A New Computational Model: The Quantum Chaotic Turing Machine

Matthew Mains
00089096

December 11, 2005

Abstract

As we saw with my paper review computer science is starting to look towards the field of chaos for ideas and new ways to develop existing ideas. The paper review looked at the effect of a chaotic function on a hash function and hash tables with a positive result. In this paper I will discuss a totally new idea that is even more impressive than before: solving an NP-Complete problem in polynomial time. The basis for this is a new theoretical model called the quantum chaotic turing machine. This paper is based on the work of Ohya and Volovich entitled *New quantum algorithm for studying NP-complete problems*. I will present the results of that paper and combine it with other results found in other publications to try and fully flush out the ideas presented therein.

1 Introduction

In the last twenty five years we have seen major steps in the realm of quantum computing: the exponential speed-up for Simon's problem, perfectly secure communication and key distribution, Shor's polynomial time factoring algorithm ([2]) and the construction of a twelve qubit computer. However in spite of all these advances we still do not know exactly how powerful a quantum Turing machine can be. There is strong evidence (see [3]) that quantum Turing machines will be unable to solve NP-Complete problems efficiently. In this paper I will introduce and explain a new model of computation that has the theoretical ability to solve NP-Complete problems in polynomial time (in the size of the input).

2 Background

2.1 Notion of NP-Completeness

In theoretical computer science we can group problems by their relative difficulty. Most problems that are studied today are in a class of problems called NP. We define a problem, R, to be a *decision problem* if given an input x, outputs 'yes' if and only

if there exists an object y such that the property R(x,y) holds and 'no' otherwise. We define the set NP to be the set of all decision problems where the object y is polynomial in the size of x, and the property R(x,y) can be checked in polynomial time on a Turing machine. The set P is defined to be the set of all decision problems that are solvable (and verifiable) in polynomial time of a Turing machine. It is immediately clear that we have the following relationship: $P \subseteq NP$. However, it is not known if $NP \subseteq P$ or $NP \nsubseteq P$.

We must now consider the hardest problems in the set NP called NP-Complete problems (called NPC). The first problem that was found to be NP-Complete was Satisfiability (or SAT for short) by Cook in 1971 (see [5]). In the same paper Cook presented a method by which other problems could be 'reduced' to the SAT problem which would show that those problems were also in NPC. It is for these reasons we will consider only the SAT problem in this paper: it is the original problem in NPC and all other NPC problems reduce to it. Hence if there was a method for solving the SAT problem in polynomial time, then we would have a polynomial time algorithm to solve other problems in NPC (see [6] for other members of NPC).

2.2 Quantum Information Processing

In order to understand the motivations behind the quantum chaotic Turing machine (QCTM) there are certain principles of quantum information that need to be understood. This section will try to present a quick summary of all topics relevant to understanding the QCTM however refer to [7] for an in-depth look at the topics presented.

Quantum information processing involves encoding information in qubits that have the ability to be not only in classical states (either 0 or 1) but also in a superposition state. The general form of a single qubit is $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = [\alpha\ \beta]^T$, where $\alpha^2 + \beta^2 = 1$. You can think of α^2 and β^2 as the probabilities of the qubit collapsing to the states 0 and 1 respectively when a measurement is applied. To form multiple qubit states we take the Tensor product of the relevant states. To perform operations on qubit states we can (without loss of generality) apply reversible unitary operations. A unitary operation is one which can be represented by a unitary matrix. One of the benefits of quantum computation is the ability to create queries in superposition, and exploit quantum mechanical properties to solve the desired problem. The general form of a query to a function f is as follows: $f(\,|\mathbf{x}, y\rangle) = |\mathbf{x}, f(\mathbf{x}) + y\rangle$.

