Quantum Computing Beyond Integer Factorization—Exploring the Potential of Quantum Search

Seiichiro Tani

Abstract

Quantum computers are novel computing devices that make effective use of quantum-mechanical phenomena, and they are expected to emerge within the next few decades. Extensive studies on them have revealed that they will be able to solve many problems significantly faster than current computers or their possible extensions. This article briefly outlines the theory and applications of quantum search to show the potential wide applicability of quantum computing.

Keywords: quantum computers, quantum algorithms, quantum communication

1. Introduction

Significant progress in computer technology has been made since Alan M. Turing, a British mathematician, invented the mathematical model of computers. State-of-the-art computers still follow Turing's model in principle. The term *classical computers* refers to all current computers and their possible extensions, which are based on Turing's model, and *classical computation/algorithms* refers to the computation/algorithms performed on classical computers.

Quantum computers are novel computing devices that make effective use of quantum-mechanical phenomena, so their mechanism is quite different from Turing's model (**Table 1**). It is therefore expected that quantum computers, which are being extensively studied all over the world, will be able to solve a variety of problems significantly faster than the intolerably long time it takes to solve them on classical computers. To solve problems on quantum computers, we need to write a series of explicit operations to be performed on the hardware of quantum computers, as in the case of current computers. Such a series of operations is called a *quantum algorithm*. The computation time on quantum computers heavily depends on the quantum algorithms, just as the computation time of current computers depends on the classical algorithms. It is therefore necessary to research and develop fast quantum algorithms as well as scalable and robust hardware for quantum computers.

2. Fast quantum algorithms developed by Shor and Grover

The most famous quantum algorithm is arguably the one developed by Peter W. Shor in 1994 [1] that factors integers exponentially faster than any known classical algorithm. The integer factoring problem is a basic mathematical problem that is very difficult in the sense that a long history of research on this problem has not yet succeeded in finding a fast classical algorithm. In fact, the security of the RSA (Rivest-Shamir-Adelman) cryptosystem, used in practice on the Internet for secure data transmission, is based on the hardness of this problem. Shor's discovery thus has the potential to affect a great number of people, even those outside academic communities, since it implies that quantum computers would be capable of breaking the cryptosystem. This is obviously an unhappy scenario.

Extensions of Shor's algorithm can solve hidden

Table 1. Comparison between classical and quantum computers.

	Classical computers	Quantum computers
Unit of information	Bit	Quantum bit (qubit)
Mathematical expression of information	Logical values (True/False)	Complex vectors
Elementary operations	Logical operations (AND, OR, NOT)	Linear operators (unitary operators)
Model of computation	Turing machine, logic circuits	Quantum Turing machine, quantum circuit

subgroup problems extremely quickly. In the research field of quantum algorithms, subsequent studies extended Shor's algorithm to a collection of more general mathematical problems. Although these extensions are highly important in a theoretical sense, they seem to have little relation to the problems that are familiar to people outside the theory field, and it would thus be unlikely that those people would appreciate the extended algorithms.

The quantum search algorithm (*quantum search* in short), developed by Lov K. Grover [2] in 1996, is also very well known. This algorithm solves the problem of finding a desired piece of data from among N pieces of data. Classical computers clearly need roughly N accesses to the data in the worst case, even if we allow a small error probability. However, the quantum search can find a desired piece of data with high probability only with approximately $\sqrt{N}$ accesses. Although this cannot achieve the exponential speedups over classical algorithms, in contrast to the case of factoring integers, the quantum search can still yield a significant speedup whenever N is very large.

One of the advantages in considering the search problem is that the definition of the problem is so simple that it can often emerge as a subproblem of various other problems that we want to solve. In other words, the algorithms for the problem potentially have wide applicability. One can imagine the following straightforward scenario; if one finds a search problem as a subproblem of some other problem, then one can solve the search problem significantly faster with the quantum search, which may imply a fast algorithm for the whole problem. In many cases, however, it is not an easy task to carve a search problem out of the original problem; even if one succeeds in doing so, it may be necessary to adapt the algorithm appropriately, often in a non-trivial way. This is why the search problem and its generalizations are still major topics in quantum computing research (**Fig. 1**).

