Another Quantum Turing Machines

Ikeda, Daisuke
Research Institute of Fundamental Information Science Kyushu University

Arikawa, Setsuo
Research Institute of Fundamental Information Science Kyushu University

出版情報：RIFIS Technical Report. 114, 1995-05-29. Research Institute of Fundamental Information Science, Kyushu University
バージョン：
権利関係：

RIFIS-TR-CS-114

RIFIS Technical Report

Another Quantum Turing Machines

Daisuke Ikeda and Setsuo Arikawa

May 29, 1995

Research Institute of Fundamental Information Science
Kyushu University 33
Fukuoka 812, Japan
E-mail: daisuke@rifis.kyushu-u.ac.jp
Phone: 092-641-1101 ex. 4459

Another Quantum Turing Machines

Daisuke Ikeda and Setsuo Arikawa
Research Institute of Fundamental Information Science,
Kyushu University 33, Fukuoka 812, Japan
{daisuke,arikawa}@rifis.kyushu-u.ac.jp

Abstract

The quantum Turing machines by Bernstein & Vazirani are based on vectors and matrices as in quantum mechanics, so that it is difficult to make clear difference between the quantum Turing machines and the usual probabilistic Turing machines. This paper gives another formulation which can make difference clear.

We consider that the superposition of configurations, a basic concept of the quantum Turing machines, should also be applied to the probabilistic Turing machines. From this viewpoint we first give a new definition of the probabilistic Turing machines. Then we define the quantum Turing machine as an extension of the probabilistic Turing machine. We show the relationship between Bernstein & Vazirani's definition and ours.

In both types of the quantum Turing machines there still remains another difficulty that the machines are required to be time-bounded in order for users to get results explicitly from the machines. We overcome this difficulty by modifying our quantum Turing machines, and show that our new machines can solve the satisfiability and the validity problems in polynomial time.

1 Introduction

The computation theory based on Turing machines does not take account of quantum mechanics which is a basic theory of physics. In quantum mechanics, physical states of some objects are changed by observations. The observer can only know the states after the observations but can not know the states before the observations. The physical states correspond to configurations in computations of Turing machine, which are to be recognized by *a user*, the observer. The user may observe all the configurations of a Turing machine, but the observations do not have any effects on the subsequent computation of the machine.

In order to introduce quantum mechanics into the computation theory, Deutsch [2] has proposed the quantum computers. In the quantum computers, as in quantum mechanics, a configuration is defined as a base vector in Hilbert space and a transition is described by a unitary matrix. A computation by a quantum computer is defined to operate a unitary matrix to vectors.

To theoretical computer scientists, the representation of computers in terms of matrices and vectors is less comprehensible than that in terms of transition functions. Bernstein & Vazirani [1] have formulated the quantum computers called quantum Turing machines in terms of the transition functions. However, a condition for the transition functions is still written in

terms of vectors and it is hard to check, given a transition function, whether it satisfies the condition or not. The definition of computations by their quantum Turing machines is exactly the same as by quantum mechanics, so that they do not make full use of their own definition. Other researches on the quantum Turing machines so far published make use of vectors and matrices [3, 4, 5, 6].

In this paper first we define quantum Turing machines without using matrices and vectors. The quantum Turing machine defined by Bernstein & Vazirani behaves according to a probabilistic distribution. Hence it is natural to take a quantum Turing machine as an extended probabilistic Turing machine. We define quantum Turing machines by extending the probabilistic Turing machines and show that our quantum Turing machines have the same powers as those by Bernstein & Vazirani.

Both Bernstein & Vazirani's machines and ours are required to be time-bounded in order for users to get results explicitly from the machines. We remove this restriction by modifying the quantum Turing machines so that they can show users the ends of their computations. In order to show the power of the new quantum Turing machines, we prove that a class of languages accepted by the new machines in polynomial time includes both NP and co-NP.

In Section 2, we introduce the concepts of superpositions of configurations and observations into the probabilistic Turing machines. In Section 3, we define the quantum Turing machine as an extension of the probabilistic Turing machine, and show a difference between Bernstein & Vazirani's definition and ours. In Section 4, we point out a difficulty which lies in both types of the quantum Turing machines, and solve it by introducing new machines called halting quantum Turing machines. In Section 5, we show the power of the halting quantum Turing machines.

2 Probabilistic Turing Machines

We start with modifying the probabilistic Turing machines from a viewpoint of quantum mechanics, by which we can extend the probabilistic Turing machines to the quantum Turing machines as seen in Section 3. The modification does not cause any essential difference from the original probabilistic Turing machines.

A probabilistic Turing machine consists of a finite control, an infinite tape that is divided into cells, and a tape head that can read and write. The tape has a leftmost cell, but it is infinite to the right.