After applying a function to a state in a superposition we need to measure it in order to get a result. Recall that measuring collapses a qubit to a classical state based on the coefficients. Ideally to distinguish between two states we would want them to be orthogonal. However this is not always possible, so if we have two states that are almost orthogonal we can guess with fairly good certainty which state we had based on the output. In the one qubit scenario you can consider a superposition to be a vector of unit length, and a measurement to be a projection onto one of the x,y axis. So almost orthogonal vectors will be distinguishable with high probability. But consider two vectors, one on the x-axis and the other at a small angle. The projection of the second vector onto the y-axis is minimal and hence repeated measurements will most likely be unable to distinguish between the two. This is the motivation for the introduction of quantum chaotic Turing machine.

3 Quantum Chaotic Turing Machine

3.1 Motivation

As we previously mentioned the current model of quantum computation, the quantum Turing machine, is believed not to be powerful enough to solve NPC problems. Though we have been able to solve problems efficient that are (currently) outside of P, but not quite in NPC (see [2]) it does not seem possible to solve NPC problems in polynomial time using current models of computation. Thus we introduce a new model of computation that is powerful enough to solve NPC problems in polynomial time. This would (in theory, not in practice) collapse the hierarchy of problems (NP,NPC) to just P (recall P is the set of problems solvable in polynomial time).

3.2 Quantum Algorithm

The proposed computation model involves using a traditional quantum Turing machine to perform an algorithm and sending the output to a second 'computer' that would perform computations based on the logistic equation to 'amplify' the results.

Based on previous work by Masuda and Ohya (see [8]) there is a unitary operator, U_f, that solves the SAT problem. The paper proposes the following steps:

1. Create the superposition
$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in(0,1)^n} |\mathbf{x}, 0\rangle$$
by applying Hadamard gates,
2. Apply U_f to obtain
$$|\psi_f\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in(0,1)^n} |\mathbf{x}, f(\mathbf{x})\rangle$$

We now want to measure the last qubit of $|\psi_f\rangle$, and with probability of $\frac{r}{2^n}$ we will measure a 1 (where r is the number of solutions to the SAT problem). If r is large enough we will be able to detect it through repeating the algorithm a number of times. However, if r is small (close to 0), we will be unable to detect if there are solutions to the SAT problem.

To simplify the analysis we will reduce the problem to a one qubit problem. After step 2 above we have

$$|\psi_f\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in(0,1)^n} |\mathbf{x}, f(\mathbf{x})\rangle = \sqrt{1-q}\,|\chi_0\rangle \otimes |0\rangle + \sqrt{q}\,|\chi_1\rangle \otimes |1\rangle$$

where $|\chi_0\rangle$ and $|\chi_1\rangle$ are normalized n-qubit states and $q = \frac{r}{2^n}$. So we have basically reduced the n-qubit problem to a one qubit system:

$$|\psi'\rangle = \sqrt{1-q}\,|0\rangle + \sqrt{q}\,|1\rangle$$

where we want to distinguish between the cases when $q = 0$ and $q > 0$ but q very small.

Using a previous result from [3] we can see that since the inner product between the two cases (where $q = 0$ vs $q > 0, q << 1$) is very small then it is not possible to distinguish between the two reliably. However, using the quantum chaos computer it is possible. We take the output of traditional quantum Turing machine and use it as input to a second computer that evolves the state by applying iterations of the logistic equation. There is more quantum theory involved, but it is beyond the scope of this paper.

3.3 Logistic Equation

This section is perhaps the most relevant to chaos in the entire paper. Recall the logistic equation:

$$Q_\mu(x) = \mu x(1 - x)$$

We saw in class that for certain values of μ this has attractive fixed points. Since we want our equation to be chaotic, attractive points are not necessarily a good thing. We do however notice that $Q_\mu(x)$ has a fixed point at $x = 0$ which is repelling for $\mu > 1$, which is good since we do not want our computer to change an amplitude of zero to a non-zero amplitude. The paper by Ohya and Volovich suggest using $\mu = 3.71$. The reasoning behind this is that this value of μ provides a function that is chaotic. Recall from class that we say a function, f, is chaotic if it has the following properties:

1. f has sensitive dependence on initial conditions
2. the Lyapunov exponent, $\lambda(x) > 1$ for all $x \in domain(f)$

But since we are only interested in distinguishing between the cases of $x = 0$ and $x > 0$, it will be sufficient to show that there is an n such that $Q_\mu^{[n]}(x) > \frac{1}{2}$.[1]

Claim:[2] Let $x_0 = \frac{1}{2^n}$ $(n > 0)$, $x_i = Q_\mu^{[i]}(x_0)$ with $\mu = 3.71$. Then there exists $m \in \{0, 1, 2, \ldots, 2n\}$ such that $x_m \geq \frac{1}{2}$.

[1] Originally I planned to show that $Q_\mu(x)$ was chaotic. However, I quickly discovered that this would be very difficult since it has an attractive fixed point at $\frac{\mu-1}{\mu}$. And it would have been overkill for this application of the logistic function

[2] This is not original work

Proof: Suppose that such an m does not exist. Then for all $m \in [0, 2n]$ we have that $x_m < \frac{1}{2}$. So

$$\frac{1}{2} > x_m = \mu x_{m-1}(1 - x_{m-1}) \geq \frac{\mu}{2} x_{m-1}$$

And therefore we have through recursively apply the above

$$\frac{1}{2} > \left(\frac{\mu}{2}\right)^m x_0 = \left(\frac{\mu}{2}\right)^m \frac{1}{2^n}$$

which implies the following relation

$$2^{n+m-1} > \mu^m$$

and so,

$$\begin{aligned} \Rightarrow n + m - 1 &> m \cdot \log \mu \\ \Rightarrow \frac{n-1}{\log \mu - 1} &> m \\ \Rightarrow \frac{1}{\log \mu - 1}(n-1) &> m \end{aligned}$$

But

$$2n > \frac{1}{\log \mu - 1}(n-1) > m,$$

which is a contradiction. So there is $m \in [0, 2n]$ with $x_m \geq \frac{1}{2}$. ■

This claim is significant since if we can take the output state of the first part of the quantum computer we can then amplify the result. It is also significant since we have shown the amplification will take polynomial time. I will now prove a similar claim that is more general.

Corollary:[3] Let $x_0 = \frac{k}{2^n}$ for k odd, $k << 2^n$ and $n > 0$, $x_i = Q_\mu^{[i]}(x_0)$ with $\mu = 3.71$. Then there exists $m \in \{0, 1, 2, \ldots, 2n\}$ such that $x_m \geq \frac{1}{2}$.

[3]This was alluded to in [1], but not proven. For completeness I have included it here.

Proof: Similar to before we assume that such an m does not exists. Then we have $m \in [0, 2n]$ we have that $x_m < \frac{1}{2}$. So

$$\frac{1}{2} > x_m = \mu x_{m-1}(1 - x_{m-1}) \geq \frac{\mu}{2} x_{m-1}$$

And therefore we have through recursively apply the above

$$\frac{1}{2} > \left(\frac{\mu}{2}\right)^m x_0 = \left(\frac{\mu}{2}\right)^m \frac{k}{2^n}$$

which implies the following relation

$$2^{n+m-1} > \mu^m \cdot k$$

and so,

$$\begin{aligned} \Rightarrow n + m - 1 &> \log(\mu^m \cdot k) \\ \Rightarrow n + m - 1 &> m \log \mu + \log k \\ \Rightarrow n - 1 - \log k &> m \cdot (\log \mu - 1) \\ \Rightarrow \frac{n - 1 - \log k}{\log \mu - 1} &> m \end{aligned}$$

And now we see that $n - 1 - \log k < n - 1$ which immediately yields

$$2n > \frac{n-1}{\log \mu - 1} > \frac{n - 1 - \log k}{\log \mu - 1} > m$$

a contradiction of our assumption for m. Hence such an m exists. ■

Now that we have the polynomial time algorithm for any choice of k, we can try to figure out which value of m would be optimal. We already have an upper bound (2n) so we will show a lower bound.

