

Quantum Search Algorithm for Set Operation

Chao-Yang Pang*

Key Software Lab., Sichuan Normal University, Chengdu 610066, China;

College of Mathematics and Software Science,

Sichuan Normal University, Chengdu 610066, China

Cong-Bao Ding

College of Physics and Electronic Engineering, Sichuan Normal University, and

Chengdu 610066, China

Ben-Qiong Hu

College of Information Management,

Chengdu University of Technology, 610059, China

Abstract

The operations of data set, such as intersection, union and complement, are the fundamental calculation in mathematics. It's very significant that designing fast algorithm for set operation. In this paper, the quantum algorithm for intersection is presented. And its running time is $O\left(\sqrt{|A| \times |B| \times |C|}\right)$ for set operation $C = A \cap B$, while classical computation needs $O(|A| \times |B|)$ steps of computation in general, where $|\cdot|$ denotes the size of set. The presented algorithm is the combination of Grover's algorithm, classical memory and classical iterative computation, and the combination method decrease the complexity of designing quantum algorithm. The method can be used to design other set operations also.

Keywords: Set operation, General Grover iteration, Grover's algorithm

*Electronic address: cyp'900@hotmail.com; Electronic address: cypang@sicnu.edu.cn

I. INTRODUCTION

The operations of data set, such as intersection operation and union operation, are fundamental calculation in mathematics. The fast computation of set operation is very important because it's the base of many sciences and techniques, such as database, image processing, signal processing. E.g, database search is based on set operation and the fast computation of set operation is very important for it.

The computation procedure of set operation on electronic computer is illustrated as below.

Suppose there are two vector sets A and B ,

$$A = \{(1, 1, 1, 1), (2, 2, 2, 2), (1, 2, 3, 4)\},$$

$$B = \{(3, 3, 3, 3), (4, 4, 4, 4), (1, 2, 3, 4)\},$$

and the intersection set $C = A \cap B = \{(1, 2, 3, 4)\}$.

Firstly, all vectors of set A (or B) are stored in electronic memory and each vector seems to be a record of database. The computation procedure of set operation $A \cap B$ is that, for every vector in set A , computer fully searches all elements in set B and matches it. Because sorting multi-dimensional vectors is no useful for the speedup of search in general, all vectors of set are unsorted. Thus, the method of full search becomes the necessary choice to calculate intersection set for electronic computer, which is low efficient when set has huge size.

In addition, the running speed of I/O (Input/Output) equipment of classical computer is the efficiency bottleneck in term of arbitrary classical algorithm [1]. It's the computation procedure of classical computer that loading data into registers *one by one* via I/O, then executing calculation instructions *one by one* [1]. If a set has huge amount of elements, the process of loading data will waste the time heavily and it's an efficiency bottleneck. E.g., server computer for the database search is more expensive than personal computer, and one important reason is that advanced I/O is used. If the size of set is huge, set operation faces the bottleneck, and there is no way to overcome it on classical computation principle.

Therefore, for the sets with huge size, electronic computer can do nothing for the requirement of fast computation. We need new computation principle and new algorithm for set operation.

Fortunately, in the last decade, quantum computation is studied and many surprising computation properties are revealed so that the research of quantum computation becomes

the one of hottest research topic currently. One of milestone of quantum computation researcher is Shor's algorithm for factoring an integer number with polynomial computation steps, which is believed to be classically impossible [2]. And Grover presented another exciting algorithm for database search in 1996. Only $O(\sqrt{N})$ steps of computation are cost by Grover's algorithm to find a marked element in an unsorted database with size N , while $O(N)$ steps are needed for classical computer [3]. The possible speedup of quantum computation is essentially enabled by the feature of quantum parallelism. This parallelism computation of quantum computer bases on the superposition property of states, which is not possible on electronic computers [4]. It's the computation procedure of quantum computer that data is loaded into the superposition of states, the superposition is operated by special unitary operation, the amplitude of solution is increased, and solution is measured out with big probability at last. The simple quantum computers have already been constructed. For example, Shor's algorithm has been demonstrated by NMR quantum computer [5] and by optical quantum computer [6] to factorize the number 15.