Definition 1. A *probabilistic Turing machine* is a quintuple $M = (K, \Sigma, \delta, q_0, F)$, where

(a) K is the finite set of *states.*

(b) Σ is an *alphabet*, i.e., the finite set including a *blank symbol* B.

(c) $q_0 \in K$ is the *initial state.*

(d) $F \subseteq K$ a the set of *final states.*

(e) δ is a partial function from $K \times \Sigma \times K \times \Sigma \times \{L, R\}$ to $[-1, 1]$ called *transition function,* where

(1-a) for any $p \in K$ and $a \in \Sigma$, $\sum_{q,b,d} \delta(p, a, q, b, d)^2 = 1$. ∎

We call a value of a transition function a *transiton value* and assume that any transiton value is not equal to 0.

Assume that a probabilistic Turing machine $M = (K, \Sigma, \delta, q_0, F)$ is in a configuration such that the current state is $p \in K$ and the head is reading a symbol $a \in \Sigma$. Then $\delta(p, a, q, b, d) = \alpha$ denotes that the probability with which M changes the state into q, the symbol into b and moves the head toward the direction d is α^2, where $q \in K, b \in \Sigma, d \in \{L, R\} - 1, \leq \alpha \leq 1, \alpha \neq 0$. A transition from a configuration C to another configuration D with transiton value α is denoted by $C \vdash D(\alpha)$.

Now we define superpositions of configurations and observations. Let M be a probabilistic Turing machine. Then a *superposition* of M is defined as a sequence of pairs of a real number α_i and a configuration $C_i\,(1 \leq i \leq m)$ of M, that is, $((\alpha_1, C_1), \ldots, (\alpha_m, C_m))$. For an initial configuration C_0, *initial superposition* is $(1, C_0)$. A real number α_i of (α_i, C_i) is called an *amplitude* of C_i and α_i^2 is called an *expectation value* of C_i. Let $S = ((\alpha_1, C_1), \ldots, (\alpha_m, C_m))$ be a superposition of M. Then an *observation* of S changes S itself into another superposition $(1, C_i)$ with an expectation value α_i^2 for some $i\,(1 \leq i \leq m)$.

Definition 2. Let M be a probabilistic Turing machine and $S = (1, C)$ be a superposition of M. Then the subsequent superposition S' of S is defined by

$$S' = ((\alpha_1, C_1), \ldots, (\alpha_m, C_m)),$$

where each $C_i\,(1 \leq i \leq m)$ is a configuration of M such that $C \vdash C_i(\alpha_i)$. M is said to move from S to S' by one step and denoted by $S \vdash S'$. ∎

In the above definition, the sum of all expectation values of S' is equal to 1, that is, $\sum \alpha_i^2 = 1$ by the condition (1-a). Thus we can identify the expectation value with the probability.

A configuration C in a traditional probabilistic Turing machine corresponds to the superposition $(1, C)$. A transition of the traditional machine $C \vdash C'$ with probability α is considered that first the machine moves $(1, C) \vdash S'$, where S' includes C', and then an observation changes S' into $(1, C')$ with the expectation value α. A user can only know the superposition $(1, C')$ that is a physical state after the observation. The time for observations can be neglected.

3 Quantum Turing Machines

In this section, we extend the probabilistic Turing machines into the quantum Turing machines. The probabilistic Turing machines defined in the previous section have to be observed at every step, because we have defined only the transitions from a superposition $(1, C)$. However, it is not necessary for a user to observe at every step, because the user want to get just results. So we define transitions from a superposition before an observation to another superposition. In these transitions, there appears another feature of quantum mechanics that might change the sum of all expectation values. If the sum is changed, we can not identify the expectation value with the probability. In quantum mechanics the sum is preserved, so that a probabilistic Turing machine needs another conditions on the transition function. The probabilistic Turing machines with the new conditions are the quantum Turing machines that we want.

Definition 3. A *quantum Turing machine* is a probabilistic Turing machine $M = (K, \Sigma, \delta, q_0, F)$, where the transition function δ is a partial function from $K \times \Sigma \times K \times \Sigma \times \{L, R\}$ to $[-1, 1]$ such that

(3-a) for any $p \in K$ and $a \in \Sigma$, $\sum_{q,b,d} \delta(p,a,q,b,d)^2 = 1$,

(3-b) for any (p_1, a_1) and $(p_2, a_2) \in K \times \Sigma$ $(p_1 \neq p_2$ or $a_1 \neq a_2)$,

$$\sum_{q,b,d} \delta(p_1,a_1,q,b,d) \cdot \delta(p_2,a_2,q,b,d) = 0,$$

(3-c) for $\delta(p,a,q,b,d) = \alpha$ and $\delta(p',a',q',b',d') = \alpha'$, if $q = q'$ then $d = d'$. ∎

Only the condition (3-c) is different from one in Bernstein & Vazirani's quantum Turing machine [1], in which the following condition corresponds to the condition (3-c):

(4-a) In $|K|$ dimension complex space $C^{|K|}$, there exists two mutual orthogonal subspaces C_L and C_R such that

$$\sum_{q \in K} \delta(p,a,q,b,d)\boldsymbol{q} \in C_d,$$

where $\boldsymbol{q}$ is a vector corresponding to a state $q \in K$.

