数理解析研究所講究録 1205 巻 2001 年 188-193

量子モデルと確率モデルの確率計算の違いによって生じる計算能力の差について

天野 正己 (Masami Amano)
京都大学大学院情報学研究科

岩間 一雄 (Kazuo Iwama)
京都大学大学院情報学研究科

Abstract

The main purpose of this paper is to show that we can exploit the difference (l_1-norm and l_2-norm) in the probability calculation between quantum and probabilistic computations to claim the difference in their computational powers. It is shown that there is a language L which contains sentences of length up to $O(n^{c+1})$ such that: (i) There is a one-way quantum finite automaton (qfa) of $O(n^{c+4})$ states which recognizes L. (ii) However, if we try to simulate this qfa by a probabilistic finite automaton (pfa) *using the same algorithm*, then it needs $\Omega(n^{2c+4})$ states. It should be noted that we do not prove real lower bounds for pfa's but show that if pfa's and qfa's use exactly the same algorithm, then qfa's need much less states.

1 Introduction

It is well known that BPP can be simulated by BQP almost directly, i.e., quantum computation with a bounded error is at least as powerful as its probabilistic counterpart [BV97]. Furthermore, it appears that quantum computation has several merits over probabilistic computation, which include: (i) Quantum computation efficiently gives us useful information about the period of a periodic function [Sho94, Sim94]. (ii) Negative values for amplitudes are allowed, which can be used, for instance, to cancel other amplitudes [DJ92] and to "shift" some amplitude from one state to another [Gro96]. (iii) Even complex numbers ($i\sin\phi+\cos\phi$) can be used for amplitudes, which allows us to do tricky operations by rotating the complex number appropriately [AF98].

In this paper, we focus our attention on more basic feature of quantum computation which has been relatively less focused on in the literature, namely, the difference in the way of *probability calculation*. It is a fundamental rule of quantum computation that if a state q has an amplitude of σ, then q will be observed not with probability σ but with probability σ^2. Suppose that there are ten pairs of state $(p_1,q_1),\cdots,(p_{10},q_{10})$ where, for each $1\leq i\leq 10$, either p_i and q_i has the amplitude $1/\sqrt{10}$ (we say that p_i is ON if it has the amplitude and OFF otherwise.). We wish to know how many p_i's are ON. This can be done by "gathering" amplitudes by applying a Fourier transform from p_i's to r_i's and observing r_{10} (see later sections for details), if all ten p_i's are ON, then the amplitude of r_{10} after Fourier transform is one and it is observed with probability one. If, for example, only three p_i's are ON, then the amplitudes of r_{10} is 3/10 and is observed with probability 9/100.

In the case of probabilistic computation, we can also gather the probability of p_i's (= 1/10 for each) simply by defining a (deterministic) transition from p_i to r_{10}. If all pairs are ON, then the probability that r_{10} is observed is again one, but if only three p_i's are ON, the probability is 3/10. If the latter case that only three p_i's are ON is associated with some erroneous situation, this probability of 3/10 is much larger than 9/100 in the quantum case. In other words quantum computation can enjoy much smaller error-probability only due to the difference in the rule of probability calculation.

The question is of course whether we can turn this feature into some concrete result or how we can translate this difference in probability into some difference in efficiency like time and space. In this paper we give an affirmative answer to this question by using quantum finite automata; we prove that there is a language L which contains sentences of length up to $O(n^{c+1})$ such that: (i) There is a one-way quantum finite automaton (qfa) of $O(n^{c+4})$ states which recognizes L. (ii) However, if we try to simulate this qfa by a probabilistic finite automaton (pfa) *using the same algorithm*, then it needs $\Omega(n^{2c+4})$ states. It should be noted that we do not prove real lower bounds for pfa's but show that if pfa's and qfa's use exactly the same algorithm (the only difference is the way of gathering amplitudes mentioned above), then qfa's need much less states. As one can see later, the algorithm is probably the best one and even if there would be another algorithm, it probably produces a similar difference in the size of finite automata only due to the difference (i.e., l_1-norm or l_2-norm) in the probability calculation.

Quantum finite automata have been quite popular in the literature since its simplicity is nice to understand merits and demerits of quantum computation [AF98, AG00, AI99, ANTV99, KW97, Nay99]. Among these papers, the first important result by Kondacs and Watrous [KW97] also exploited the feature of l_2-norm, although they did not mention explicitly, when proving that 2-way qfa's can accept non-regular languages. Thus this scheme of exploiting the feature of l_2-norm could be another new technique which can bring out the power of quantum computation. Ambainis and Freivalds [AF98] also proved a quadratic (and even exponential) differences in the size of qfa's and pfa's for *one-letter* languages, but their technique is based on the rotation of complex numbers, which is completely different from ours.

2 Problem EQ

Suppose that Alice and Bob have n-bit numbers x and y and they wish to know whether or not $x=y$. This problem, called EQ, is one of the most famous problems for which its randomized communication complexity ($=\Theta(\log n)$) is significantly smaller than its deterministic communication complexity ($=n+1$) [KN97]. In this paper, we need a little bit more accurate argument on the value of randomized (and one-way) communication complexity: Consider the following protocol M_{EQ}: (i) Alice selects a single prime p among the smallest N primes. (ii) Then she divides x by p and sends Bob p and the residue

a. (iii) Bob also divides his number y by p and compares his residue with a. They accept (x, y) iff those residues coincide.

It is obvious that if $x = y$ then protocol M_{EQ} accepts (x, y) with probability one. Let $E(N)$ be the maximum (error) probability that M_{EQ} accepts (x, y) even if $x \neq y$. To compute $E(N)$, we need the following lemma: In this paper, $\log n$ always means $\log_2 n$ and $\lceil f(n) \rceil$ for a function $f(n)$ is simply written as $f(n)$.

Lemma 1. Suppose that $x \neq y$ and let S be a set of primes such that $x = y \bmod p$ for any p in S. Also, let s be the maximum size of such a set S for n-bit integers x and y. Then $s = \Theta(n/\log n)$.