As well known, the elements of set are arbitrary data (or random data), the size of set is very big possibly, and all data are stored in electronic memory unorderedly and temporarily. If we want to study the question that how to use quantum computation to perform set operation, there are three works must be considered at least. The first work is that how to express the information of a set using quantum state. The second work is that how to load the information of a set into quantum state from electronic memory. The matching function between two elements f_c is a computation, such as judging if two vector is equal. And the third work is that how to embed the computation f_c into quantum search algorithm.

For the first work (i.e., how to express the set information using state), there are two expression methods currently. One of method is proposed by Latorre that the information of classical data is encoded in the amplitude of a state. Latorre used his method to expression image data and detail information is lost [7]. Latorre's method is useful for image compression, but it's not suitable to express the information of general set because distortion of information is not permitted in set operation. The other quantum expression is proposed by Pang that all elements are regarded as sequence of database record $\{record_0, record_1, \dots, record_{N-1}\}$ and the entangled state $\frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} |i\rangle_{register1} |record_i\rangle_{register2} \right)$ is used to express the information of set [8, 9, 10, 11, 12, 13, 14]. Two registers are entangled in Pang's method, and it is no distortion theoretically. In addition, the operation of set is

equivalent to operating the entangled state.

For the second work (i.e., how to load the set information into state from electronic memory), the conception of Quantum Loading Scheme (QLS) should be introduced. QLS is the unitary operation that loading all information of data set into quantum state from electronic memory. Nielsen and Chuang point out briefly that future quantum computer should have QLS [4, Section 6.5]. Vittorio Giovannetti, Seth Lloyd and et.al., present a simple QLS instance with few qubits [15]. Pang presents a QLS instance using the path interference of molecule [11]. Pang's study shows that, for a vector $\vec{a} = \{a_1, a_1, \dots, a_{N-1}\}$, there is a unitary operation U_{QLS} [11] such that

$$U_{QLS} : |0\rangle_{register1}|0\rangle_{register2}|ancilla\rangle_{register3} \longrightarrow \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} |i\rangle_{register1}|a_i\rangle_{register2} \right) |ancilla\rangle_{register3}$$

, where classical data a_1, a_1, \dots, a_{N-1} are used as control signals to flip the particles.

Fig.1 is the illustration of QLS.

The study of Seth Lloyd's group and Pang's study show also that QLS is fast and has running time $O(\log_2 N)$, while classical loading scheme via I/O has running time $O(N)$ that is the efficiency bottleneck in term of arbitrary classical algorithm. Pang also present a variant QLS named unitary operation U_L as below[11]

$$U_L : \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} |i\rangle_{register1}|0\rangle_{register2} \right) |ancilla\rangle \longrightarrow \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} |i\rangle_{register1}|a_i\rangle_{register2} \right) |ancilla\rangle$$

The function of operation U_L is that loading data from electronic memory into two entangled registers according indices of data.

In sum, Nielsen and Chuang points out that QLS has to be existed, and the study of Seth Lloyd's group and Pang's study both demonstrate the existence of QLS.

For the third work (i.e., how to embed other computation into quantum search algorithm), the conception of the general Grover iteration should be introduced. Additional computation always goes with the search of database in general. E.g., suppose there is a database to save the student scores of many subjects. And if we want to find the student who has maximum average score, the additional computation f_c that calculating average score is also needed. Famous Grover's algorithm can find a database record according to the given index, and it

