is isomorphic to one of these functions: $\boldsymbol{f_{n,k}^{(1)}}$, $\boldsymbol{k \geq \lceil n/2 \rceil}$.

Equivalence transformation

- **Indeed, these functions with exact quantum 1-query complexity can be essentially transformed into DJ problem by padding some zeros into the input string. So, $QE(f) = 1$ if and only if f can be computed by DJ algorithm.**

Results concerning time-space complexity

Due to the previous results of Grover, Buhrman, Klauck, Ambainis, Watrous, and Cleve etc., I would like to report a number of results regarding Time-Space complexity of probabilistic automata and quantum automata for recognizing the following languages:

1. $\mathrm{L_{EQ}(n)} = \{\mathrm{x}\,\#^n\mathrm{y} \mid \mathrm{x,y} \in \{0,1\}^n, \mathrm{EQ(x,y)} = 1\}$

2. $\mathrm{L_{INT}(n)} = \{\mathrm{x}\,\#^n\mathrm{y} \mid \mathrm{x,y} \in \{0,1\}^n, \mathrm{INT(x,y)} = 1\}$

3. $\mathrm{L_{NE}(n)} = \{\mathrm{x}\,\#^n\mathrm{y} \mid \mathrm{x,y} \in \{0,1\}^n, \mathrm{REN(x,y)} = 1\}$

Result 1

- $L_{EQ}(n) = \{x\ \#^n y | x, y \in \{0,1\}^n, EQ(x,y) = 1\}$

- There is a 2PFA that accepts the language $L_{EQ}(n)$ in the time T using the space S such that $T \cdot S = \mathbf{O}(n \cdot \log n)$. Let A be a DTM that accepts the language $L_{EQ}(n)$ in time T' using space S'. Then,
- $T' \cdot S' = \mathbf{\Omega}(n^2)$.

Result 1

$$L_{EQ}(n) = \{x\#^n y | x, y \in \{0,1\}^n, EQ(x,y) = 1\}$$

- (Time complexity) 2DFA recognize $L_{EQ}(n)$ with O(n) time.
- (Space complexity) It is clear that 2DFA recognize $L_{EQ}(n)$ with $O(n^2)$ states, i.e. $O(\log n)$ space.
- It is clear that 2PFA will use the same time and space complexity to recognize the language.

Proof idea of Result 1

$$L_{EQ}(n) = \{x\,\#^n y \mid x, y \in \{0,1\}^n, EQ(x,y) = 1\}$$

- Time-space complexity ($T \cdot S$) for 2PFA

Proof (main idea)

1. Choose randomly a prime p
2. Calculate $\mathrm{Num}(x)$ with the input “x-region”
3. Skip the “#-region”
4. Calculate $\mathrm{Num}(y)$ with the input “y-region”
5. If $\mathrm{Num}(x) = \mathrm{Num}(y)$, accept the input.

   O.W reject

All the steps can be done in a 2PFA

Time: O(n) Space: O(log n) (no more than n^6 states)

Proof ideas of Result 1

$$\mathrm{L_{EQ}(n)} = \{\mathrm{x}\,\#^{n}\mathrm{y} \mid \mathrm{x,y} \in \{0,1\}^{n}, \mathrm{EQ(x,y)} = 1\}$$

- Time-space complexity ($\mathrm{T} \cdot \mathrm{S}$) for 2PFA

All the steps can be done in a 2PFA

Time: O(n) Space: O(log n) (no more than n^6 states)

- Lower bound for DTM

The deterministic communication complexity for $\mathrm{EQ(x,y)}$ is $\boldsymbol{\Omega}(n)$.