Proof. Let p_i be the i-th largest prime and $\pi(n)$ be the number of different primes $\leq n$. Then the prime number theorem says that $\lim_{n\to\infty} \frac{\pi(n)}{n/\log_e n} = 1$, which means that $p_{n/\log n} = \Theta(n)$. Consequently, there must be a constant c such that $p_{n/\log n} \cdot p_{n/\log n+1} \cdot \cdots \cdot p_{cn/\log n} > 2^n$ since $n^{n/\log n} = 2^n$. Thus a n-bit integer z has at most $cn/\log n$ different prime factors. Note that $x = y \bmod a$ iff $|x-y| = 0 \bmod a$. Hence, $s \leq cn/\log n$. Also it turns out by the prime theorem that there is an n-bit integer z such that it has $c'n/\log n$ different prime factors for some constant c', which proves that $s \geq c'n/\log n$. ■

In this paper, N_0 denotes this number s which is $\Theta(n/\log n)$. Then

Lemma 2. $E(N)$ is N_0/N.

For example, if we use $N = n^2/\log n$ different primes in M_{EQ}, its error-rate is $1/n$.

3 Our Languages and qfa's

A one-way qfa is the following model: (i) Its input head always moves one position to the right each step. (ii) Global state transitions must be unitary. (iii) Its states are partitioned into *accepting*, *rejecting* and *non-halting* states. (iv) Observation is carried out every step, and if acceptance or rejection is observed, then the computation ends. Otherwise, computation continues after evenly distributing the amplitudes of accepting and rejecting states to non-halting states. We omit the details, see for example [KW97]. In this paper, we consider the following three languages.

$L_0(n) = \{w\sharp w^R \mid w \in \{0,1\}^n\}$,
$L_1(n) = \{w_1\sharp w_2\sharp\sharp w_3\sharp w_4\sharp \mid w_1, w_2, w_3, w_4 \in \{0,1\}^n$,
$(w_1 = w_2^R) \vee ((w_1w_2) = (w_3w_4)^R)\}$,
$L_2(n,k) = \{w_{11}\sharp w_{12}\sharp\sharp w_{13}\sharp w_{14}\sharp\sharp\sharp \cdots \sharp\sharp\sharp w_{i1}\sharp w_{i2}\sharp\sharp w_{i3}\sharp w_{i4}$ $\sharp\sharp\sharp \cdots \sharp\sharp\sharp w_{k1}\sharp w_{k2}\sharp\sharp w_{k3}\sharp w_{k4}\sharp \mid$
$w_{i1}, w_{i2}, w_{i3}, w_{i4} \in \{0,1\}^n, 1 \leq i \leq k$ and $1 \leq \exists j \leq k$
s.t. $(w_{j1} = w_{j2}^R) \wedge ($for all $1 \leq i \leq j-1$, $(w_{i1}w_{i2}) = (w_{i3}w_{i4})^R)\}$.

In the next section, we first construct a qfa M_0^Q, which accepts strings $x \in L_0$ with probability 1 and strings $y \notin L_0$ with probability at most $\frac{1}{n}$. M_0^Q simulates the protocol M_{EQ} in the following way (see Fig 1). Given an input string $¢w_1\sharp w_2\$$ ($¢$ is the leftmost and $\$$ is the rightmost symbols), M_0^Q first splits into N different states $q_{p_1}, \cdots, q_{p_i}, \cdots, q_{p_N}$ with the same amplitude by reading $¢$. Then from q_{p_i}, submachine M_{1i} divides integer w_1 by the i-th prime p_i. This computation ends up in some state of M_{1i} which corresponds to the residue of the division. This residue information is shifted to the next submachine M_{2i}, and then M_{2i} carries out a completely opposite operation while reading w_2. If (and only if) two residues are the same, M_{2i} ends up in some specific state q_i^0. M_0^Q then applies a Fourier transform from q_i^0 to s_i for $1 \leq i \leq N$. M_0^Q thus simulates M_{EQ} by setting s_N as its only accepting state.

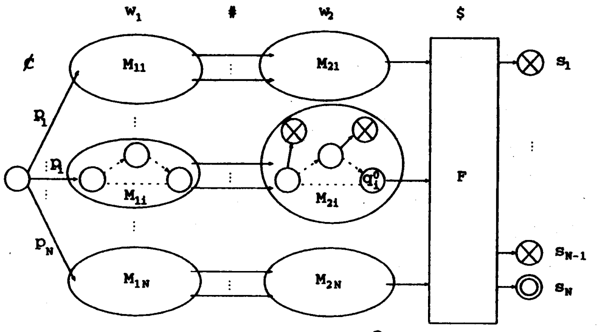

Fig 1. qfa M_0^Q

We can use exactly the same state transition for the probabilistic counterpart, pfa M_0^P, except for deterministic transitions from q_i^0 to s_N. As mentioned before we can achieve a quadratic difference in the probability of error, like $(1/n)^2$ for M_0^Q v.s. $(1/n)$ for M_0^P. This could be traded to a quadratic difference in the necessary number of different primes or a quadratic difference in the size of automata. However, note that we do not need this small (like $1/n$ or $1/n^2$) error rate, but something like 1/3 is enough, by the definition. Then here are some difficulties: First of all, it seems hard to calculate the number of states very accurately, i.e., the number of states which is just right to achieve this kind of constant error rate. Furthermore, even if we could do that, there would be no big difference in the size of automata that corresponds to the difference in the error probabilities between, e.g., 1/3 and 1/9.

There is a standard technique to overcome these difficulties, namely, the use of iteration. Consider the following string:

$$w_{11}\sharp w_{12}\sharp\sharp w_{21}\sharp w_{22}\sharp\sharp \cdots \sharp w_{n1}\sharp w_{n2}$$

where the accepting condition is that for some $1 \leq j \leq n$, $w_{j1} = w_{j2}^R$. When all pairs (w_{j1}, w_{j2}) do not satisfy this condition, the (error) probability of accepting such a string is roughly $O\left(\frac{1}{n}\right) \times n = O(1)$, which appears desirable for our purpose.