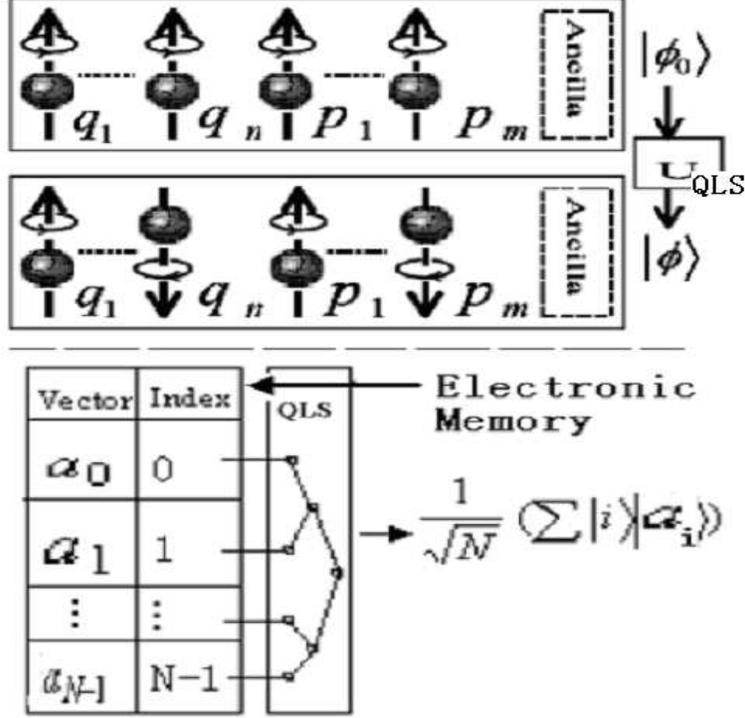


FIG. 1: The Illustration of Quantum Loading Scheme (QLS): The function of QLS is to load all information of vector $\vec{a} = \{a_0, a_1, \dots, a_{N-1}\}$ into the superposition of states of quantum CPU from electronic memory efficiently. In QLS, classical data a_0, a_1, \dots, a_{N-1} are used as control signals to flip the particles. QLS has time complexity $O(\log_2 N)$ and the I/O efficiency bottleneck of classical computer is broken by it.

the base of many quantum search algorithms. However, Grover's algorithm is invalid for this kind of search, we need to improve Grover's algorithm. Pang presents a general Grover iteration for the search case with additional computation [8, 9, 10, 11, 12, 13, 14], which is derived from the study of quantum image compression [8, 9, 10, 11, 12, 13, 14].

The Grover iteration is defined as [3, 4]

$$G = (2|\xi\rangle\langle\xi| - I) O_f \quad (1)$$

, where O_f is the oracle that flips the phase of state in Grover iteration, and $|\xi\rangle = \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} |i\rangle \right)$.

The **General Grover Iteration (GGI)** $G_{general}$ [8, 9, 10, 11, 12, 13, 14] is defined as

$$G_{general} = (2|\xi\rangle\langle\xi| - I) (U_L)^\dagger (O_c)^\dagger O_f O_c U_L$$

, where O_c denotes the other computation oracle for additional function f_c .

Fig.2 illu

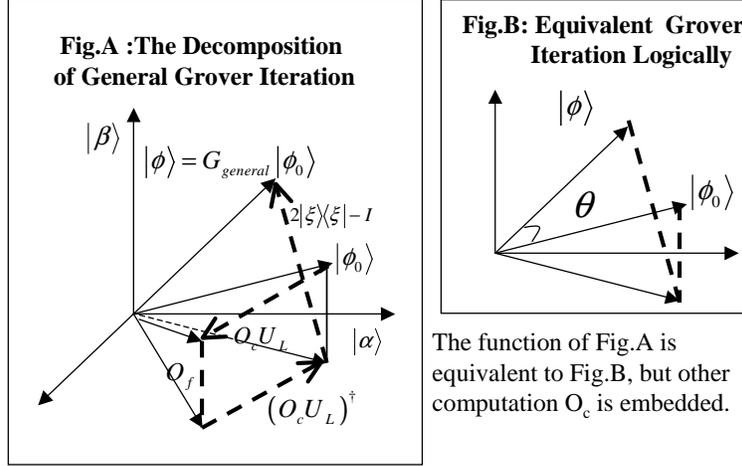


FIG. 2: The Illustration of General Grover Iteration

Similar to Grover iteration, $G_{general}$ act on initial state $|\xi\rangle = \frac{1}{\sqrt{N}} \left(\sum_{i=0}^{N-1} |i\rangle_{register1} |0\rangle_{register2} \right)$ $O(\sqrt{N})$ times and the solution will be found if the solution is unique [8, 9, 10, 11, 12, 13, 14].

Grover's algorithm is very useful, and many improved algorithms and many properties are studied by many experts [16, 17, 18, 19, 20]. Boyer, Brassard, Hoyer, and Tap present an improved algorithm named BBHT algorithm in this paper.

Suppose there are sequence of data $T[i]$ ($0 \leq i < N$). The various steps of the BBHT [16] are:

Step1. Initialize $\Gamma = 1$ and $\lambda = 6/5$ (Any value of λ strictly between 1 and $4/3$ would do.)