It is hard to check, given a transition function, whether it satisfies the condition (4-a) or not. In fact, there are 2^n possible ways to divide n base vectors into two groups, even if the sum of the dimensions of C_L and C_R are restricted to $n = |K|$. On the other hand, it is easy to check whether a given transition function satisfies the condition (3-c)

Now we extend the transitions in Section 2 into those from a superposition before an observation. Then the expectation value can also be identified with the probability as in Theorem 1. From now on, $(\alpha_1 C_1, \ldots, \alpha_m C_m)$ also denotes a superposition $((\alpha_1, C_1), \ldots, (\alpha_m, C_m))$.

Definition 4. Let M be a quantum Turing machine, $S = (\alpha_1 C_1, \ldots, \alpha_m C_m)$ be a superposition of M, and each D_{ij} $(1 \leq i \leq m, 1 \leq j \leq n_i)$ be a configuration of M and β_{ij} be a transiton value such that $C_i \vdash D_{ij}(\beta_{ij})$. Then the subsequent superposition S' of S is defined as follows:

$$S' = \left(\sum_{(k,l) \in V_{ij}} \alpha_k \beta_{kl}, D_{ij} \right)$$

for all i, j, where V_{ij} is the set of all (k,l) such that $D_{kl} = D_{ij}$. M is said to move from S to S' by one step and denoted by $S \vdash S'$. ∎

Note that each V_{ij} contains at least one element (i,j). In the above definition, if there is no configuration D such that $C_i \vdash D$ for a configuration C_i, then we consider that M moves from C_i into itself with the expectation value 1. If there exist configurations in a superposition such that $D_{ij} = D_{kl}$ $(i \neq k)$, then M is said to have *interference*. Figure 1 shows transitions of a quantum Turing machine, where there are no interferences, i.e., the case $D_{ij} \neq D_{kl} (i \neq k)$.

Lemma 1. Let M a quantum Turing machine and $S = (\alpha_1 C_1, \ldots, \alpha_m C_m)$ be any superposition of M. Then

$$\sum_{i=1}^{m} \alpha_i^2 = 1.$$

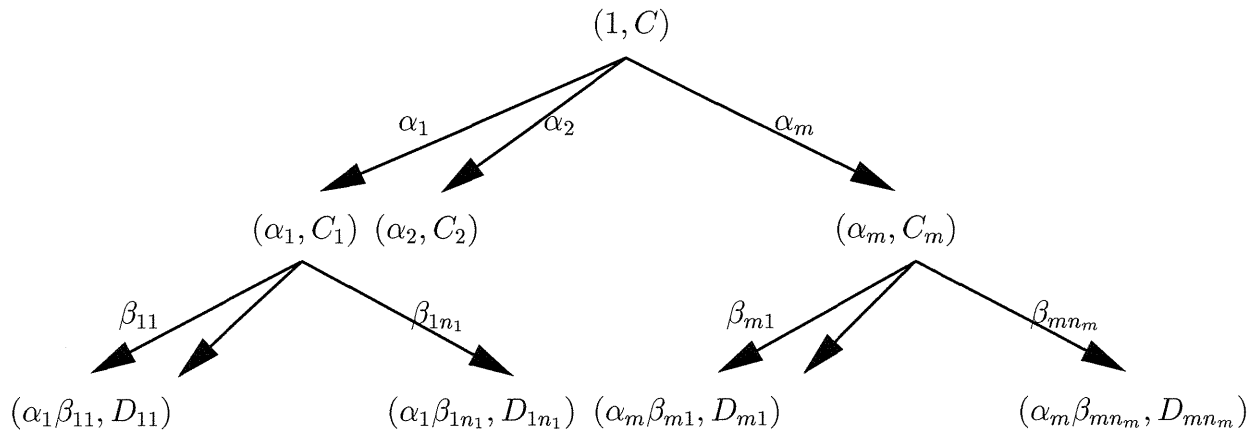

Figure 1: A superposition of a quantum Turing machine

Proof: We prove it by an induction on step t. The case $t = 1$ is trivial, because a transition is from an initial superposition and the condition (3-a) holds.