This argument does not seem to cause any problem for pfa's but it does for qfa's for the following reason: After checking w_{11} and w_{12}, the qfa is in a single accepting state if the condition is met, which is completely fine. However, if $w_{11} \neq w_{12}^R$ and the observation is not accepting, then there are many small amplitudes distributed to many different states. Note that we have to continue the calculation for w_{21} and w_{22} which should be started from a single state. (It may be possible to start the new computation from each non-halting state, but that will result in an exponential blow-up in the number of states and no clear separation in the size of automata either.) One can see easily that we cannot use a Fourier transform this time to gather the amplitudes since there are many different patterns in the distribution of states which have a small nonzero amplitudes.

This is the reason why the next language $L_1(n)$ plays an important role. Suppose that $w_1 \neq w_2^R$. Then the resulting distribution of amplitudes is quite complicated as mentioned above. However, no matter how it is complicated, we can completely "reverse" the computation for $w_1\sharp w_2$ by reading $w_3\sharp w_4$ if $(w_1w_2) = (w_3w_4)^R$. This reverse computation should end up in a single state of amplitude one (actually it is a little less than one) since the original computation for $w_1 \neq w_2^R$ starts from the (single) initial state. One can now see that the third language, $L_2(n,k)$, is exactly for the iteration purpose mentioned above.

4 Main Results

As mentioned in the previous section, we sequentially construct our qfa's and corresponding pfa's for $L_0(n)$, $L_1(n)$ and $L_2(n, n^c)$. Recall that N is the number of primes used in protocol M_{EQ} and $N_0 = \Theta(n/\log n)$.

Lemma 3. There exists a qfa M_0^Q which accepts strings in L_0 with probability one and strings not in L_0 with probability at most $\left(\frac{N_0}{N}\right)^2$. The number of states in M_0^Q is $\Theta(N^2 \log N)$.

Proof. M_0^Q has the following states: (i) An initial state q_0, (ii) $q_{p_k,j_k,1}$ (in submachine M_{1i} of Fig 1), (iii) $q_{p_k,j_k,2}$ (in M_{2i} of Fig 1), (iv) $q_{p_k,j_k,rej}$ (also in M_{2i} of Fig 1), (v) s_l, where $1 \leq k \leq N$, $1 \leq j_k \leq p_k - 1$ and $1 \leq l \leq N$. p_k denotes the k-th largest prime (but we exclude two from p_k for the reason mentioned later). s_N is only one accepting state, $q_{p_k,j_k,rej}$ and s_l $(1 \leq l \leq N-1)$ are rejecting states and all the others are non-halting states. We give a complete state transition diagram of M_0^Q in Table 1, where $V_\sigma|Q\rangle = \alpha_1|Q_1\rangle + \cdots + \alpha_i|Q_i\rangle + \cdots + \alpha_m|Q_m\rangle$ means that if M_0^Q reads symbol σ in state Q, it moves to each state Q_i with amplitude α_i ($|\alpha_1|^2 + \cdots + |\alpha_m|^2 = 1$).

When reading ¢ of the input string ¢w_1♯w_2\$, M_0^Q splits into N submachines (denoted by M_{1i} in Fig 1) with equal amplitudes (see transition (1) of Table 1). The k-th submachine M_k computes the residue when dividing w_1 by p_k (by using transition $(2-a)$ to $(2-d)$ in Table 1). This division can be done simply by simulating the usual division procedure as shown in Fig 2 (a) and (b) for $w_1 = 110001$ and $p_2 = 101$ $(= 5)$. State j in Fig 2 (b) corresponds to $q_{p_2,j,1}$. The starting state is 0 and by reading the first symbol 1 it goes to state 1. By reading the second symbol 1, it goes to state 3 $(= 11)$. Now reading 0, it goes to state 1 since $110 = 1 \bmod 101$. This continues until reading the last symbol 1 and M_0^Q ends up in state 4. It should be noted that these state transitions are reversible: For example, if the machine reaches state 2 $(= 10)$ from some state Q by reading 0, then Q must be state 1 since Q cannot be greater than 2. (Reason: If Q is greater than 2, it means that the quotient will be 1 after reading a new symbol. Since M_0^Q reads 0 as the new symbol, the least significant bit of the residue when divided by 5 must be 1, which excludes state 2 as its next state.) Hence the quotient must have been 0, and so the previous state must be 1. (Note that this argument holds because we excluded two from p_k which is only one even prime.)

Thus, if $w_1 \bmod p_k = j_k$, then M_0^Q is in superposition $\frac{1}{\sqrt{N}}\sum_{k=1}^{N}|q_{p_k,j_k,1}\rangle$ after M_0^Q read w_1. Then M_0^Q reads ♯ and this superposition is "shifted" to $\frac{1}{\sqrt{N}}\sum_{k=1}^{N}|q_{p_k,j_k,2}\rangle$, where M_0^Q checks if $w_2^R \bmod p_k$ is also j_k by using transition $(4-a)$ to $(4-d)$ in Table 1. This job can be done by completely reversing the previous procedure of dividing w_1 by p_k. Actually, the state transitions are obtained by simply reversing the directions of previous state diagrams. Since previous transitions are reversible, new transitions are also reversible. Now one can see that the k-th submachine M_{2k} is in state $q_{p_k,0,2}$ iff the two residues are the same.