Step2. Choose j uniformly at random among the nonnegative integers not bigger than Γ .

Step3. Apply j iterations of Grover's algorithm starting from the state $|\Psi_0\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle$.

Step4. Observe register: let i_0 be the outcome.

Step5. If $T[i_0] = x$, the problem is solved, where $T[i]$ is the sequence of data. And **exit**.

Step6. Otherwise, set Γ to $\min\{\lambda\Gamma, \sqrt{N}\}$ and go to **step 2**.

The above BBHT algorithm is used to solve the case that the number of solutions t is unknown. BBHT algorithm requires that $1 \leq t \leq \frac{3}{4}N$. If $t > \frac{3}{4}N$, we can applied classical

full search method to get parts of solution efficiently, and call BBHT algorithm again. The case of no solution is handled by BBHT algorithm also.

BBHT algorithm has time complexity $O\left(\sqrt{\frac{N}{t}}\right)$. And the probability of finding a solution is bigger than $\frac{1}{2}$ (i.e., after repeating BBHT twice or more, a solution will be found with probability 100% approximately).

BBHT algorithm is the combination between quantum algorithm and classical iteration. And the benefit of this combination lies on many circuit are saved, while pure quantum algorithm will cost exponential numbers of circuit when the number of solution is unknown. BBHT algorithm will be used in this paper [4].

II. THE QUANTUM ALGORITHM FOR INTERSECTION OPERATION

A. Unitary Operation and Data Structure

Without losing generality, suppose that set is comprised by many vectors (or records) and let $A = \{\vec{a}_0, \vec{a}_1 \dots \vec{a}_{N-1}\}$, where $N = 2^n$ (otherwise, add special vector such that $N = 2^n$). As the same, we have set $B = \{\vec{b}_0, \vec{b}_1 \dots \vec{b}_{M-1}\}$, $M = 2^m$.

The match function between two vectors is defines as

$$f_c(\vec{a}_i, \vec{b}_j) = \begin{cases} 1 & \text{if } \vec{a}_i = \vec{b}_j \\ 0 & \text{otherwise} \end{cases}$$

The model of intersection operation $C = A \cap B$ is to find two records \vec{a}_{i_0} and \vec{b}_{j_0} such that $\vec{a}_{i_0} = \vec{b}_{j_0}$ (i.e., $f_c(\vec{a}_i, \vec{b}_j) = 1$). we have the following data structure and unitary operation for this model.

DS1. Save set A in electronic memory as a database, and each vector \vec{a}_i is a record with unique index i . As the same, save set B , and each vector \vec{b}_j has index j .

DS2. Construct five registers that have format

$$|i\rangle_{\text{register1}} |j\rangle_{\text{register2}} |\vec{a}_i\rangle_{\text{register3}} |\vec{b}_j\rangle_{\text{register4}} |f_c(\vec{a}_i, \vec{b}_j)\rangle_{\text{register5}}$$

That is, the 1st, 2nd, 3rd, 4th, 5th register is used respectively to save index i , index j , vector \vec{a}_i , vector \vec{b}_j and the value of match function f_c .

DS3. Initialize the five registers as zero value:

$$|0\rangle_{\text{register1}}|0\rangle_{\text{register2}}|0\rangle_{\text{register3}}|0\rangle_{\text{register4}}|0\rangle_{\text{register5}}$$

DS4. Construct Hadamard transform:

$$H : |0\rangle|0\rangle|0\rangle|0\rangle|0\rangle \longrightarrow \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{\text{ket}} |0\rangle|0\rangle|0\rangle \right)$$

,where each ket denotes a register, not a single qubit.

DS5. Construct Quantum Loading Scheme :

$$U_L : \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{\text{ket}} |0\rangle|0\rangle|0\rangle \right) \longrightarrow \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{\text{ket}} |\vec{a}_i\rangle|\vec{b}_j\rangle|0\rangle \right) \quad (2)$$

The function of U_L is to load data into entangled state from electronic memory according to index.