Let $S = (\alpha_1 C_1, \ldots, \alpha_m C_m)$ be a superposition of M at step t, and assume that

$$\sum_{i=1}^{m} \alpha_i^2 = 1. \tag{1}$$

Let P be the sum of the expectation values of the configurations in S, and each D_{ij} be a configuration and $\beta_{ij}\,(1 \leq i \leq m, 1 \leq j \leq n_i)$ be an amplitude such that $C_i \vdash D_{ij}(\beta_{ij})$. For any D_{ij} and $D_{kl}(i \neq k)$, if $D_{ij} \neq D_{kl}$ then

$$P = \sum_{i,j} \alpha_i^2 \beta_{ij}^2 = \alpha_1^2 \sum_{j=1}^{n_1} \beta_{1j}^2 + \cdots + \alpha_m^2 \sum_{j=1}^{n_m} \beta_{mj}^2 = \sum_{i=1}^{m} \alpha_i^2 = 1,$$

because the equation (1) and the condition (3-a) hold.

Next we consider the case where there is a configuration which appears more than once in S'. Let D_{1l} be configurations of M and $\beta_{il}\,(1 \leq l \leq n_1)$ be amplitudes such that $C_1 \vdash D_{1l}(\beta_{1l})$ and $C_k \vdash D_{1l}(\beta_{kl})$, where $k \neq 1$. Then, since the amplitude of D_{1l} is $(\alpha_1\beta_{1l} + \alpha_k\beta_{kl})$, the expectation value of D_{1l} is

$$(\alpha_1\beta_{1l} + \alpha_k\beta_{kl})^2 = \alpha_1^2\beta_{1l}^2 + \alpha_k^2\beta_{kl}^2 + 2\alpha_1\alpha_k\beta_{1l}\beta_{kl}. \tag{2}$$

Now we consider the transition function which is applied to the transitions $C_1 \vdash D_{1l}(\beta_{1l})$ and $C_k \vdash D_{1l}(\beta_{kl})$. Let $p_i \in K, a_i \in \Sigma\,(i = 1, k)$ be the current state and the symbol scanned by the head in $C_i\,(i = 1, k)$, respectively. The head positions in C_1 and C_k are identical, and so are the move directions. Therefore the symbols newly written are also identical. We denote these states, directions and symbols by $q_l \in K, d \in \{L, R\}, b_l \in \Sigma$, respectively. Then the transiton values applied to $C_i(i = 1, k)$ are

$$\begin{aligned} \delta(p_1, a_1, q_l, b_l, d) &= \beta_{1l}, \\ \delta(p_k, a_k, q_l, b_l, d) &= \beta_{kl}. \end{aligned} \tag{3}$$

From the equations (2) and (3), the sum P_1 of the expectation values of D_{1j} for all j ($1 \leq j \leq n_1$) is

$$\begin{aligned} P_1 &= \alpha_1^2 \sum_{j=1}^{n_1} \beta_{1j}^2 + \alpha_k^2 \sum_{j=1}^{n_k} \beta_{kj}^2 + 2\alpha_1\alpha_k \sum_l \beta_{1l}\beta_{kl} \\ &= \alpha_1^2 \sum_{j=1}^{n_1} \beta_{1j}^2 + \alpha_k^2 \sum_{j=1}^{n_k} \beta_{kj}^2 + 2\alpha_1\alpha_k \sum_{q_l,b_l,d} \delta(p_1,a_1,q_l,b_l,d)\cdot\delta(p_k,a_k,q_l,b_l,d). \end{aligned}$$

Since $C_1 \neq C_k$, $(p_1,a_1) \neq (p_k,a_k)$. Therefore

$$\sum_{q_l,b_l,d} \delta(p_1,a_1,q_l,b_l,d)\cdot\delta(p_k,a_k,q_l,b_l,d) = 0$$

by the condition (3-b). Thus we have:

$$P_1 = \alpha_1^2 \sum_{j=1}^{n_1} \beta_{1j}^2 + \alpha_k^2 \sum_{j=1}^{n_k} \beta_{kj}^2 = \alpha_1^2 + \alpha_k^2$$

A similar argument to the above is valid also for interferences of configurations except for D_{1l}. Hence we have:

$$P = \sum_{i=1}^{m} \alpha_i^2 = 1$$

by the equation (1). ∎

As seen in the proof of Lemma 1, we have made the conditions (3-b) and (3-c) to prevent invalid interferences from occurring. If there is no interference terms except for the last one in the equation (2), the sum of all expectation values are not equal to 1. However there must exist configurations which negate the last term by the conditions (3-b) and (3-c).

Now we prove the converse of Lemma 1. Althouh the probabilistic Turing machines are assumed to be observed at each step, we do not need this assumption in the rest of this section.