Finally by reading \$, Fourier transform is carried out only from these zero-residue states $q_{p_k,0,2}$ to s_l. Other states $q_{p_k,j,2}$ $(j \neq 0)$ go to rejecting states $q_{p_k,j,rej}$. If the residues are the same in only t submachines out of the k ones, the amplitude of s_N is given as

$$\frac{1}{N}\sum_{|t|}\sum_{l=1}^{N}\exp\left(\frac{2\pi i}{N}kl\right)|s_l\rangle$$

$$= \frac{t}{N}|s_N\rangle + \frac{1}{N}\sum_{|t|}\sum_{l=1}^{N-1}\exp\left(\frac{2\pi i}{N}kl\right)|s_l\rangle,$$

which is equal to t/N. Thus the probability of acceptance is $\left(\frac{t}{N}\right)^2$. If the input string is in L_0, then this probability becomes 1. Otherwise, it is at most $(N_0/N)^2$ by Lemma 2. The number of states in M_0^Q is given as

$$1 + 2\sum_{k=1}^{N}p_k + \sum_{k=1}^{N}(p_k - 1) + N$$

$$= 1 + 3\sum_{k=1}^{N}p_k \leq 1 + 3 \cdot N \cdot p_N = O(N^2 \log N),$$

which completes the proof. ■

$$\begin{array}{ll}
(1) & V_{¢}|q_0\rangle = \frac{1}{\sqrt{N}}\sum_{k=1}^{N}|q_{p_k,0,1}\rangle, \\
(2-a) & V_0|q_{p_k,j,1}\rangle = |q_{p_k,2j,1}\rangle \quad (0 \leq j < \frac{p_k}{2}), \\
(2-b) & V_0|q_{p_k,j,1}\rangle = |q_{p_k,2j-p_k,1}\rangle \quad (\frac{p_k}{2} < j < p_k), \\
(2-c) & V_1|q_{p_k,j,1}\rangle = |q_{p_k,2j+1,1}\rangle \quad (0 \leq j < \frac{p_k}{2} - 1), \\
(2-d) & V_1|q_{p_k,j,1}\rangle = |q_{p_k,2j+1-p_k,1}\rangle \ (\frac{p_k}{2} - 1 < j < p_k), \\
(3) & V_{\sharp}|q_{p_k,j,1}\rangle = |q_{p_k,j,2}\rangle, \\
(4-a) & V_0|q_{p_k,j,2}\rangle = |q_{p_k,\frac{j}{2},2}\rangle \quad (j:\ even), \\
(4-b) & V_0|q_{p_k,j,2}\rangle = |q_{p_k,\frac{j+p_k}{2},2}\rangle \quad (j:\ odd), \\
(4-c) & V_1|q_{p_k,j,2}\rangle = |q_{p_k,\frac{j-1+p_k}{2},2}\rangle \quad (j:\ even), \\
(4-d) & V_1|q_{p_k,j,2}\rangle = |q_{p_k,\frac{j-1}{2},2}\rangle \quad (j:\ odd), \\
(5-a) & V_{\$}|q_{p_k,0,2}\rangle = \frac{1}{\sqrt{N}}\sum_{l=1}^{N}\exp\left(\frac{2\pi i}{N}kl\right)|s_l\rangle, \\
(5-b) & V_{\$}|q_{p_k,j,2}\rangle = |q_{p_k,j,rej}\rangle \quad (1 \leq j < p_k).
\end{array}$$

Table 1: State transition diagram of M_0^Q

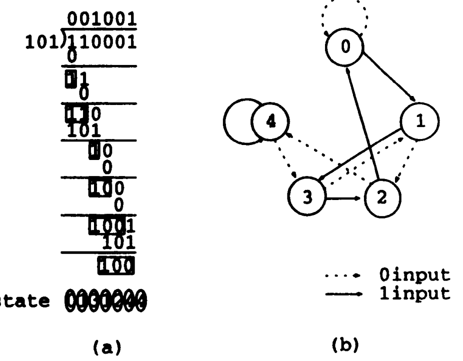

Fig 2. division procedure for $w_1 = 110001$ and $p_k = 5$

Let us consider the pfa whose state transition is exactly the same as M_0^Q of $f(N)$ states excepting that the state transitions from $q_{p_k,0,2}$ to s_l for Fourier transform are replaced by simple (deterministic) transitions from $q_{p_k,0,2}$ to s_N. We call such a pfa *emulates* the qfa. Suppose that M^P emulates M^Q. Then the size of M^P is almost the same as that of M^Q, i.e., it is also $\Theta(f(N))$ if the latter is $f(N)$, since the Fourier transform does not make much difference in the number of states.

Lemma 4. Suppose that M_0^P emulates M_0^Q. Then M_0^P accepts strings in L_0 with probability one and those not in L_0 with probability N_0/N.

Let us set, for example, $N = N_0\sqrt{n}$. Then the error-rate of M_0^Q is $(N_0/N)^2 = \frac{1}{n}$ and its size is $O(n^3/\log n)$. To achieve the same error-rate by a pfa, we have to set $N = N_0 n$, which needs $O(n^4/\log n)$ states.

Remark. Suppose that we have once designed a specific qfa M_0^Q (similar for M_0^P). Then it can work for inputs of any length or it does not reject the input only for the reason that its length is not $2n+1$. The above calculation of the acceptance and rejection rates is only true when our input is restricted to strings $\subseteq \{0,1\}^n\sharp\{0,1\}^n$.

Now we shall design a qfa M_1^Q which recognizes the second language $L_1(n)$. N_0' also denotes the number s in Lemma 1 but for x and y of length $2n$.

Lemma 5. There exists a qfa M_1^Q which accepts strings in L_1 with probability $1 - (\frac{N_0}{N_1})^2 + (\frac{N_0}{N_1})^4$ and strings not in L_1 with at most $(\frac{N_0}{N_1})^2 + (\frac{N_0'}{N_2})^2 \cdot (1 - \frac{N_0}{N_1})^2 \cdot (1 + \frac{N_0}{N_1})^2$. M_1^Q has $\Theta((N_1 N_2)^2 \log N_1 \cdot \log N_2)$ states.