Claim: [4] Let $x_0 = \frac{k}{2^n}$ for k odd, $0 < k << 2^n$ and $n > 0$, $x_i = Q_\mu^{[i]}(x_0)$ with $\mu = 3.71$. If there exists $m \in [0, 2n]$ for $x_m > \frac{1}{2}$ then $m > \frac{n-1}{\log \mu}$.
Proof: Since $0 \leq x_i \leq 1$ we have that for any $i > 0$

$$x_i = \mu x_{i-1}(1 - x_{i-1}) \leq \mu x_{i-1}$$

Applying the above relation recursively we find

$$x_i \leq \mu^i x_0$$

[4] This is adaptation of the proof in the paper [1], instead of using $\frac{1}{2^n}$ I use $\frac{k}{2^n}$.

Since we know the m exists such that $x_m > \frac{1}{2}$ so

$$x_0 \geq \frac{1}{\mu^m} x_m > \frac{1}{2 \cdot \mu^m}$$

Hence,

$$\begin{aligned} \frac{1}{2 \cdot \mu^m} &< x_0 \\ \Rightarrow \frac{1}{2 \cdot \mu^m} &< \frac{k}{2^n} \\ \Rightarrow 2^n &< 2k\mu^m \\ \Rightarrow n &< m \log \mu + \log 2k \\ \Rightarrow \frac{n - \log 2k}{\log \mu} &< m \end{aligned}$$

And since $\log 2k > 1$ for all k odd, $k > 0$, we have

$$m > \frac{n-1}{\log \mu}$$

as required. ■

Now that we have a range of values for m, we can just choose one at random and use it for solving the SAT problem. Since we will be repeating applications of the algorithm many times (since the probability of successfully distinguishing between states is $> \frac{1}{2}$) it is reasonable to choose a different m for each repetition.

4 Conclusion

At the beginning of the paper we discussed the notion of NPC and the SAT problem. We have shown that given the new model of computation we can solve the SAT problem in polynomial time. The most important change in the new model was the addition of a chaotic amplifier that took amplitudes of qubit states and amplified so that the two possible states were actually measurably different. The amplifier was based off the logistic equation that we studied in class, and I showed that it was sufficient for the purposes of the quantum chaotic Turing machine. In this report I left out a lot of the specific details of the actual system since it would be beyond the scope of the assignment, however it appears that the approach is at least feasible. It is very interesting that normally chaos has been seen as a weakness in models since it can introduce very large errors. However, this time it is not the case; we have strengthened a model with the addition of chaos.

References

[1] M. Ohya, I. Volovich, *New quantum algorithm for studying NP-complete problems*, arXiv:quant-ph/040621

[2] P. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM Journal on Computing 26: pages 1484-1509, 1997

[3] C. Bennett, E. Bernstein, G. Brassard, U. Vazirani, *Strengths and Weaknesses of Quantum Computing*, arXiv:quant-ph/9701001

[4] M. Mains, *Lower Bounds for Quantum Algorithms on Independent Set and Subgraph Isomorphism*, submitted for grading in CS467/CO481: Introduction to Quantum Information Processing, 2005

[5] S. Cook, *The Complexity of Theorem Proving Procedures*, Proceedings Third Annual ACM Symposium on Thoery of Computing, pages 151-158, May 1971

[6] P. Crescenzi, V. Kann, *A compendium of NP optimization problems*, http://www.nada.kth.se/ viggo/problemlist/compendium.html, visited Dec 11, 2005

[7] I. Chuang, M. Nielsen, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, 2000

[8] N. Masuda, *NP problem in Quantum Algorithm*, Open Systems and Information Dynamics 7 No. 1 (2000), pages 33-39t