Quantum Algorithms for Some Well-Known NP Problems

Commun. Theor. Phys. (Beijing, China) **37** (2002) pp 424–426

Vol. 37, No. 4, April 15, 2002

Quantum Algorithms for Some Well-Known NP Problems

GUO Hao,[1] LONG Gui-Lu,[1−5] and LI Feng[6]

[1]Department of Physics, Tsinghua University, Beijing 100084, China

[2]Institute of Theoretical Physics, the Chinese Academy of Sciences, Beijing 100080, China

[3]Centre for Nuclear Theory, Lanzhou National Laboratory of Heavy Ions, the Chinese Academy of Sciences, Lanzhou 730000, China

[4]The Key Laboratory of Quantum Information and Measurements, MOE, Beijing 100084, China

[5]Center of Atomic and Molecular Nanosciences, Tsinghua University, Beijing 100084, China

[6]Basic Education Section, Capital University of Economics and Business (West Campus), Beijing 100070, China

(Received August 14, 2001)

Abstract *It is known that quantum computer is more powerful than classical computer. In this paper we present quantum algorithms for some famous NP problems in graph theory and combination theory, these quantum algorithms are at least quadratically faster than the classical ones.*

PACS numbers: 03.67.Lx, 89.70.+c
Key words: quantum algorithms, NP problem, graph theory, combination theory

1 Introduction

The basic conceptions of the quantum computer were first presented by Feynman.[1] Since Shor gave the famous quantum algorithm for the factoring problem in 1994,[2] the study of quantum algorithm becomes very hot. In the classical computation theory, the study of the algorithms for the NP-complete (NPC) problem is one important theme. At present, Shor's factorization algorithm is polynomial. But the nature of the factorization problem, whether NPC or not, is not yet clear. It is sometimes called NP intermediate (NPI). On the other hand, Feynman's anticipation that quantum computer is powerful in simulating quantum system is confirmed. Recently, it has been found that quantum computer can solve quantum chaos and localization problems with exponential gain compared with classical algorithms.[3] The SAT problem is the first discovered NP-complete problem, all other NP problems can be converted to the SAT problem in polynomial time in a Turing machine. If we find an efficient quantum algorithm for the SAT problem, then all NP problems can be ***solved*** by the quantum computer efficiently. Masanori and Masuda[4] discussed a quantum algorithm for the SAT problem. The algorithm can solve the problem in polynomial steps. However the algorithm depends on a stringent assumption that the state of a qubit can be physically distinguishable. This assumption is very difficult to be realized because when the qubit is in a superposition, the distinguishability of the state is very difficult. Nevertheless this assumption can be dropped by using a quantum search algorithm[5] so that the amplitudes of the desired state can be amplified, and this has been done for the Hamiltonian circuit problem.[6] Though it is not exponentially faster than classical algorithm, it is still faster than classical computers. Such the algorithms are still a significant improvement. The wide ranges of such the problems make them important. In this paper we will give quantum algorithms for some famous NP problems.

2 The SAT and Some NP Problems

Let us briefly review the SAT problem and other NP problems. Let $X \equiv \{x_1, x_2, \ldots, x_n\}$ be a set. x_k and its negation $\bar{x}_k$ $(k = 1, 2, \ldots, n)$ are called literals. Let X' denote the set $\{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$. The power set of X' is $2^{X'}$ and a subset C of $2^{X'}$ is called a clause. A truth assignment of X is a function $t : X \to \{1, 0\}$, where 1 denotes "true" and 0 denotes "false". A truth assignment t of X makes a clause C satisfiable iff at least one literal of C is true under t.

C is satisfiable iff $t(C) = 1$, and the truth value of C is written as $t(C) \equiv \bigvee_{x \in C} t(x)$. Moreover the set C' of clauses C_j $(j = 1, 2, \ldots, m)$ is called satisfiable iff the meet of all truth values of C_j is 1, that is to say

$$t(C') \equiv \bigwedge_{j=1}^{m} C_j = 1\,.$$

We can write the SAT problem as follows.

Given now a set $X = \{x_1, \ldots, x_n\}$ and a set $C' = \{C_1, \ldots, C_m\}$ of clauses, determine whether there exists a truth assignment to make C' satisfiable.

The details of the quantum algorithm for the SAT problem can be found in Ref. [4], the essence of the SAT problem is to determine whether the following Boolean logical expression can be satisfied by a truth assignment,

$$\wedge_{j=1}^{m}(\vee_{x \in C_j} x)\,. \tag{1}$$

When we design a quantum algorithm for some NP problems, a natural idea is to determine the satisfiability of an expression similar to Eq. (1). We will discuss the quantum algorithms for the following NP problems in the graph theory and combination theory:

1) The minimum vertex covering problem (VC);
2) The minimum edge covering problem (EC);
3) The hitting set problem (HS);
4) The independent set problem (IND);
5) The clique problem;
6) The set packing problem;
7) The set covering problem (SC);
8) The triplet exactly covering problem (3XC);
9) The 3-dimension matching problem (3DM);

Their quantum algorithms are similar, all can be constructed by exploiting the SAT algorithm and Grover's searching algorithm. So we will just discuss the algorithm for the VC in details and others briefly.