Proof. Again a complete state transition diagram is shown in Table 2, where accepting states are $s_{N_1,0,p_l,f}$ such that $0 \le f \le p_l - 1$ and t_{N_1}. Rejecting states are $q_{p_k,e,p_l,f,rej}$ such that $e \ne 0$ or $f \ne 0$, $0 \le e \le p_k - 1$, $0 \le f \le p_l - 1$, $t_{p_k,0,y}$ such that $1 \le y \le N_2 - 1$, and t_z such that $1 \le z \le N_1 - 1$. All other states are non-halting.

M_1^Q checks whether $w_1 = w_2^R$ using N_1 primes and also whether $(w_1 w_2) = (w_3 w_4)^R$ using N_2 primes. Note that those two jobs have to be done at the same using composite automata while reading $w_1\sharp w_2$. Hence M_1^Q first splits into $N_1 \cdot N_2$ submachines, each of which is denoted by $M(k,l)$, $1 \le k \le N_1$, $1 \le l \le N_2$. As shown in Fig 3, $M(k,l)$ has six stages, from stage 1 thorough stage 6. It might be convenient to think that each state of $M(k,l)$ be a pair of state (q_L, q_R) and to think $M(k,l)$ be a composite of M_L and M_R. In stages 1 and 2, M_L has a similar state transitions to those of Table 1 for checking $w_1 \ne w_2^R$. M_R has also similar transitions but only for the first part of it, i.e., to compute $w_1 w_2 \bmod p_l$. This portion of transitions are given in (2) to (4) of Table 2.

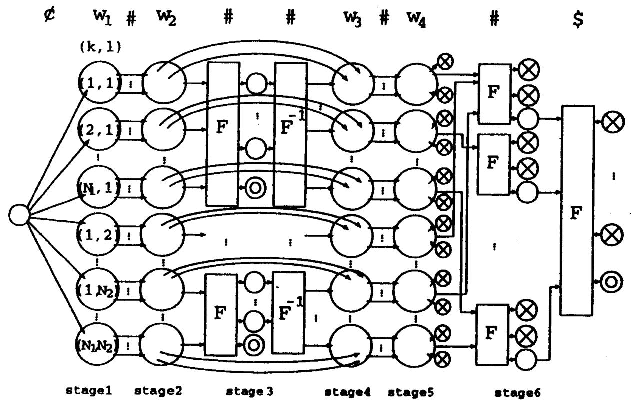

Fig 3. qfa M_1^Q

Now we go to stage 3. Here M_L, reading the first $\sharp$, carries out the Fourier transform exactly as M_0^Q (see $(5-a)$ in Table 2). After that M_L, reading the second $\sharp$, execute Inverse Fourier transform from states $s_{m,0,p_l,f}$ $(1 \le m \le N_1 - 1)$, i.e., from the states after the first Fourier transform excepting the accepting states, which is shown in $(6-a)$ of Table 2. In this stage, M_R does nothing; it just shifts the state information about $(w_1 w_2) \bmod p_l$ (but only when $w_1 \ne w_2^R$) to stage 4.

Stages 4 and 5 are for the complete reverse operation of stages 2 and 1. By doing this, the amplitudes for state q_L, which were once in turmoil after stage 2, are reorganized and gathered in specific states, namely $q_{p_k,0,p_l,0,4}$ if $(w_1 w_2) = (w_3 w_4)^R$. Therefore, what we do is to gather the amplitude of $q_{p_k,0,p_l,0,4}$ to $t_{p_k,0,N_2}$ by Fourier transform reading $\sharp$. Now reading the rightmost symbol, we do another Fourier transform, which gathers the amplitudes of $t_{p_k,0,N_2}$ to t_{N_1}.

The analysis of error probability is a little bit complicated. The basic idea is as follows: When $w_1 \ne w_2^R$, a small amplitude, $\frac{1}{\sqrt{N_2}}\frac{N_0}{N_1}$ is "taken" by each of the N_2 accepting states in stage 3. This is basically the same as M_0^Q since its probability of observation is $\sum_{l=1}^{N_2}\left(\frac{1}{\sqrt{N_2}} \cdot \frac{N_0}{N_1}\right)^2 = \left(\frac{N_0}{N_1}\right)^2$. So, the problem is how much of the remaining amplitudes distributed on other states in this stage can be retrieved in the final accepting state t_{N_1} when $(w_1 w_2) = (w_3 w_4)^R$. If we could retrieve 100%, then the accepting probability at state t_{N_1} when $(w_1 w_2) = (w_3 w_4)^R$ is $1 - \left(\frac{N_0}{N_1}\right)^2$. Unfortunately, we cannot do that but the loss is very small and our accepting probability at state t_{N_1} is

$$\left(1 - \left(\frac{N_0}{N_1}\right)^2\right)^2 = 1 - 2\left(\frac{N_0}{N_1}\right)^2 + \left(\frac{N_0}{N_1}\right)^4,$$

which turns out to be enough for our purpose. ■

Suppose that we set $N_1 = N_0\sqrt{n}$, and $N_2 = dN_0'$. Then $\frac{N_0}{N_1} = \frac{1}{\sqrt{n}}$ and $\frac{N_0'}{N_2} \le \frac{1}{2}$ if we select a sufficiently large constant d. Namely, M_1^Q accepts strings in L_1 with probability $1 - \frac{1}{n} + \frac{1}{n^2}$ and those not in L_1 with probability at most $\frac{1}{4} + \frac{1}{2n} + \frac{1}{n^2}$. The number of states is $\Theta(\frac{n^5}{\log^2 n})$. The probability distribution on acceptance and rejection in each state is illustrated in Fig 4.

