How to Exploit the Difference in Probability Calculation between Quantum and Probabilistic Computations

Masami Amano[*†‡] and Kazuo Iwama[*†]
[*] Quantum Computation and Information, ERATO, Japan Science and Technology Corporation (JST)
[†] School of Infomatics, Kyoto University, Kyoto, Japan
([‡] presently IBM Tokyo Research Laboratory, Kanagawa, Japan)

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. 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. (ii) Negative values for amplitudes are allowed, which can be used, for instance, to cancel other amplitudes and to "shift" some amplitude from one state to another. (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.

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 $1 \le i \le 10$, 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. We give an affirmative answer to this question by showing the following two theorems.

We consider the problem such 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 ($= n + 1$). For the problem, we 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 $x \ne y$. To compute $E(N)$, we need the following lemma: In the followings, $\log n$ 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 $x \ne y$ and let S be a set of primes such that $x = y \bmod p$ for any p in S. Also, let N_0 be the maximum size of such a set S for n-bit integers x and y. Then $N_0 = \Theta(n/\log n)$.

Theorem 1. For any integer c, there is a one-way quantum finite automaton (qfa) M^Q such that M^Q recognizes $L(n, n^c)$ and the number of states in M^Q is $O\left(\frac{n^{c+4}}{\log^2 n}\right)$, where $L(n, k)$ is defined as follows:
$L(n,k) = \{w_{11}\sharp w_{12}\sharp\sharp w_{13}\sharp w_{14}\sharp\sharp\sharp \cdots \sharp\sharp\sharp w_{i1}\sharp w_{i2}\sharp\sharp w_{i3}\sharp w_{i4}\sharp\sharp\sharp \cdots \sharp\sharp\sharp w_{k1}\sharp w_{k2}\sharp\sharp w_{k3}\sharp w_{k4}\sharp \mid w_{i1},\, w_{i2},\, w_{i3},\, w_{i4} \in \{0,1\}^n,$ $1 \le i \le k$ and $1 \le \exists j \le k$ s.t. $(w_{j1} = w_{j2}^R) \wedge$ (for all $1 \le i \le j-1$, $(w_{i1}w_{i2}) = (w_{i3}w_{i4})^R)\}$.

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

Our main idea to costruct M^Q in the above theorem is to simulate the protocol M_{EQ} and to gather amplitudes of the accepting states by applying a Fourier transform.

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. 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.

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