DS6. Design oracle O_c to compute the value of match function f_c :

$$O_c : \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{\text{ket}} |\vec{a}_i\rangle|\vec{b}_j\rangle|0\rangle \right) \longrightarrow \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{\text{ket}} |\vec{a}_i\rangle|\vec{b}_j\rangle |f_c(\vec{a}_i, \vec{b}_j)\rangle \right)$$

,where

$$f_c(\vec{a}_i, \vec{b}_j) = \begin{cases} 1 & \text{if } \vec{a}_i = \vec{b}_j \\ 0 & \text{otherwise} \end{cases}$$

DS7. Design oracle O_f :

$$O_f : \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{\text{ket}} |\vec{a}_i\rangle|\vec{b}_j\rangle |f_c(\vec{a}_i, \vec{b}_j)\rangle_{\text{register5}} \right) \longrightarrow$$

$$\frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (-1)^{f(\text{register5})} \underbrace{|i\rangle|j\rangle}_{\text{ket}} |\vec{a}_i\rangle|\vec{b}_j\rangle |f_c(\vec{a}_i, \vec{b}_j)\rangle_{\text{register5}} \right)$$

The above oracle O_f is the oracle in Grover's algorithm [3, 4], which flips the phase of state.

DS8. Construct General Grover Iteration G_{general} :

$$G_{\text{general}} = (2|\xi\rangle\langle\xi| - I) (U_L)^\dagger (O_c)^\dagger O_f O_c U_L \quad (3)$$

The function of operation G_{general} is equivalent to the Grover iteration (see Fig.2).

If set C has unique element (i.e., $|C| = 1$), $G_{general}$ acting on $|\Psi_0\rangle = \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{|i\rangle|j\rangle} |0\rangle|0\rangle|0\rangle \right) O(\sqrt{MN})$ times will generate intersection set. However, the case $|C| > 1$ often happens, where $|\cdot|$ denotes the size of set. Therefore, we must design other improved algorithm to compute all elements of set $C = A \cap B$.

B. Subroutine 1: Find An Element in Set $C = A \cap B$

Step1. Initial $\Gamma = 1$, $\lambda = \frac{6}{5}$.

Step2. Choose k uniformly at random among the nonnegative integers not bigger than Γ .

Step3. Apply k times of general Grover iteration $G_{general}$ starting from the state

$$|\Psi_0\rangle = \frac{1}{\sqrt{MN}} \left(\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \underbrace{|i\rangle|j\rangle}_{|i\rangle|j\rangle} |0\rangle|0\rangle|0\rangle \right) |ancilla\rangle.$$

Step4. Observe the first and second register: let i_0 and j_0 be the output.

Step5. If $\vec{a}_{i_0} = \vec{b}_{j_0}$, return result i_0 and j_0 , and exit.

Step6. Otherwise, set Γ to $\min\{\lambda\Gamma, \sqrt{MN}\}$ and go to **step2**.

Subroutine 1 is similar to BBHT algorithm, and the main different between the two algorithms is that Grover iteration is replaced by general Grover iteration $G_{general}$.

Similar to BBHT algorithm, we assume that $1 \leq |C| \leq \frac{3}{4}|A| \times |B|$ in this paper, where $|\cdot|$ denotes the size of set. If $|C| > \frac{3}{4}|A| \times |B|$, we can applied classical full search method to got parts of solutions efficiently, and call subroutine 1 again. Similar to BBHT, the case $A \cap B = \emptyset$ (i.e., empty set) is handled by subroutine 1.

Similar to BBHT algorithm, subroutine 1 has time complexity $O\left(\sqrt{\frac{|A| \times |B|}{|A \cap B|}}\right)$.

Similar to BBHT algorithm, the output of subroutine 1 is a solution or not a solution. And the probability that output is a solution is bigger than $\frac{1}{2}$, and the probability that output is not a solution is less than $\frac{1}{2}$. Repeating subroutine 1 twice, a solution will be obtained.

C. Quantum Search Algorithm for $C = A \cap B$ (Q-Intersection)

Step1. $C = \emptyset$ (i.e., empty set) and $nFlag = FALSE$; Save all elements of set A and B in a database, each element is a record. And the database is in electronic memory.