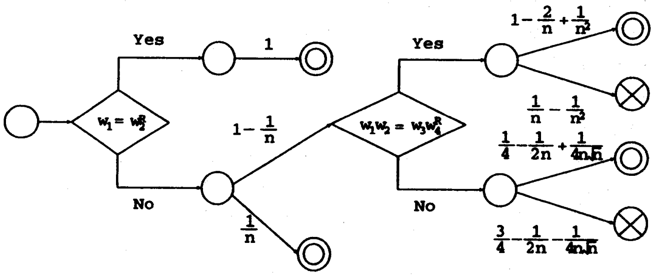
Yes
1
No
$w_1 = w_2^R$
$1-\frac{1}{n}$
$\frac{1}{n}$
Yes
No
$w_1w_2 = w_3w_4^R$
$1-\frac{2}{n}+\frac{1}{n^2}$
$\frac{1}{n}-\frac{1}{n^2}$
$\frac{1}{4}-\frac{1}{2n}+\frac{1}{4n\sqrt{n}}$
$\frac{3}{4}-\frac{1}{2n}-\frac{1}{4n\sqrt{n}}$

Fig 4. probability distribution when $N_1 = N_0\sqrt{n}$, $N_2 = dN_0'$

Let us consider pfa M_1^P which recognizes $L_1(n)$. The state transition of M_1^P is the same as that of M_1^Q except Fourier transform and Inverse Fourier transform only M_1^Q performs. If string x satisfies $w_1 \ne w_2^R$, then M_1^P accepts x with at most probability $\frac{N_0}{N_1}$ after reading $w_1\sharp w_2$. It should be noted that M_1^Q accepts such a string with at most probability $\left(\frac{N_0}{N_1}\right)^2$ after reading $w_1 \sharp w_2$.

Lemma 6. Suppose that M_1^P emulates M_1^Q. Then M_1^P accepts strings in L_1 with probability 1 and those not in L_1 with probability at most $\frac{N_0}{N_1} + \left(1 - \frac{N_0}{N_1}\right) \cdot \frac{N_0'}{N_2}$. The number of states is approximately the same, i.e., $\Theta((N_1 N_2)^2 \log N_1 \log N_2)$. (Proof is omitted.)

If we set $N_1 = N_0 n$ and $N_2 = dN_0'$, then strings such that $w_1 \neq w_2^R$ are accepted with probability at most $\frac{1}{n}$ after reading $w_1 \sharp w_2$. Thus this probability is the same as the qfa such that $N_1 = N_0\sqrt{n}$ and $N_2 = dN_0'$, but number of states of this pfa is $\Omega\left(\frac{n^6}{\log^2 n}\right)$.

Now we are ready to give our main theorem:

Theorem 1. For any integer c, there is a qfa M^Q such that M^Q recognizes $L(n, n^c)$ and the number of states in M^Q is $O\left(\frac{n^{c+4}}{\log^2 n}\right)$.

Proof. The construction of M^Q is easy: We just add a new deterministic transition from the last accepting state in stage 6 of M_1^Q to its initial state by $\sharp$, by which we can manage iteration. Also, we need some small changes to manage the very end of the string: Formally speaking, transition (11) in Table 2 is modified into

$$V_\sharp |t_{p_k,0,N_2}\rangle = \frac{1}{\sqrt{N_1}} \sum_{z=1}^{N_1} \exp\left(\frac{2\pi i}{N_2} kz\right) |t_z\rangle,$$

t_{N_1} is now not an accepting state but a non-halting state and two new transitions

$$\begin{array}{ll} (10-c) & V_\$ |q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,e,p_l,f,rej}\rangle \\ (12) & V_\sharp |t_{N_1}\rangle = |q_1\rangle \end{array}$$

are added.

We set $N_1 = 2N_0 n^{c/2}$ and $N_2 = dN_0'$. Then $N_0/N_1 = \frac{1}{2n^{c/2}}$ and $N_0'/N_2 < \frac{1}{2}$ if we select a sufficiently large constant as d. Suppose that M^Q has not stopped yet and is now reading the i-th block $w_{i1}\sharp w_{i2}\sharp\sharp w_{i3}\sharp w_{i4}$. Then, we can conclude the following by Lemma 5: (i) If $w_{i1} = w_{i2}^R$, then M^Q accepts the input with probability one. (ii) If $(w_{i1}w_{i2}) = (w_{i3}w_{i4})^R$, then $(ii-a)$ M^Q also accepts the input with probability $1/4n^c$ and $(ii-b)$ rejects the input with $\frac{1}{4n^c} - \frac{1}{16n^{2c}}$ and $(ii-c)$ goes back to the initial state with $1 - \frac{2}{4n^c} + \frac{1}{16n^{2c}}$. ($iii$) If $(w_{i1}w_{i2}) \neq (w_{i3}w_{i4})^R$, then $(iii-a)$ M^Q accepts the input with $\frac{1}{4n^c}$, $(iii-b)$ rejects it with $\frac{3}{4} - \frac{1}{8n^c} - \frac{1}{16n^{2c}}$ and $(iii-c)$ goes back to the initial state with $\frac{1}{4} - \frac{1}{8n^c} + \frac{1}{16n^{2c}}$. The number of state is $O(n^{c+4}/\log^2 n)$.

Recall that the number of iteration is n^c and suppose that the input x is in $L(n, n^c)$. Then, the probability that x is rejected is equal to the probability that $(ii-b)$ happens before (i) happens. The probability that $(ii-b)$ happens is at most $\frac{1}{4n^c}$ per iteration, and so the probability that $(ii-b)$ happens in some iteration is at most $n^c \cdot \frac{1}{4n^c} = \frac{1}{4}$. Therefore the probability that x is finally accepted is well larger than 1/2. Suppose conversely that x is not in $L(n, n^c)$. Then the probability that $(ii-a)$ happens in some iteration is the same as above and is at most $\frac{1}{4}$. If M^Q does not meet a block such that $(w_{i1}w_{i2}) \neq (w_{i3}w_{i4})^R$ until the end, then the accepting probability is at most this 1/4. If M^Q does meet such a block in some iteration, it rejects x with probability at least $(1 - \frac{1}{4})(\frac{3}{4} - \frac{1}{8n^c} - \frac{1}{16n^{2c}})$ which is again well above 1/2. Thus M^Q recognizes $L(n, n^c)$. ■

Theorem 2. Suppose that M^P which emulates M^Q recognizes $L(n, n^c)$. Then the number of states of M^P is $\Omega(n^{2c+4}/\log^2 n)$.