3. Quantum bits and superposition

The unit of information on quantum computers is called a quantum bit, or a *qubit*. A qubit represents any superposition of 0 and 1, including 0 or 1 as a special case (**Fig. 2**). Similarly, two qubits represent any superposition of 00, 01, 10, and 11. Moreover, n qubits represent any superposition of 2^n bit-strings: 0…0 through 1…1. The superposition is called the *quantum state* over the qubits. In terms of linear algebra, we can describe it as follows: Consider a two-dimensional complex Euclidean space with two orthogonal basis vectors, $\vec{0}$ and $\vec{1}$, of unit length, and identify $\vec{0}$ and $\vec{1}$ with "0" and "1," respectively. Then, a superposition of 0 and 1 means simply a linear combination of the unit vectors $\alpha\vec{0} + \beta\vec{1}$, where α,β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. Similarly, a superposition over n qubits is a linear combination of 2^n orthogonal vectors of unit length. Since qubits are identified with vectors, the operations over qubits should be mappings over vectors. More concretely, the operations must be unitary operators[*1] or orthogonal projectors, which are linear transformations over complex vectors. In particular, applying a set of orthogonal projectors summed to the identity is called *measurement*, which produces a classical outcome and a quantum state. To get a classical value as the output of a quantum algorithm, we thus need to take measurements over a quantum state generated by the algorithm.

4. The search problem and quantum algorithms

To solve a problem such as the search problem that depends on N input data $X_1, \cdots, X_N$, we need to access the input data. Formally, we can think of this as follows:

*1 A unitary operator is a linear operator that maps a vector to another such that it does not change the inner product of any two vectors. Intuitively, the operator does not change the angles between any two vectors and the length of any vector.

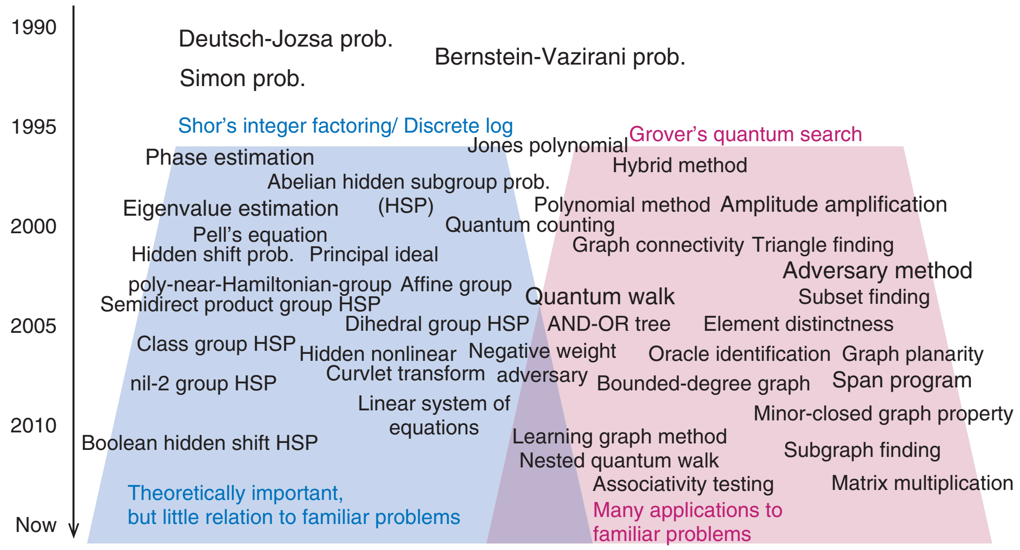


Fig. 1. Quantum algorithms.