Assuming that the DTM use T time, there is most T/n rounds that "x-region" communicate with "y-region". Suppose the space using by the DTM is S, therefore

$$\frac{T}{n} \times S = \boldsymbol{\Omega}(n) \Rightarrow T \times S = \boldsymbol{\Omega}(n^2)$$

Result 2

- $L_{INT}(n) = \{x\,\#^n y | x, y \in \{0,1\}^n, INT(x,y) = 1\}$

- There is a 2QCFA that accepts the language $L_{INT}(n)$ in the time T using the space S such that
- $T \cdot S = \mathbf{O}\left(n^{3/2} \cdot \log n\right)$.
- Let A be a PTM that accepts the language $L_{INT}(n)$ in time T' using space S' . Then, $T' \cdot S' = \mathbf{\Omega}\,(n^2)$.

Result 2

- $L_{INT}(n) = \{x\#^n y | x, y \in \{0,1\}^n, INT(x,y) = 1\}$
- Proof main idea

We will use the 2QCFA to simulate quantum query algorithm.

Theorem 7. *The computation of a quantum query algorithm $\mathcal{A}$ for a Boolean function $f : \{0,1\}^n \to \{0,1\}$ can be simulated by a 2QCFA $\mathcal{M}$. Moreover, if the quantum query algorithm $\mathcal{A}$ uses t queries and l quantum basis states, then the 2QCFA $\mathcal{M}$ uses $\mathbf{O}(l)$ quantum basis states, $\mathbf{O}(n^2)$ classical states, and $\mathbf{O}(t \cdot n)$ time.*

Result 2

- $L_{INT}(n) = \{x \#^n y | x, y \in \{0,1\}^n, INT(x, y) = 1\}$

- Proof main idea

  We will use the 2QCFA to simulate quantum query algorithm.

  Let $z = x \wedge y$ be a bit-wise AND of x and y, run the quantum search (Grover) algorithm on z. We can find out there is a 2QCFA recognizing $L_{INT}(n)$.

Result 2

SUN YAT-SEN UNIVERSITY

- $L_{INT}(n) = \{x\,\#^n y \mid x, y \in \{0,1\}^n, INT(x,y) = 1\}$
- Proof main idea

  We will use the 2QCFA to simulate quantum query algorithm.

- Let $z = x \wedge y$ is a bit-wise AND of x and y, run the quantum search (Grover) algorithm on z. We can find out there is a 2QCFA recognizing $L_{INT}(n)$.
- Grover algorithm:

  $\mathbf{O}(\sqrt{n})$ queries, O(n) quantum basis states

Time for 2QCFA: $\mathbf{O}(\sqrt{n}) \times \mathbf{O}(n) = \mathbf{O}\left(n^{3/2}\right)$

Space for 2QCFA: classical states $\mathbf{O}(n^2)$, quantum states $\mathbf{O}(n)$,
$S = O(\log n^2 + \log n) = O(\log n)$

$$T \cdot S = \mathbf{O}\left(n^{3/2} \cdot \log n\right)$$

Result 2

- $L_{INT}(n) = \{x\,\#^n y \mid x, y \in \{0,1\}^n, INT(x,y) = 1\}$

- Lower bound for PTM
- The probabilistic communication complexity for $INT(x,y)$ is $\boldsymbol{\Omega}(n)$.
- We can prove that the time-space complexity for $L_{INT}(n)$ is $\boldsymbol{\Omega}(n^2)$.

Result 3

- It has been proved (Klauck, STOC'00) that the exact one-way quantum finite automata have no advantage over the classical finite automata in recognizing languages.

- How about exact two-way quantum finite automata ?

Result 3

- We will show that **exact** 2QCFA have time-space advantages over their classical counterparts.

Let us consider the sequence of functions studied in (STOC'13).