Step2. while ($nFlag = FALSE$)

{

Step2-1. Call subroutine 1 to find a solution $\vec{a}_{i_0} = \vec{b}_{j_0}$;

Step2-2. If there is no output from subroutine 1, $nFlag = TRUE$;

If $\vec{a}_{i_0} \notin C$, $C \leftarrow C \cup \{\vec{a}_{i_0}\}$. And update the database such that the records of vector \vec{a}_{i_0} and \vec{b}_{j_0} are different from all records and the two vectors are also different. Continue. (Notice: The case $\vec{a}_{i_0} \in C$ will not happen in next calling subroutine 1 because database is updated.)

}

Step3. Call subroutine 1 to find a solution again. If there is no output from subroutine 1, halt the algorithm. Otherwise, let $nFlag = FALSE$ and go to **step2**.

D. The Analysis of Time Complexity for Q-Intersection

Conclusion: Algorithm Q-Intersection has time complexity $O\left(\sqrt{|A| \times |B| \times |C|}\right)$, where $|\cdot|$ denotes the size of set.

Proof: Similar to BBHT algorithm, the case $|C| = 0$ is handled by this algorithm, and running time is $O(\sqrt{MN})$. The following discussion is under the condition $|C| \geq 1$.

Before firstly calling subroutine 1, there are $t = |C| = |A \cap B|$ numbers of unknown solutions. The sizes of set A and B are both constant during the whole calculation. Thus, the scale of problem is $t = |C|$. Suppose we need $I_t = I_{|C|}$ steps of computation to obtain all solutions. During the first computation of calling subroutine 1, $c\sqrt{\frac{MN}{|C|}}$ steps of computation are cost, where c denotes a constant, $|A| = N$ and $|B| = M$. The output of subroutine 1 is a solution or not a solution. After executing step2-1 (i.e., subroutine 1), two cases are happened. And the first case is that the output of subroutine 1 is a solution, and the second case is that the output is not a solution. The probability of the first case P_{case1} is bigger than $\frac{1}{2}$, while the probability of the second case P_{case2} is less than $\frac{1}{2}$. That is, there is a real number ε ($0 \leq \varepsilon \leq \frac{1}{2}$) such that $P_{case1} = \frac{1}{2} + \varepsilon$ and $P_{case2} = \frac{1}{2} - \varepsilon$. When the first case happen, the scale of problem becomes $t - 1 = |C| - 1$, and $I_{t-1} = I_{|C|-1}$ computation steps will be needed for all of remnant calculations. When the second case happen, the scale of problem is still $|C|$, and $I_t = I_{|C|}$ computation steps will be needed again.

When secondly calling subroutine 1, the situation is same. When the scale of problem

(i.e., the number of unknown solutions) is t , $c\sqrt{\frac{MN}{t}}$ steps of computation are cost for calling subroutine 1, and the number of remnant steps is $(\frac{1}{2} + \varepsilon)I_{t-1} + (\frac{1}{2} - \varepsilon)I_t$. That is,

$$I_t = c\sqrt{\frac{MN}{t}} + (\frac{1}{2} + \varepsilon)I_{t-1} + (\frac{1}{2} - \varepsilon)I_t$$

With the iterative computation increasing, the scale of problem t become smaller. When $t = 1$, $O(\sqrt{MN})$ steps of computation will be cost by subroutine 1, i.e., $I_1 = c_1\sqrt{MN}$, where c_1 is a constant.

The time complexity can be analyzed by the above way. Therefore, the following recursion equation is obtained to calculate time complexity:

$$\begin{cases} I_t = c\sqrt{\frac{MN}{t}} + (\frac{1}{2} + \varepsilon)I_{t-1} + (\frac{1}{2} - \varepsilon)I_t \\ I_1 = c_1\sqrt{MN} \\ 1 \leq t \leq |C|, 0 \leq \varepsilon \leq \frac{1}{2} \end{cases} \quad (4)$$

, where I_t denotes the number of computation steps when there are t number of unknown solution.