3 Quantum Algorithms for Some NP Problems

3.1 *The Minimum Vertex Covering Problem*

Let $G = (V, E)$ be a simple graph, V is the vertex set and E is the edge set. If there exists a vertex subset C which satisfies $C \subseteq V$ and that if for every edge $e = (u, v)$ of graph G we have $\{u, v\} \cap C \neq \emptyset$, where u, v are the two vertex of edge e, then C is called a vertex covering set of G. That means at least one vertex of each edge of the graph G is in set C. The VC problem is stated as follows. Given a graph $G = (V, E)$ and a positive integer $k \leq |V|$, whether G has a vertex covering set C such that $|C| \leq k$?

The quantum algorithm is as follows. Let $n = |V|, m = |E|$, so there are n vertexes and m edges. Define n "vertex" Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$, corresponding to the n vertexes of the graph G, so X can be viewed as denoting the vertex set V of G. And then for each edge of G, we define an "edge" clause $C_i = \{x_{i_1}, x_{i_2}\}, i = 1, 2, \ldots, m$, where x_{i_1} and x_{i_2} denote the two vertexes of the edge C_i. Let $C' = \{C_1, C_2, \ldots, C_m\}$. We can consider the vertex covering set C as a truth assignment on X. We set that if $x_i \in C$, then $t(x_i) = 1$. The truth value of clause C_i is $t(x_{i_1}) \vee t(x_{i_2})$. So whether G has a vertex covering set C is equivalent to whether there exists a truth assignment on X which makes

$$t(C') = \wedge_{i=1}^{m} t(C_i) = \wedge_{i=1}^{m} (t(x_{i_1}) \vee t(x_{i_2})) = 1\,,$$

which is a special case of the SAT problem. Certainly, C must satisfy $|C| \leq k$.

According to the above analysis, we are able to design the quantum algorithm. We need a quantum computer with $n + h + l + 1$ qubits, where the first n qubits denote n vertex Boolean variables, the second h qubits are used for counting, the next l qubits are dust qubits which are necessary by exploiting the SAT algorithms, the last one qubit is used to output the result.

1) Initially we set all qubits on 0. Performing the Hadamard transformation on the first n qubits, we get

$$\begin{aligned} |\psi\rangle &= \frac{1}{\sqrt{2^n}} \sum_{x_1, x_2, \ldots, x_n = 0}^{1} \otimes_{i=1}^{n} |x_i\rangle \otimes^{h+l} |0\rangle \otimes |0\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n - 1} |x\rangle \otimes^{h+l} |0\rangle \otimes |0\rangle\,. \end{aligned} \tag{2}$$

Thus all possible truth assignments appear in the first n qubits in the superposition.

2) We request that $|C| \leq k$, while at most n qubits can be on $|1\rangle$, so we must remove the states with more than k $|1\rangle$. We perform the following loop.

For $j = 1$ to n,

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x_1, x_2, \ldots, x_n = 0}^{1} |x_1, x_2, \ldots, x_n\rangle |s = s + x_j\rangle \otimes_1^l |0\rangle |0\rangle.$$

For $j = j + 1$, $s = 0$ initially, and it is obviously $h \leq \log_2(x)$. Then carry addition operator is used and it can be implemented by quantum elementary gates. Let s_x denote the number of $|1\rangle$ in $|x\rangle = |x_1, x_2, \ldots, x_n\rangle$, so $|\psi\rangle$ becomes

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n - 1} |x\rangle |s_x\rangle \otimes_1^l |0\rangle \otimes |0\rangle\,.$$

3) We hope to eliminate the components with $S_x > k$, so we use Grover's searching algorithm on the h counting qubits to amplify the amplitude of the components we needed. It will take $O(2^{h/2}) = O(\sqrt{n})$ repetitions. After this step, we get

$$|\psi\rangle = \frac{1}{\sqrt{C_n^k}} \sum_{x=0}^{2^n - 1} |x\rangle |s_x \leq k\rangle \otimes_1^l |0\rangle \otimes |0\rangle\,.$$

4) Using the SAT algorithm to compute the C_n^k inputs, the function we compute is

$$f(x_1, x_2, \ldots, x_n) = \wedge_{i=1}^{m} t(C_i) = \wedge_{i=1}^{m} (t(x_{i_1}) \vee t(x_{i_2}))\,.$$

We can use elementary quantum gates to construct a quantum circuit to implement the function. And we get

$$|\psi\rangle = \frac{1}{\sqrt{C_n^k}} \sum_{x=0}^{2^n - 1} |x\rangle |s_x \leq k\rangle \otimes_1^l |y_x\rangle \otimes |f(x)\rangle\,. \tag{3}$$

The $|y_x\rangle$ is dust output.