Let us first recall the function $NE(x_1, x_2, x_3)$ as follows:

- $NE(x_1, x_2, x_3) = 0 \; if \; x_1 = x_2 = x_3$ and
- $NE(x_1, x_2, x_3) = 1$ otherwise. Now we can define a sequence of functions NE^d as follows:
- (1) $NE^0(x_1) = x_1$ and
- (2)$NE^d\left(x_1, \ldots, x_{3^d}\right) = NE(NE^{d-1}\left(x_1, \ldots, x_{3^{d-1}}\right),$ $NE^{d-1}\left(x_{3^{d-1}+1}, \ldots, x_{2\cdot 3^{d-1}}\right), NE^{d-1}\left(x_{2\cdot 3^{d-1}+1,} \ldots, x_{3^d}\right))$

Result 3

- We will show that **exact** 2QCFA have advantage over their classical counterparts.

Let $n = 3^d$, we now define the function

$$RNE(x, y) = NE^d(x_1 \wedge y_1, \ldots, x_n \wedge y_n),$$

where $x, y \in \{0,1\}^n$, and let us consider the following language

$$L_{NE}(n) = \{x\#^n y \mid x, y \in \{0,1\}^n, \mathrm{RNE}(x, y) = 1\}.$$

Result 3

- There is an **exact** 2QCFA that accepts the language $L_{NE}(n)$ in the time T using the space S such that $T \cdot S = \mathbf{O}(n^{1.87} \cdot \log n)$.

- Let A be a PTM that accepts the language $L_{NE}(n)$ in time T' using space S'. Then, $T' \cdot S' = \boldsymbol{\Omega}(n^2)$.

- Proof (main idea)

Using the idea of Ambainis's exact query algorithm in (SOTC'13)

Result 3

中山大學 SUN YAT-SEN UNIVERSITY

- There is an exact 2QCFA that accepts the language $L_{NE}(n)$ in the time T using the space S such that $T \cdot S = \mathbf{O}(n^{1.87} \cdot \log n)$.

- Let A be a PTM that accepts the language $L_{NE}(n)$ in time T' using space S'. Then, $T' \cdot S' = \mathbf{\Omega}(n^2)$.

- Lower bound for PTM
- The probabilistic communication complexity for $RNE(x, y)$ is $\mathbf{\Omega}(n)$.
- We can prove that the time-space complexity for $RNE(x, y)$ is $\mathbf{\Omega}(n^2)$.

Conclusions

- $DJ_n^k = \begin{cases} 0 & if\,|\mathrm{x}| \leq k \;\; or\; |\mathrm{x}| \geq n-k \\ 1 & if\,|\mathrm{x}| = n/2 \end{cases}$

Theorem. $Q_E\left(DJ_n^k\right) = k+1$ and $D\left(DJ_n^k\right) = n/2+k+1.$

Theorem. Any symmetric partial Boolean function f has $Q_E(f) = 1$ if and only if f can be computed by the Deutsch-Jozsa algorithm.

Theorem. Two-way probabilistic finite automata (2PFA) are strictly better than deterministic Turing machines (DTM);

Two-way finite automata with quantum and classical states (2QCFA) are strictly better than probabilistic Turing machines (PTM).

Problems

- Let $f: \{0,1\}^n \rightarrow \{0,1\}$ be an n-bit symmetric partial Boolean function with domain of definition D, and let $0 \leq k < \lfloor \frac{n}{2} \rfloor$. Then, for $2k+1 \leq \deg(f) \leq 2(k+1)$, how to characterize f by giving all functions with degrees from $2k+1$ to $2k+2$?
- For the function $DW_n^{k,l}$ defined as:

$$\mathrm{DW}_n^{k,l}(x) = \begin{cases} 0 & \text{if } |x| = k, \\ 1 & \text{if } |x| = l, \end{cases}$$

can we give optimal exact quantum query algorithms for any k and l?

- We have studied the time-space complexity of 2PFA vs DTM, and of 2QCFA vs PTM, but their definitions for space complexity are different, so these results need be further considered.
- How about for 2PFA vs exact 2QCFA?
- How about for PTM vs QTM?

A useful reference

- H. Buhrman and R. de Wolf, Complexity measures and decision tree complexity: a survey, Theoretical Computer Science, 288 (2002) 1-43.

Thank you for your attention!

2018MPAIS