Proof. M^P is constructed by applying a similar modification as above to M_1^P. Then it turns out from Lemma 6 that if we set $N_1 = \frac{1}{a}N_0 n^c$, $N_2 = dN_0'$ then the number of states in M_1^P is $\Omega(n^{2c+4}/\log^2 n)$. Now suppose that the input x includes a long repetition of blocks such that $(w_{i1}w_{i2}) = (w_{i3}w_{i4})^R$. Then x is accepted in each iteration with probability a/n^c. Therefore the probability that this happens in the first k iterations is

$$\sum_{i=1}^{k} \left(1 - \frac{a}{n^c}\right)^{x-1} \cdot \frac{a}{n^c} = 1 - \left(1 - \frac{a}{n^c}\right)^k.$$

Since the number of repetitions ($= k$) can be as large as n^c,

$$\lim_{n \to \infty} \left(1 - \frac{a}{n^c}\right)^{n^c} = \frac{1}{e^a}.$$

Thus if we select a sufficiently large constant a, then the probability of acceptance can be almost 1. Such a M^P cannot recognize $L(n, n^c)$ obviously, which proves the theorem. ■

5 Concluding Remarks

The question in this paper is whether or not we can exploit the difference in probability calculation between quantum and probabilistic computations. We have shown that the answer is yes using quantum finite automata. However, what remains apparently is whether or not we can exploit this property for other types of models and/or for other types of problems which are preferably less artificial. Also it should be an important future research to obtain a general lower bound for the number of states which is needed to recognize $L_2(n, n^c)$ by pfa's.

Acknowledgment. We are grateful to Mario Szegedy for these many valuable comments to this research.

$$\begin{array}{ll}
(1) & V_{\not{c}} |q_0\rangle = \frac{1}{\sqrt{N_1 N_2}} \sum_{k=1}^{N_1} \sum_{l=1}^{N_2} |q_{p_k,0,p_l,0,1}\rangle, \\
(2-a) & V_0 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e,p_l,2f,1}\rangle \\
 & \qquad (0 \le e < \frac{p_k}{2}, \quad 0 \le f < \frac{p_l}{2}), \\
(2-b) & V_0 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e,p_l,2f-p_l,1}\rangle \\
 & \qquad (0 \le e < \frac{p_k}{2}, \quad \frac{p_l}{2} < f < p_l), \\
(2-c) & V_0 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e-p_k,p_l,2f,1}\rangle \\
 & \qquad (\frac{p_k}{2} < e < p_k, \quad 0 \le f < \frac{p_l}{2}), \\
(2-d) & V_0 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e-p_k,p_l,2f-p_l,1}\rangle \\
 & \qquad (\frac{p_k}{2} < e < p_k, \quad \frac{p_l}{2} < f < p_l), \\
(2-e) & V_1 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e+1,p_l,2f+1,1}\rangle \\
 & \qquad (0 \le e < \frac{p_k}{2} - 1, \quad 0 \le f < \frac{p_l}{2} - 1), \\
(2-f) & V_1 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e+1,p_l,2f+1-p_l,1}\rangle \\
 & \qquad (0 \le e < \frac{p_k}{2} - 1, \quad \frac{p_l}{2} - 1 < f < p_l), \\
(2-g) & V_1 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e+1-p_k,p_l,2f+1,1}\rangle \\
 & \qquad (\frac{p_k}{2} - 1 < e < p_k, \quad 0 \le f < \frac{p_l}{2} - 1), \\
(2-h) & V_1 |q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,2e+1-p_k,p_l,2f+1-p_l,1}\rangle \\
 & \qquad (\frac{p_k}{2} - 1 < e < p_k, \quad \frac{p_l}{2} - 1 < f < p_l),
\end{array}$$

(3) $V_{\sharp}|q_{p_k,e,p_l,f,1}\rangle = |q_{p_k,e,p_l,f,2}\rangle,$

(4 − a) $V_0|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e}{2},p_l,2f,2}\rangle$ $(e:\ even,\quad 0 \le f < \frac{p_l}{2}),$

(4 − b) $V_0|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e}{2},p_l,2f-p_l,2}\rangle$ $(e:\ even,\quad \frac{p_l}{2} < f < p_l),$

(4 − c) $V_0|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e+p_k}{2},p_l,2f,2}\rangle$ $(e:\ odd,\quad 0 \le f < \frac{p_l}{2}),$

(4 − d) $V_0|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e+p_k}{2},p_l,2f-p_l,2}\rangle$ $(e:\ odd,\quad \frac{p_l}{2} < f < p_l),$

(4 − e) $V_1|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e-1+p_k}{2},p_l,2f+1,2}\rangle$ $(e:\ even,\quad 0 \le f < \frac{p_l}{2}-1),$

(4 − f) $V_1|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e-1+p_k}{2},p_l,2f+1-p_l,2}\rangle$ $(e:\ even,\quad \frac{p_l}{2}-1 < f < p_l),$

(4 − g) $V_1|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e-1}{2},p_l,2f+1,2}\rangle$ $(e:\ odd,\quad 0 \le f < \frac{p_l}{2}-1),$

(4 − h) $V_1|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,\frac{e-1}{2},p_l,2f+1-p_l,2}\rangle$ $(e:\ odd,\quad \frac{p_l}{2}-1 < f < p_l),$

(5 − a) $V_{\sharp}|q_{p_k,0,p_l,f,2}\rangle = \frac{1}{\sqrt{N_1}}\sum_m \exp(\frac{2\pi i}{N_1}km)|s_{m,0,p_l,f}\rangle,$

(5 − b) $V_{\sharp}|q_{p_k,e,p_l,f,2}\rangle = |q_{p_k,e,p_l,f}\rangle \quad (1 \le e < p_k),$

(6 − a) $V_{\sharp}|s_{m,0,p_l,f}\rangle = \frac{1}{\sqrt{N_1}}\sum_r \exp(-\frac{2\pi i}{N_1}mr)|q_{p_r,0,p_l,f,3}\rangle$ $(1 \le m \le N_1 - 1),$