By Eq.4, we have

$$I_t - I_{t-1} = 2c\frac{\sqrt{MN}}{\sqrt{t}} - 2\varepsilon(I_t - I_{t-1}) \quad (5)$$

Performing $(I_{|C|} - I_{|C|-1}) + (I_{|C|-1} - I_{|C|-2}) + \dots (I_2 - I_1)$, we have

$$I_{|C|} - I_1 = 2c\sqrt{MN}\left(\sum_{i=2}^{|C|} \frac{1}{\sqrt{i}}\right) - 2\varepsilon(I_{|C|} - I_1)$$

Because $I_{|C|} \geq I_1$ and $\varepsilon \geq 0$, we have

$$I_{|C|} - I_1 \leq 2c\sqrt{MN}\left(\sum_{i=2}^{|C|} \frac{1}{\sqrt{i}}\right)$$

That is,

$$I_{|C|} \leq 2c\sqrt{MN}\left(\sum_{i=2}^{|C|} \frac{1}{\sqrt{i}}\right) + I_1$$

We have

$$I_{|C|} \leq 2c\sqrt{MN} \left(\int_1^{|C|} \frac{1}{\sqrt{x}} dx \right) + I_1$$

$$I_{|C|} \leq 4c\sqrt{|C|MN} + (c_1 - 4c)\sqrt{MN}$$

i.e.,

$$I_{|C|} \leq \left(4c + \frac{c_1 - 4c}{\sqrt{|C|}} \right) \sqrt{|C|MN}$$

Because $|C| \geq 1$, we have

$$\begin{cases} I_{|C|} \leq c_1\sqrt{|C|MN} & \text{if } c_1 \geq 4c \\ I_{|C|} \leq 4c\sqrt{|C|MN} & \text{if } c_1 < 4c \end{cases}$$

Thus, there is a constant $\lambda > 0$ such that

$$I_{|C|} \leq \lambda\sqrt{|C|MN}$$

That is,

$$I_{|C|} = O(\sqrt{|A| \times |B| \times |C|}) \quad (6)$$

Formula 6 shows that Q_Intersection algorithm has time complexity $O(\sqrt{|A| \times |B| \times |C|})$. That is, only $O(\sqrt{|A| \times |B| \times |C|})$ steps of computation are needed to calculate intersection set $C = A \cap B$, while $O(|A| \times |B|)$ steps are needed for classical computation. The quantum algorithm Q_Intersection is fast than classical method by $O(\sqrt{\frac{|A| \times |B|}{|C|}})$ factors.

In addition, the probability of BBHT algorithm to find a solution is bigger than $\frac{1}{2}$. Similar to BBHT algorithm, The successful probability of subroutine 1 is bigger than $\frac{1}{2}$. That is, calling subroutine 1 twice can find a solution with 100% probability approximately. Step2 and step 3 of Q_Intersection algorithm guarantee the successful probability is close to 100% approximately.

Because $A \cup B = I - A \cap B$, the presented algorithm can be used to calculate union operation also. In sum, using the method of Q_Intersection to perform set operation is possible.

III. CONCLUSION

The set operations, such as intersection, union and complement, are the fundamental calculation in mathematics. Set operation is the base of many sciences and techniques, such as database, signal processing and image processing. E.g., database is based on set operation. Designing fast algorithm for set operation is significant.

Full search method is the common method of set operation in general because sorting multi-dimensional data is not very useful to improve running speed. Full search has time complexity $O(|A| \times |B|)$ for the intersection $C = A \cap B$, which is very slow still when the size of set is huge. Electronic computer loads data into register *one by one* from memory, and the efficiency bottleneck is formed.

In this paper, the quantum search algorithm for intersection operation of set (named Q_Intersection) is presented, which is the combination of Grover's algorithm, classical memory, classical iteration. Using the method of Q_Intersection, the quantum algorithms for other set operations can be designed also.

The advantages of Q_Intersection are listed as below.

1. Q_Intersection has time complexity $O\left(\sqrt{|A| \times |B| \times |C|}\right)$, while classical algorithm has time complexity $O(|A| \times |B|)$, where $|\cdot|$ denotes the size of set. Q_Intersection is fast than classical method by $O\left(\sqrt{\frac{|A||B|}{|C|}}\right)$.

2. All information of data can be loaded into quantum state by $O(\log_2 |A| |B|)$ steps of computation in Q_Intersection, and all data is loaded at a same time, while classical computer can only load data one by one and $O(|A| |B|)$ steps are needed. And the efficiency bottleneck of electronic computer is evaded.

3. In step2-2 of Q_Intersection, the data in the electronic memory is often updated according to the output of quantum algorithm, which simplifies the design of quantum algorithm. As well known, data in the superposition of states can not be updated as a given number and can not be measured when unitary operation acting on this superposition. This defect make designing quantum algorithm very difficult. Q_Intersection shows that the combination between quantum algorithm and classical memory is useful to decrease the complexity of designing quantum algorithm.