Lemma 2. Let $S = (\alpha_1 C_1, \ldots, \alpha_m C_m)$ be any superposition of a probabilistic Turing machine $M = (K, \Sigma, \delta, q_0, F)$. Then M is a quntum Turing machine if $\sum_{i=1}^{m} \alpha_i^2 = 1$.

Proof: The condition (3-a) holds, because M is a probabilistic Turing machine. Also the condition (3-b) trivially holds. Now assume that the condition (3-c) does not hold. Consider the following quntum Turing machine:

$$\begin{aligned} \delta(q_0,\diamond,q_1,1,R) &= 1/\sqrt{2} \\ \delta(q_0,\diamond,q_2,0,R) &= 1/\sqrt{2} \\ \delta(q_1,\diamond,q_3,\diamond,L) &= 1 \\ \delta(q_2,\diamond,q_4,\diamond,R) &= 1 \\ \delta(q_3,1,q_f,0,R) &= 1 \\ \delta(q_4,\diamond,q_f,\diamond,L) &= 1, \end{aligned}$$

where the $\diamond$ denotes any symbol in Σ. For an input $x = a_1a_2a_3 \in \Sigma^*$, the above machine moves as follows:

$$\begin{aligned} (1, q_0a_1a_2a_3) &\vdash ((1/\sqrt{2}, 1q_1a_2a_3), (1/\sqrt{2}, 0q_2a_2a_3)) \\ &\vdash ((1/\sqrt{2}, q_31a_2a_3), (1/\sqrt{2}, 0a_2q_4a_3)) \\ &\vdash (\sqrt{2}, 0q_fa_2a_3) = S. \end{aligned}$$

Thus $\sum \alpha_i^2 \neq 1$ for the above S. Hence the condition (3-c) holds if $\sum \alpha_i^2 = 1$. ∎

We obtain the following theorem by Lemma 1 and Lemma 2.

Theorem 1. Let $S = (\alpha_1 C_1, \ldots, \alpha_m C_m)$ be any superposition of a probabilistic Turing machine $M = (K, \Sigma, \delta, q_0, F)$. Then M is a quntum Turing machine if and only if $\sum_{i=1}^{m} \alpha_i^2 = 1$.

Due to this theorem, we hereafter use *probability* instead of *expectation value*. The conditions (3-a),(3-b) and (4-a) hold if and only if the sum of the expectation values is equal to 1 [1]. By this result and the above theorem, Bernstein & Vazirani's quntum Turing machines are equivalent to ours.

4 Halting Quantum Turing Machines

In this section, we describe a problem of the quantum Turing machines, and solve it by introducing the quantum Turing machines with new ability, called the halting quantum Turing machines. First consider the following example. Let M be a quantum Turing machine and C_i $(1 = 1, 2)$, D_j $(1 \leq j \leq 4)$, and E_k $(k = 1, 2)$ be the configurations of M at step $t, t+1$, and $t+2$, respectively. Figure 2 shows the transitions between these configurations. The numbers beside the arrows denote transiton values. If M is observed only at step $t+2$, the