(6 − b) $V_{\sharp}|q_{p_k,e,p_l,f}\rangle = |q_{p_k,e,p_l,f,3}\rangle \quad (1 \le e < p_k),$

(7 − a) $V_0|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e,p_l,\frac{f}{2},3}\rangle$ $(0 \le e < \frac{p_k}{2},\quad f:\ even),$

(7 − b) $V_0|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e,p_l,\frac{f+p_l}{2},3}\rangle$ $(0 \le e < \frac{p_k}{2},\quad f:\ odd),$

(7 − c) $V_0|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e-p_k,p_l,\frac{f}{2},3}\rangle$ $(\frac{p_k}{2} < e < p_k,\quad f:\ even),$

(7 − d) $V_0|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e-p_k,p_l,\frac{f+p_l}{2},3}\rangle$ $(\frac{p_k}{2} < e < p_k,\quad f:\ odd),$

(7 − e) $V_1|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e+1,p_l,\frac{f-1+p_l}{2},3}\rangle$ $(0 \le e < \frac{p_k}{2}-1,\quad f:\ even),$

(7 − f) $V_1|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e+1,p_l,\frac{f-1}{2},3}\rangle$ $(0 \le e < \frac{p_k}{2}-1,\quad f:\ odd),$

(7 − g) $V_1|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e+1-p_k,p_l,\frac{f-1+p_l}{2},3}\rangle$ $(\frac{p_k}{2}-1 < e < p_k,\quad f:\ even),$

(7 − h) $V_1|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,2e+1-p_k,p_l,\frac{f-1}{2},3}\rangle$ $(\frac{p_k}{2}-1 < e < p_k,\quad f:\ odd),$

(8) $V_{\sharp}|q_{p_k,e,p_l,f,3}\rangle = |q_{p_k,e,p_l,f,4}\rangle,$

(9 − a) $V_0|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e}{2},p_l,\frac{f}{2},4}\rangle$ $(e:\ even,\quad f:\ even),$

(9 − b) $V_0|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e}{2},p_l,\frac{f+p_l}{2},4}\rangle$ $(e:\ even,\quad f:\ odd),$

(9 − c) $V_0|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e+p_k}{2},p_l,\frac{f}{2},4}\rangle$ $(e:\ odd,\quad f:\ even),$

(9 − d) $V_0|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e+p_k}{2},p_l,p_l,\frac{f+p_l}{2},4}\rangle$ $(e:\ odd,\quad f:\ odd),$

(9 − e) $V_1|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e-1+p_k}{2},p_l,\frac{f-1+p_l}{2},4}\rangle$ $(e:\ even,\quad f:\ even),$

(9 − f) $V_1|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e-1+p_k}{2},p_l,\frac{f-1}{2},4}\rangle$ $(e:\ even,\quad f:\ odd),$

(9 − g) $V_1|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e-1}{2},p_l,\frac{f-1+p_l}{2},4}\rangle$ $(e:\ odd,\quad f:\ even),$

(9 − h) $V_1|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,\frac{e-1}{2},p_l,\frac{f-1}{2},4}\rangle$ $(e:\ odd,\quad f:\ odd),$

(10 − a) $V_{\sharp}|q_{p_k,0,p_l,0,4}\rangle = \frac{1}{\sqrt{N_2}}\sum_y \exp(\frac{2\pi i}{N_2}ly)|t_{p_k,0,y}\rangle,$

(10 − b) $V_{\sharp}|q_{p_k,e,p_l,f,4}\rangle = |q_{p_k,e,p_l,f,rej}\rangle$ $(1 \le f < p_l),$

(11) $V_{\$}|t_{p_k,0,N_2}\rangle = \frac{1}{\sqrt{N_1}}\sum_{z=1}^{N_1} \exp\left(\frac{2\pi i}{N_2}kz\right)|t_z\rangle,$

Table 2: state transition diagram of M_1^Q

Reference

[AF98] A. Ambainis and R. Freivalds, "1-way quantum finite automata: strengths, weaknesses and generalizations," *Proceedings of the 39th IEEE Conference on Foundations of Computer Science*, 332-341, 1998.

[AG00] F. Ablayev and A. Gainutdinova, "On the Lower Bounds for One-Way Quantum Automata", *Proceedings of the 25th International Symposium on Mathematical Foundations of Computer Science*, 132-140, 2000.

[AI99] M. Amano and K. Iwama, "Undecidability on Quantum Finite Automata", *Proceedings of the 31st ACM Symposium on Theory of Computing*, 368-375, 1999.

[ANTV99] A. Ambainis, A. Nayak, A. Ta-Shma and U. Vazirani, "Dense quantum coding and a lower bound for 1-way quantum automata", *In Proceedings of the 31st ACM Symposium on Theory of Computing*, 376-383, 1999.

[BV97] E. Bernstein and U. Vazirani, "Quantum complexity theory", *SIAM Journal on Computing*, volume 26, number 5, 1411-1473, 1997.

[DJ92] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation", *Proceedings of the Royal society of London*, Series A, volume 439, 553-558, 1992.

[Gro96] L. Grover, "A fast quantum mechanical algorithm for database search," *Proceedings of the 28th ACM Symposium on Theory of Computing*, 212-219, 1996.

[KN97] E. Kushilevitz and N. Nisan, "Communication Complexity", *Cambridge University Press*, 1997.

[KW97] A. Kondacs and J. Watrous, "On the power of quantum finite state automata," *Proceedings of the 38th IEEE Conference on Foundations of Computer Science*, 66-75, 1997.

[Nay99] A. Nayak, "Optimal lower bounds for quantum automata and random access codes", *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, 369-376, 1999.

[Sho94] P. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, 124-134, 1994.

[Sim94] D. Simon, "On the power of quantum computation" *Proceedings of the 35th IEEE Conference on Foundations of Computer Science*, 116-123, 1994.