	Superpositon	Mathematical expression
one qubit	Superposition of {0,1}	A vector in 2-dimensional complex Euclidean space
two qubits	Superposition of {00,01,10,11}	A vector in 4-dimensional complex Euclidean space
n qubits	Superposition of 2^n strings of {0…00,0…01,…,1…11}	A vector in 2^n-dimensional complex Euclidean space

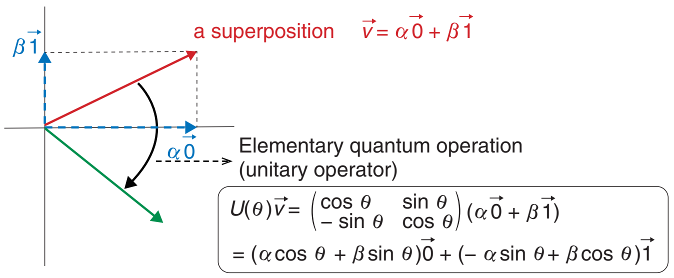

a superposition $\vec{v} = \alpha\vec{0} + \beta\vec{1}$
$\beta\vec{1}$
$\alpha\vec{0}$
Elementary quantum operation (unitary operator)
$U(\theta)\vec{v} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} (\alpha\vec{0} + \beta\vec{1})$
$= (\alpha\cos\theta + \beta\sin\theta)\vec{0} + (-\alpha\sin\theta + \beta\cos\theta)\vec{1}$

Fig. 2. Qubits and superposition.

We receive X_k by making a query with the index k. We call such a query a *classical query*. On quantum computers, the index associated with a query is expressed with qubits, and thus a query, in this case called a *quantum query*, will be a superposition of classical queries over all indices k ranging from 1 to N; accordingly, the answer to the quantum query will be the corresponding superposition of all X_k, where k ranges from 1 to N. With the ability to make quantum queries, the quantum search may be stated as follows (we assume for simplicity that each piece of input data is either 0 or 1, but this is not essential to the quantum search).

Theorem (Quantum Search) *Given N input data* $X_1, \cdots, X_N \in \{0,1\}$, *there exists a quantum algorithm that finds an index i with* $X_i = 1$ *with high probability by making approximately* $\sqrt{N}$ *data accesses (i.e., quantum queries).*

To estimate the total number of steps required to solve a problem, it is necessary to count the number of steps taken to process the input data obtained via queries, as well as the number of accesses to the input data. However, we will focus only on the number of accesses to the input data, since it is a dominant factor in the search problem and the other problems dealt with in this article.

As an application of the theorem, let us consider the problem of testing the planarity of a given graph.

Example 1: Graph planarity testing

We say that a graph is planar if the graph can be drawn without crossing any pair of edges on a two-

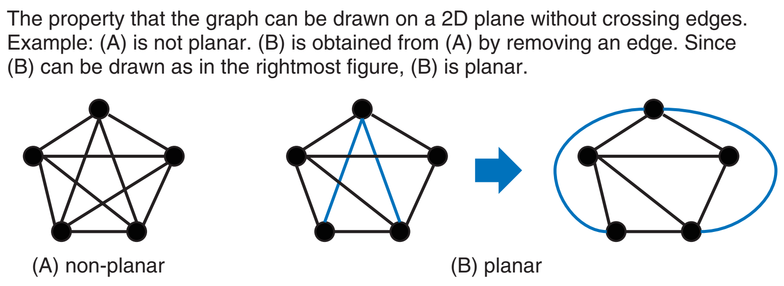

The property that the graph can be drawn on a 2D plane without crossing edges.
Example: (A) is not planar. (B) is obtained from (A) by removing an edge. Since (B) can be drawn as in the rightmost figure, (B) is planar.
(A) non-planar
(B) planar

Fig. 3. Planarity of a graph.