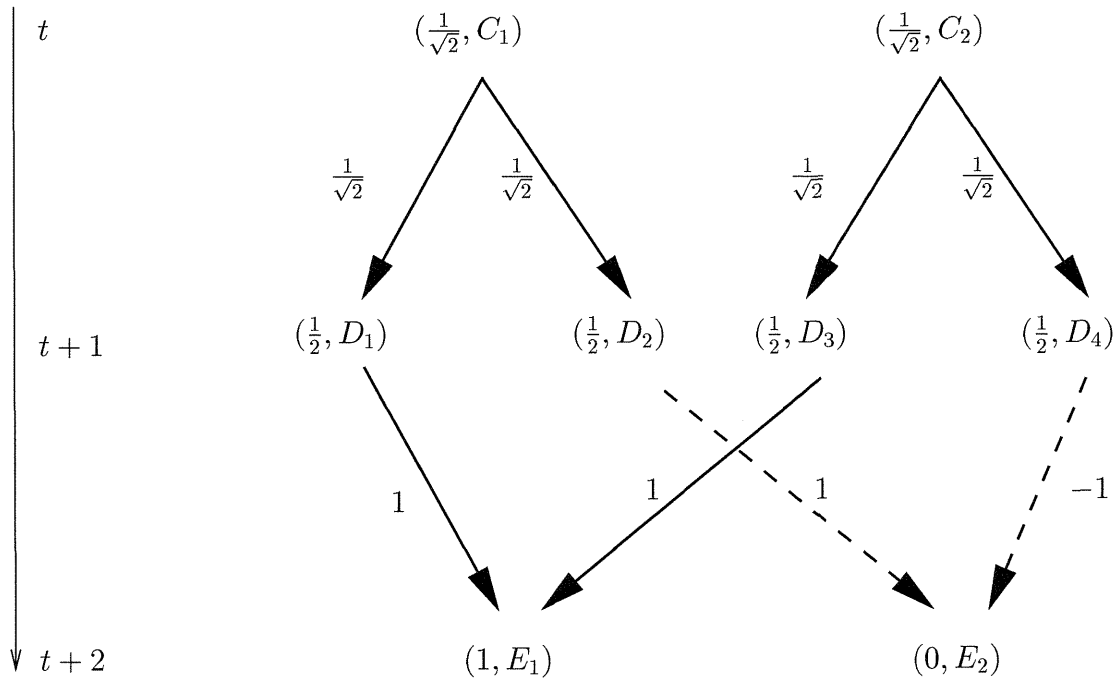

Figure 2: An example of quantum Turing machines

configuration of M is E_1 with probability 1, and we can not know whether the computation path is $C_1 \vdash D_1 \vdash E_1$ or $C_2 \vdash D_3 \vdash E_1$. On the other hand, if M is observed at each step, a

configuration of M is either C_1 or C_2 with the same probability at step t. At step $t+1$ each probability of $D_j (1 \leq j \leq 4)$ are also the same. The probability that a configuration of M is E_1 at step $t+2$ is $1/2 \cdot 1/2 + 1/2 \cdot 1/2 = 1/2$ if M is observed at each step.

The above example shows that an observation changes the rest of a computation of a quantum Turing machine. Then the following problem arises: A user can not know when to observe a quantum Turing machine to get results. Bernstein & Vazirani have defined the class *BQP*.

Definition 5. (Bernstein & Vazirani) $L \in$ BQP if there exists a quantum Turing machine M and a polynomial $p(n)$ such that, when M is observed at step $p(|x|)$ on a given input x, if $x \in L$, then M is an in acceptable configuration with probability more than or equal to $2/3$; otherwise, M is not in an acceptable configuration with probability more than or equal to $2/3$.

In the above definition, a quantum Turing machine is time-bounded, so that it is no matter when the user should observe the quantum Turing machine. The user, however, should know the time. This is a conspicuous restriction on quantum Turing machines. So we introduce new quantum Turing machines called halting quantum Turing machines to remove the restriction. For a superposition S, $|S|$ denotes the number of different configurations in S.

Definition 6. A *halting quantum Turing machine* M is a quntum Turing machine such that, for any superposition S of M, M tells whether $|S| = 1$ or not. Let C_h be a halting configuration of M. M is said to *stop* if $|S| = 1$ and $S = (1, C_h)$. ∎

Intuitively a halting quantum Turing machine is a quntum Turing machine which has a speccall cell called a *halting cell*. Before a computation, the blank symbol is written on the halting cell. When $|S| = 1$ in the computation, the machine writes 1 on the cell. The halting cell is always allowed to be observed, that is, any observaiton of the halting cell does not change the rest of the computation even if $|S| > 1$. Because the blank symbol is written on the halting cell in all the configurations in S if $|S| > 1$, the cell does not have any effects on interference. Thus a user is allowed to observe only the halting cell at any time. After 1 is written on the halting cell, the user can observe the other parts of M and check whether M accepts a given input.

Let M be a halting quantum Turing machine , $x \in \Sigma^*$ be an input for M, and $T(n)$ be a function. Then M is said to *accept* x if a superposition includes an acceptable configuration when M stops. $L(M)$ denotes the set of all words accepted by M. M is said to be $T(n)$ *time-bounded* if x is accepted by M in $O(T(|x|))$ steps.

5 Power of Halting Quantum Turing Machine

In this section, we show the power of halting quantum Turing machines. Let Q be the set of all languages accepted by polynomial time bounded halting quantum Turing machines.

In general, even if the halting cell shows the stop, a user can not decide whether a given input is accepted or not without observation. The following proposition, however, assures that the user can decide whether the halting quantum Turing machine M accepts an input only with an observation of the halting cell, where $L(M) \in$ Q.

Proposition 1. Let $L \in$ Q be a language, $x \in \Sigma^*$ be an input and M be a halting quantum Turing machine which accepts L in polynomial time $p(n)$. Then there exists a polynomial

$p'(n) = O(p(n))$ and a halting quantum Turing machine M' such that, at step $p'(|x|)$, if $x \in L$, then 1 is written on halting cell; otherwise 0 is written.

Proof: We construct the following halting quantum Turing machine M'. M' simulates M on an input x counting the steps of M up to $p(|x|)$. If M stops within $p(|x|)$ steps and accepts the input, then M' writes 1 on the halting cell, otherwise, writes 0. For this M', observations of the halting cell at any step are also allowed. ∎

Now we show the relationship between Q and co-NP by using the validity problem:

$$\text{VALIDITY} = \{f \mid \text{Boolean formula } f \text{ is valid}\}.$$

Theorem 2. VALIDITY $\in$ Q.