Acknowledgments

The first author thanks his teacher prof. G.-C. Guo because the author's main study methods are learned from his lab. The first author thanks prof. Z. F. Han's for he guiding the author up till now. The first author thanks prof. J. Zhang and prof. Z.-L. Pu for their help.

-
- [1] Suo-Fei Tang, Computer Organization and Architecture, 2nd ed., Higher Education Press, Beijing, 2000.
 - [2] P. W. Shor, Algorithms for quantum computation discretelog and factoring, Proc. of the 35th Annual Symposium on the Foundations of Computer Science, Los Alamitos, 1994, pp.20-24.
 - [3] L.K. Grover,Afast quantum mechanical algorithm for database search, in: Proc. of the Twenty-EighthAnnual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, May 1996, pp. 212C219.
 - [4] M. A. Nielsen, I. L. Chuang, Quantum Computationand and Quantum Information, 1st ed., Cambridge University Press, Cambridge, England, 2000
 - [5] L. M. K. Vandersypen, M. Steffen, G. Breytal, C. S. Yannonil, M. H. Sherwood, I. L. Chuang, Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance, Information and computation: Classical and quantum aspects, Nature, 414 (2001) 883-887.
 - [6] Chao-Yang Lu, Daniel E. Browne, Tao Yang, Jian-Wei Pan, Demonstration of a Compiled Version of Shor's Quantum Factoring Algorithm Using Photonic Qubits, Phys. Rev. Lett. 99 (2007) 250504
 - [7] Jose I. Latorre, Iimage compression and entanglement, 2005, arXiv:quant-ph/0510031,
 - [8] Chao-Yang Pang, Quantum image compression, Postdoctoral report, Key Laboratory of Quantum Information, Uni. of Science and Technology of China (CAS), Hefei,China, Jun 2006
 - [9] Chao-Yang Pang, Zheng-Wei Zhou, Guang-Can Guo, A hybrid quantum encoding algorithm of vector quantization for image compression, Chin. Phys. 15 (2006) 3039-3043
 - [10] Chao-Yang Pang, Zheng-Wei Zhou, Ping-Xing Chen, Guang-Can Guo, Design of quantum vq iteration and quantum vq encoding algorithm taking \sqrt{n} steps for data compression,

Chin. phys. 15 (2006) 618-623

- [11] C. Y. Pang, Loading N-Dimensional Vector into Quantum Registers from Classical Memory with $O(\log N)$ Steps, 2006, arXiv:quant-ph/0612061
- [12] Chao-Yang Pang, Cong-Bao Ding, Ben-Qiong Hu, Quantum pattern recognition of classical signal, 2007, arXiv: quant-ph/0707.0936
- [13] Chao-Yang Pang and Ben-Qiong Hu, Quantum discrete fourier tranform with classical output for signal processing, 2007, arXiv:quant-ph/0706.2451
- [14] Chao-Yang. Pang, Zheng-Wei Zhou, and Guang-Can Guo, Quantum discrete cosine transform for image compression, 2006, arxiv:quant-ph/0601043
- [15] V. Giovannetti, S. Lloyd, and L Maccone, Quantum random access memory, 2007, arXiv:quant-ph/0708.1879
- [16] M. Boyer, G. Brassard, P. Hoyer, A. Tap, Tight bounds on quantum searching, Fortsch. Phys. 46 (1998) 493-506
- [17] C. Durr, and P. Hoyer, A Quantum Algorithm for Finding the Minimum, 1996, arXiv:quant-ph/9607014
- [18] G. L. Long, Grover algorithm with zero theoretical failure rate, Phys. Rev. A 64 (2001) 022307
- [19] E. Biham, O. Biham, D. Biron, M. Grassl, and D. A. Lidar, Grover's quantum search algorithm for an arbitrary initial amplitude distribution, Phys. Rev. A 60 (1999) 2742-2745
- [20] E. Biham, O. Biham, D. Biron, M. Grassl, D. A. Lidar, and D. Shapira, Analysis of generalized Grover quantum search algorithm using recursion equations, Phys. Rev. A 63 (2000) 012310