5) Finally we should determine whether there exists $f(x) = 1$ by using Grover's searching algorithm. We need to measure the result qubit. The time needed for this step is $O(\sqrt{C_n^k})$.

Thus we finished the algorithm. At the worst case, the algorithm is not a polynomial one, but it is still quadratically faster than the classical one.

3.2 *Quantum Algorithms for Other NP Problems*

We discuss the quantum algorithms for the other NP problems. The EC problem, the IND problem and the HS problem are simple and similar to the VC problem. Readers can easily give quantum algorithm for them by the ideas of the above algorithm, so we do not discuss them in details. The CLIQUE problem, the SP problem and the SC problem are a little bit more difficult, but the basic ideas are the same. We will analyze the most complicated problem, the 3XC problem and the 3DM problem.

The 3XC problem Given a set $U = \{u_1, u_2, \ldots, u_{3n}\}$ with $3n$ elements, let 2^{3U} denote all the subsets of U with exact three elements. $C \subseteq 2^{3U}$, the question is whether there exists a subset $C' \subseteq C$ such that C' exactly covers U ($|C'|$ must be n at this case). That is, if $C' = \{S_1, S_2, \ldots, S_n\}$, then $S_1 \cup S_2 \cup \cdots S_n = U$, and for any $i \neq j, S_i \cap S_j = \emptyset$.

Algorithm Let $m = |C|, C = \{S_{i_1}, S_{i_2}, \ldots, S_{i_m}\}$. Define m Boolean variables $x_{i_1}, x_{i_2}, \ldots, x_{i_m}$ corresponding to $S_{i_1}, S_{i_2}, \ldots, S_{i_m}$ respectively. Define the clause $C_{ij} = \{x_i, x_j | S_i \cap S_j \neq \emptyset\}$. We let $t(x_i) = 1$ iff $S_i \in C'$. The question becomes to find a C' such that $\vee_{C_{ij}}(t(x_i) \wedge t(x_j)) = 0$, and the number of "1" among x_i is exactly n. The next procedures are the same as the VC algorithm.

The 3DM problem W, X, Y are three sets which do not intersect each other, and $|W| = |X| = |Y| = q$, $M \subseteq W \times X \times Y$. The question is whether there exists $M' \subseteq M$ such that M' is a perfect pair set, namely each pair of elements of M' has no common components. We can see that $|M'|$ must be q.

Algorithm Let $n = |M|$, define n Boolean variables $x_1, x_2, \ldots, x_n$ corresponding to each element of M respectively. Define the clause $C_{ij} = \{x_i, x_j | x_i, x_j \in M$ and x_i, x_j have common components$\}$. Let $S = \{C_{ij} |$ for all $i, j\}$. We set $t(x_i) = 1$, iff $x_i \in M'$. Then the question if there exists a perfect pair set is equivalent to that if there exits a truth assignment which makes $M = \{x_1, x_2, \ldots, x_n\}$ have exactly q "1" and that $\vee_{C_{ij} \in S}(t(x_i) \wedge t(x_j)) = 0$. The other details are similar to those of the VC algorithm.

4 Summary

We have present quantum algorithms for one kind of NP problems. These algorithms are quadratically faster than classical algorithms. These quantum algorithms can be modified. We can only make some mathematical technique to change slightly to solve other NP problems. Though these algorithms are not polynomial, they are still significantly faster than the classical ones. In fact, except simulating properties of quantum systems, it is still an open question whether quantum computer can solve all NP problems polynomially. From our discussions we see that the bottleneck rests wholly on the problem if we can find a polynomial quantum search algorithm. At the moment, it seems that such a hope is illusive as it has been shown that Grover's quantum search algorithm is the optimal one though some improvements in the successful probability need be made.[7]

References

[1] R. Feynman, Opt. News **11** (1985) 11; Found. Phys. **16** (1986) 507.

[2] P. Shor, *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, IEEE Computer Society Press, Los Alamos, CA (1994) p. 124.

[3] B. Georgeot and D.L. Shepelyansky, Phys. Rev. Lett. **86** (2001) 2890.

[4] Masanori Ohya and Natsuki Masuda, NP Problem in Quantum Algorithm, Open Systems and Information Dynamics **7** (2000) 33.

[5] L.K. Grover, Phys. Rev. **79** (1997) 325.

[6] H. GUO, G.L. LONG, Y. SUN, and X.L. XIU, Commun. Theor. Phys. (Beijing, China) **35** (2001) 385.

[7] G.L. Long, Phys. Rev. **A64** (2001) 022307, also available in quant-ph/0106071.