Proof: Consider the halting quantum Turing machine M which realizes Algorithm 1. Assume that an input for the algorithm is a Boolean formula f, and it is written on the tape from the leftmost cell to the right. $a_1, a_2, \ldots$ denote cells starting from the cell next to the input on the right. In Algorithm 1, a statement within the parentheses [and] , say

$$[a_1 \leftarrow 0(1/\sqrt{2}), a_1 \leftarrow 1(1/\sqrt{2})],$$

means that, for two transitions which write 0 and 1 on the cell a_1, the transiton values are both $1/\sqrt{2}$.

Algorithm 1

1 count the number of variables in the input f;
  ▷ *Let it be m.*
2 **for** $i = 1$ to m **do** $[a_i \leftarrow 0(1/\sqrt{2}), a_i \leftarrow 1(1/\sqrt{2})]$;
  ▷ *assign 0 or 1 to each variable with same probability.*
3 $a_{m+1} \leftarrow f(a_1, \cdots, a_m)$;
4 **for** $i = m$ to 1 **do**
5 **if** $a_i = 0$ **then do**
6 **if** $a_{m+1} = 1$ **then**
7 $[a_{m+1} \leftarrow 0(-1/\sqrt{2}), a_{m+1} \leftarrow 1(1/\sqrt{2})]$;
8 $a_i \leftarrow 1$;
9 **else** change state into p_1, write i next to a_{m+1};
  ▷ *Assume that there is no* $\delta(p_1, *, *, *)$.
10 **end**
11 **else do**
12 **if** $a_{m+1} = 1$ **then**
13 $[a_{m+1} \leftarrow 0(1/\sqrt{2}), a_{m+1} \leftarrow 1(1/\sqrt{2})]$;
14 $a_i \leftarrow 1$;
15 **else** change state into p_1, write i next to a_{m+1};
16 **end**
17 **end**

We show that M accepts VALIDITY in polynomial time. Let S_0 be a superposition just after the line 3 on an input f. Then

$$S_0 = (\frac{1}{\sqrt{2^m}}, x_1 x_2 \ldots x_m; x_{m+1}), \ |S_0| = 2^m,$$

where each $x_i \in \{0,1\}(1 \leq i \leq m)$ and $x_{m+1} = f(x_1, \ldots, x_m)$. In this proof, a configuration is represented only by the contents of the tape except for the input. Semicolon(;) is used to distinguish a truth assignment from its value f.

(1) The case $f \in$ VALIDITY. For all configurations in S_0, $x_{m+1} = 1$. We show it by an induction on m such that for any $m \geq 1$, M moves from S_0 to the superposition S', where $S' = (1, \overbrace{11\ldots1}^{m}; 1)$.

If $m = 1$, then M moves from S_0 as follows:

$$\begin{aligned} S_0 &= ((\frac{1}{\sqrt{2}}, 0; 1), (\frac{1}{\sqrt{2}}, 1; 1)) \\ &\vdash ((-\frac{1}{\sqrt{2^2}}, 0; 0), (\frac{1}{\sqrt{2^2}}, 0; 1), (\frac{1}{\sqrt{2^2}}, 1; 0), (\frac{1}{\sqrt{2^2}}, 1; 1)) \\ &\vdash ((-\frac{1}{\sqrt{2^2}}, 1; 1), (\frac{1}{\sqrt{2^2}}, 1; 1), (\frac{1}{\sqrt{2^2}}, 1; 0), (\frac{1}{\sqrt{2^2}}, 1; 1)) \\ &= (1, 1; 1) \end{aligned}$$

Figure 3 shows the above transition.