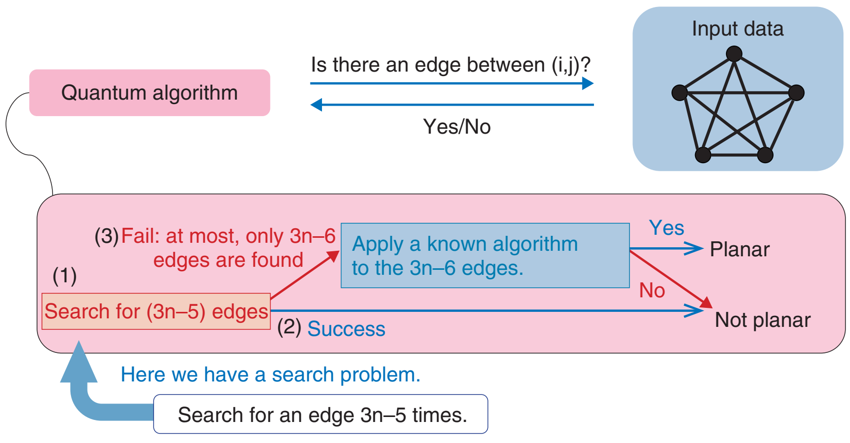

Input data
Is there an edge between (i,j)?
Yes/No
Quantum algorithm
(3) Fail: at most, only 3n–6 edges are found
Apply a known algorithm to the 3n–6 edges.
Yes
Planar
No
(1)
Search for (3n–5) edges
(2) Success
Not planar
Here we have a search problem.
Search for an edge 3n–5 times.

Fig. 4. Quantum algorithm for testing planarity of a graph.

dimensional plane (**Fig. 3**). It is well known that a lot of graph-theoretic problems that are hard for general graphs can be solved efficiently on planar graphs. Testing the planarity of graphs is hence one of the particularly important problems in testing graph properties. Let us define our problem more formally. We assume that graph G is an undirected graph[*2] consisting of n vertices. The input data are given as the information consisting of whether an edge exists between each pair of vertices, more concretely, the set of X_{ij} defined as follows; For each $i = 1, \cdots, n$ and $j = 1, \cdots, n$ with $i < j$, define $X_{ij} = 1$ if there is an edge between (i, j), and $X_{ij} = 0$ otherwise (that is, X_{ij} is the (i, j)-element of the adjacency matrix[*3] of G). The problem is determining with high probability if the graph represented by these $\frac{1}{2}n(n-1)$ pieces of data, X_{ij}, is planar. The key tool for carving out a search problem is a very old theorem discovered by Euler in the 18th century.

Theorem: *If a graph with n vertices is planar, then the graph has at most* $3n - 6$ *edges.*

With this theorem, we divide the problem into the following subproblems (**Fig. 4**).

(1) Identify $3n - 5$ edges from among the $\frac{1}{2}n(n-1)$ candidates of edges (in particular, this means identifying all edges, if the graph has at most $3n - 6$ edges).
(2) If Step (1) identifies exactly $3n - 5$ edges, the graph is not planar according to Euler's theorem, since $3n - 5 > 3n - 6$.
(3) If Step (1) identifies at most $3n - 6$ edges, all edges have been identified; then, without further queries, determine whether the graph is planar with a known algorithm.

Note that only Step (1) makes queries. With a little

*2 An undirected graph is a graph whose edges have no directions.
*3 For a graph with n vertices, the adjacency matrix A of the graph is an n-by-n matrix such that each element A[i,j] represents the existence of an edge between the vertices i and j.

more thought, it is possible to see that Step (1) can be realized by solving $3n - 5$ times the problem of searching for an edge. We can hence speed up Step (1) with the quantum search (plus some modifications). Consequently, we can prove that roughly $n^{1.5}$ quantum queries are sufficient to test the planarity of a graph on quantum computers, while nearly n^2 classical queries are required on classical computers [3]. We should emphasize that the key to carving out a search problem is the effective use of a theorem in the field of graph theory. There are many other important properties for which quantum speedups can be achieved, examples of which include testing graph connectivity[*4] and determining the existence of Hamiltonian paths[*5].

5. A generalization of quantum search

Let us sample a piece of input data at random on a classical computer. In other words, we choose a piece of data from among $X_1, \cdots X_N$ uniformly at random and access it. If there is exactly one desired piece of data among the N input data, the probability that a randomly chosen piece is the desired one is only $1/N$. In fact, very roughly speaking, what the quantum search does is to iterate this random sampling *in superposition* $\sqrt{N}$ times. Note that amplifying the success probability $1/N$ of the random sampling nearly to one requires roughly N times on classical computers. In this sense, we would say that the quantum search amplifies the success probability nearly to one with substantially fewer iterations. This viewpoint can be stated more generally as follows.

Theorem (Quantum Amplitude Amplification [4])
Consider a classical algorithm that solves a problem with probability p by accessing input data c times. Then, it is possible to construct a quantum algorithm that solves the problem with a probability close to one by accessing input data roughly $c/\sqrt{p}$ times. Moreover, if the exact value of p is known, it is possible to amplify the probability to one.

In the case of random sampling for the search problem, the parameters will be $c = 1$, $p = 1/N$. The resulting quantum algorithm thus accesses the input data (roughly) $c/\sqrt{p} = \sqrt{N}$ times. This means that a special case of the theorem is Grover's algorithm. Intuitively, the theorem says that roughly $1/\sqrt{p}$ iterations of the base classical algorithm *in superposition* can amplify the success probability p nearly to one. A similar idea can be applied to even distributed algorithms on a quantum network[*6] consisting of multiple quantum computers.

Example 2: The leader election problem

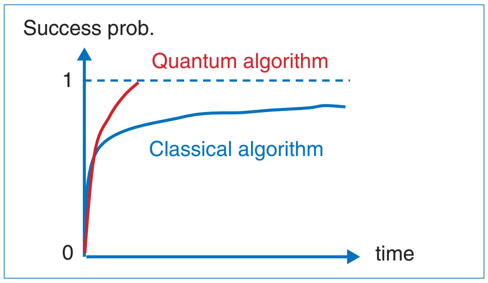
The leader election problem is a fundamental problem in the distributed computing field in that it often appears as a subproblem of various distributed computing problems. The goal of the problem is to elect a unique leader from among all nodes in a network (**Fig. 5**). The problem is seemingly easy, but in the most general case where each node does not necessarily have a unique identifier, it is mathematically proved that the problem cannot be solved within any bounded time with probability one by using classical computation and communication (i.e., sending/receiving *bits* through communication channels). In contrast, we proved that, if quantum communication and computation can be used, the problem can be solved within a certain bounded time with probability one [5]. This means that classical distributed computing and its quantum counterpart are *qualitatively* different. The first proof of our result did not use the quantum amplitude amplification, but an idea of another proof that employs the quantum amplitude amplification is given as follows.

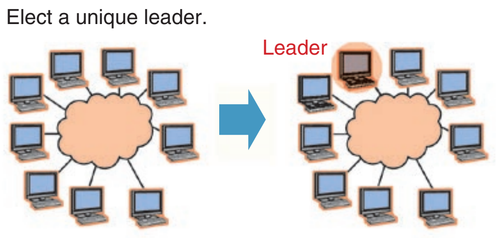
First consider the following simple classical algorithm: (1) Every node flips a coin; (2) the algorithm succeeds only when there is exactly one node that sees *heads*, in which case, the node will be elected as a leader. We can easily see that the success probability of this algorithm is $p = n/2^n$, where n is the number of nodes. Since we know the exact value of the success probability, we can construct a quantum algorithm that amplifies the success probability to one by repeating the coin flipping roughly $1/\sqrt{p} = \sqrt{2^n/n}$ times together with the quantum amplitude amplification. In fact, a much more sophisticated version of this idea flips coins roughly n times.

[*4] The connectivity of a graph is the property that, for any two vertices of the graph, there exists a path, i.e., a sequence of edges, from one vertex to another.
[*5] A Hamilton path of a graph is a sequence of edges of the graph, such that the sequence visits every vertex exactly once.
[*6] A quantum network consists of multiple quantum computers and quantum communication links between them. Quantum communication is the communication of qubits, which has already been achieved for practical use—for example, QKD (quantum key distribution)—by transmitting photons in a quantum state through optical fibers.

Leader election problem (LE)*
The goal is for all nodes in a network to collaborate in electing a unique leader from among the nodes.

* A fundamental distributed computing problem that emerges as a subproblem of various distributed computing problems.