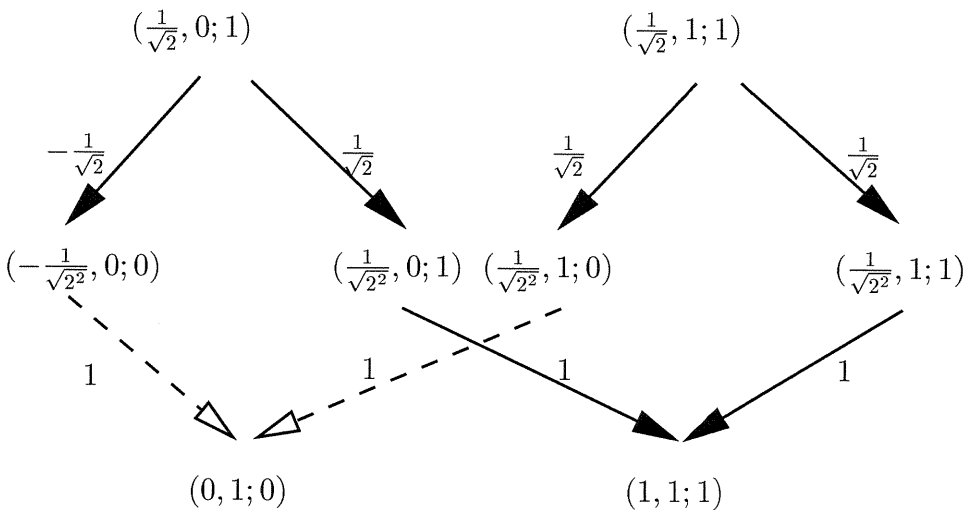

Figure 3: The case $m = 1$

Assume the case m. M moves from S_0 to the superposition T, where

$$S_0 \vdash^* ((\frac{1}{\sqrt{2}}, 01\ldots1; 1), (\frac{1}{\sqrt{2}}, 11\ldots1; 1)) = T,$$

because there is no interference between a configuration with the cell a_1 written 0 and one with the cell written 1. The contents of all other cells $a_j (j \geq 2)$ are not changed in the rest of the computation, so that they do not have any effect on it. Therefore we can prove the case $m + 1$ in a similar way to the case $m = 1$. This procedure is finished in polynomial time.

(2) The case $f \notin$ VALIDITY. There are configurations in S_0 such that $a_{m+1} = 0$. Since M changes its state into p_1 from one of such configurations and since then it never moves after that, M does not enter to any acceptable configuration.

By the claims (1) and (2), if $f \in$ VALIDITY, then M moves from S_0 to $(1, C_h)$ in polynomial time, where C_h is a halting configuration. Hence the proof is completed by Proposition 1. ∎

Corollary 1. co-NP $\subseteq$ Q.

By modifying the above discussion in the following way, we can also prove that there exists a halting quantum Turing machine which accepts the satisfiability problem in polynomial time. First we exchange the symbol 0 for 1 in the halting cell in Proposition 1. We also replace the statements $a_{m+1} = 1$ in line 6 and 12 in Algorithm 1 with $a_{m+1} = 0$. Then it is clear that, for any input formula f, the halting quantum Turing machine constructed in this way writes 0 on the halting cell if and only if $f \notin$ SAT, where

$$\text{SAT} = \{f \mid \text{Boolean formula } f \text{ is satisfiable}\}.$$

Hence we have the following theorem and corollary.

Theorem 3. SAT $\in$ Q.

Corollary 2. NP $\subseteq$ Q.

6 Discussion

We have introduced the superpositions of configurations and observations into the probabilistic Turing machines and modified their transitions. In our formulation, a transition from a configuration to another configuration is interpreted in the following way: A probabilistic Turing machine moves from a superposition of only one configuration to another superposition of some configurations, and then a configuration is chosen from the latter superposition with some expectation value by an observation.

We have extended these probabilistic Turing machines into the quantum Turing machines. We have presented a necessary and sufficient condition for the total probabilities of configurations to be 1, which is more easily checkable than that by Bernstein & Vazirani. We have also shown that our quantum Turing machines are equivalent to Bernstein & Vazirani's.

Since a user can not know configurations of a quantum Turing machine without an observation, (s)he can neither know when the machine stops nor get results explicitly. To solve this problem, we have introduced the halting quantum Turing machine, which is a quantum Turing machine with an additional function. To show the power of the halting quantum Turing machines, we have presented a polynomial time algorithm which solves the validity and the satisfiability problems. While this result shows that the function newly added to halting quantum Turing machine is powerful, it does not immediately means the quantum Turing machines themselves are powerful.

References

[1] E. Bernstein and U. Vazirani, Quantum Complexity Theory, in *Proceedings of 1993 ACM Symposium on Theory of Computing* (1993), pp. 11-20.

[2] D. Deutsch, Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer, *Proceedings of the Royal Society of London*, Vol. **A 400** (1985), pp. 97-117.

[3] D. Deutsch and R. Jozsa, Rapid solution of problems by quantum computation, *roceedings of the Royal Society of London.* Vol. **A439**, pp. 553-558 (1992).

[4] P. Shor, Algorithms for Quantuim Computation: Discrete Log and Factoring, in *Proceedings 35th IEEE Symposium on Foundations of Computer Sciende*, IEEE Press (1994), pp. 124-134.

[5] D. Simon, On the power of quantum computation, in *Proceedings 35th IEEE Symposium on Foundations of Computer Sciende*, IEEE Press (1994), pp. 116-123.

[6] A. Yao, Quantum Circuit Complexity, *Proceedings of 34th Symposium on Foundations of Computer Science* (1993), pp. 352-361.