Classical computation and communication	Cannot solve LE within any bounded time with probability one §.
Quantum computation and communication	Can solve LE within a certain bounded time with probability one.

§ Without the assumption that every party has a unique identifier.

Fig. 5. Quantum algorithm for electing a leader.

6. Conclusion

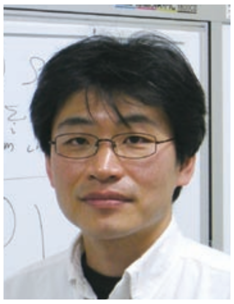
The development of quantum computation theory has progressed significantly in the last two decades. Recent research on quantum algorithms shows rich connections with classical computer science; state-of-the-art quantum search algorithms stem from (the quantum version of) random walk[*7] or semidefinite programming[*8], which are major technical tools in classical computer science. These connections will make it possible to make progress in both quantum and classical computer science in a collaborative way. Moreover, it is obviously important to study how to transform the high-level description of a quantum algorithm into a low-level description, i.e., a quantum circuit. A lot of fundamental techniques for this transformation have been intensively studied as well (e.g., [6]). Nevertheless, there are still a lot of fundamental problems to be solved before we can gain a deeper understanding of the potential power of quantum computing. To solve these problems and acquire new knowledge, we need to develop more novel techniques by making the best use of various ideas in related fields such as mathematics, physics, and classical computer science.

References

[1] P. W. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring," Proc. of 35th Annual IEEE Symposium on Foundations of Computer Science, pp. 124–134, Santa Fe, NM , USA, November 1994.
[2] L. K. Grover, "A Fast Quantum Mechanical Algorithm for Database Search," Proc. of 28th Annual ACM Symposium on Theory of Computing, pp. 212–219, Philadelphia, PA, USA, May 1996.
[3] A. Ambainis, K. Iwama, M. Nakanishi, H. Nishimura, R. Raymond, S. Tani, and S. Yamashita, "Quantum Query Complexity of Boolean Functions with Small On-sets," Proc. of 19th International Symposium on Algorithms and Computation, Vol. 5369 of Lecture Notes in Computer Science, pp. 907–918, Springer, 2008.
[4] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, "Quantum Amplitude Amplification and Estimation," Quantum Computation and Information, Vol. 305 of Contemporary Mathematics, pp. 53–74. American Mathematical Society, 2002.
[5] S. Tani, H. Kobayashi, and K. Matsumoto, "Exact Quantum Algorithms for the Leader Election Problem," ACM Transactions on Computation Theory, Vol. 4, No. 1, Article 1, 2012.
[6] Y. Takahashi and S. Tani, "Collapse of the Hierarchy of Constant-Depth Exact Quantum Circuits," Proc. of 28th IEEE Conference on Computational Complexity, pp. 168–178, Stanford, CA, USA, June 2013.

*7 Random walk on a graph is a stochastic process such that a particle continues to move one vertex to one of its neighbors chosen at random. It is a major technical tool for building efficient probabilistic algorithms.

*8 A semidefinite program (SDP) is a mathematical program that is described with positive-semidefinite matrices. There are efficient general algorithms that approximately solve SDPs.



Seiichiro Tani

Distinguished Scientist, Media Information Laboratory, NTT Communication Science Laboratories.

He received the B.E. in information science from Kyoto University in 1993, and the M.S. and Ph.D. in computer science from the University of Tokyo in 1995 and 2006, respectively. He joined NTT in 1995 and studied algorithms for LSI testing, multi-point communication, and distributed computing problems. He is currently working on algorithmic theory and complexity of quantum computing. From 2004 to 2009, he was also a researcher of the ERATO Quantum Computation and Information Project of Japan Science and Technology Agency (JST). From 2010 to 2011, he was a visiting researcher at the Institute for Quantum Computing, University of Waterloo, Ontario, Canada. He received the 2000 IEICE ISS Best Paper Award. He is a member of IEICE (Institute of Electronics, Information and Communication Engineers), IPSJ (Information Processing Society of Japan), IEEE (Institute of Electrical and Electronics Engineers), and ACM (Association for Computing Machinery).