Quantum Haar Wavelet Transforms and Their Applications

Darwin Gosal[*] and Wayne Lawton[†]
National University of Singapore, Singapore 119260

November 5, 2001

Fourier transform has been shown to be a powerful tool in many area of science. However, there is another class of unitary transforms, the wavelet transforms, which are as useful as the Fourier transform. Wavelet transforms are used to expose the multi-scale structure of a signal and very useful for image processing and data compression. In this paper, we construct quantum algorithms for Haar wavelet transforms and show its application in analyzing the multi-scale structure of the dynamical system by the Logistic Map ($x \rightarrow \lambda x(1-x)$), where λ takes value in the interval $[0, 4)$.

1 Introduction

Information is stored, transmitted and processed by physical means [1]. Thus, the concept of information and computation can be formulated in the context of a physical theory and the study of information requires experimentation. This sentence leads to non-trivial consequences in the world of quantum mechanics.

The field of quantum information science has undergone explosive activity over the past few years. This quantum information science has generated three great developments in the latter part of the 20th century, they are quantum computation, quantum cryptography, and quantum error correction. These three developments in quantum information science all pose great challenges for both 21st century science and technology.

Quantum Computation not only fascinates physicists, it also interests computer scientists and mathematicians. For computer scientist, quantum

[*]Email: `gosaldar@sps.nus.edu.sg`
[†]Email: `wlawton@math.nus.edu.sg`

[1]As what Rolf Landauer said "Information is physical"

computation introduces new complexity classes that can transform some NP-Complete problems to the Polynomial complexity.

Several quantum algorithms are known, the most famous example is Deutsch's algorithm for deciding whether a function is even or balanced. One of the most remarkable quantum algorithms is Shor's algorithm. This algorithm basically depends on quantum Fourier transform (QFT). As we know the Fourier transform is good for analyzing the periodicity of a function, but wavelet transform is able to analyze the multiscale structure of the function.

It is sufficient to use some basic empirically-based principles of quantum behavior in order to explain the basic principles of Quantum Computing and to develop quantum algorithms. In order to formulate these principles we need to introduce some basic concepts.

2 Quantum Mechanics

Quantum theory is a mathematical model of the physical world. To characterize the model, we need to specify how it represents: states, observables, measurement, and dynamics.

2.1 Axioms of quantum mechanics

Quantum mechanics is a mathematical framework for the development of physical theories. On its own quantum mechanics doesn't tell you what laws a physical system must obey, but it does provide a mathematical and conceptual framework for the development of such laws. Therefore, we need some axioms to provide a connection between the physical world and the mathematical formalism of quantum mechanics.

States Associated to any isolated physical system is a complex vector space with inner product (that is, a Hilbert space[2]) known as the *state space* of the system. The system is completely described by its *state vector*, which is a unit vector in the system's state space. In short, a state vector (which is a ray in a Hilbert space) can be thought of as the mathematical representative of the physical notion of 'state' of the system.

Observables An observable is a property of a physical system that in principle can be measured.The observables of the system can be represented

[2]which I will elaborate more on the next subsection

mathematically by self-adjoint operator that act on the Hilbert space $\mathcal{H}$. An operator is a linear map taking vectors to vectors

$$\mathbf{A} : |\psi\rangle \to \mathbf{A}|\psi\rangle, \mathbf{A}\left(a|\psi\rangle + b|\psi\rangle\right) = a\mathbf{A}|\psi\rangle + b\mathbf{A}|\psi\rangle \tag{1}$$

$\mathbf{A}$ is self-adjoint[3] if and only if $\mathbf{A} = \mathbf{A}^\dagger$.

Measurement Quantum measurements are described by a collection $\{\mathbf{M}_m\}$ of *measurement operators*. These are operators acting on the state space of the system being measured. The index m refers to the measurement outcomes that may occur in the experiment. If the state of the quantum system is $|\psi\rangle$ immediately before the measurement then the probability that result m occurs is given by

$$P(m) = \langle\psi|\mathbf{M}_m^\dagger \mathbf{M}_m|\psi\rangle \tag{2}$$

and the state of the system after measurement is

$$\frac{\mathbf{M}_m|\psi\rangle}{\sqrt{\langle\psi|\mathbf{M}_m^\dagger \mathbf{M}_m|\psi\rangle}}. \tag{3}$$

The measurement operators satisfy the completeness equation,

$$\sum_m \mathbf{M}_m^\dagger \mathbf{M}_m = \mathbf{I} \tag{4}$$

The completeness equation expresses the fact that probabilities sum to one.

Dynamics The evolution of a *closed* quantum system is described by a *unitary transformation*. That is, the state $|\psi\rangle$ of the system at time t_1 is related to the state $|\psi'\rangle$ of the system at time t_2 by a unitary operator $\mathbf{U}$ which depends only on the times t_1 and t_2,

$$|\psi'\rangle = \mathbf{U}|\psi\rangle. \tag{5}$$

The time evolution of the state of a closed quantum system is described by the *Schrödinger equation*,

[3]or Hermitian

$$i\hbar \frac{d|\psi\rangle}{dt} = \mathbf{H}|\psi\rangle. \quad (6)$$

where $\mathbf{H}$ is a fixed Hermitian operator knowns as the *Hamiltonian* of the closed system. In the case where $\mathbf{H}$ is t-independent; it can be shown that $\mathbf{U} = e^{-it\mathbf{H}}$.

This completes the mathematical formulation of quantum mechanics. Further reading on the foundation of quantum mechanics can be found on [8]. A very useful bibliographic guide to the foundations of quantum mechanics and quantum information can be found on [11].

2.2 Hilbert space

In this subsection I will define Hilbert spaces $\mathcal{H}$ that furnishes the mathematical basis for the treatment of quantum mechanics in terms of those concepts which are subsequently needed in quantum mechanics.

Hilbert space is a vector space over complex numbers $\mathbb{C}$. In the following I shall denote the points of Hilbert space by $|\phi\rangle$, $|\chi\rangle$, $|\psi\rangle$, $\ldots$, complex numbers by $a, b, c, \ldots$, and positive integers by $i, j, k, \ldots$.

These are the five Hilbert spaces axioms:

First axiom A linear vector space $\mathcal{S}$, with a carrier H, over a field $\mathcal{K}$ with the carrier K is an algebra $\mathcal{S} = \langle H, +, ^{-1}, \mathbf{0}, K, +_f, \times_f, 0, 1, \cdot\rangle$ such that $\langle H, +, ^{-1}, \mathbf{0}\rangle$ is a commutative group, $\mathcal{K} = \langle \mathcal{K}, +_f, \times_f, 0, 1\rangle$ is a field, and $\cdot : \mathcal{K} \times H \to H$ is a scalar multiplication satisfying the following axioms for any $a, b \in K$, $|\phi\rangle$, $|\psi\rangle$, and $|\chi\rangle \in H$:

- Commutative law of addition

$$|\psi\rangle + |\phi\rangle = |\phi\rangle + |\psi\rangle$$

- Associative law of addition

$$(|\phi\rangle + |\psi\rangle) + |\chi\rangle = |\phi\rangle + (|\psi\rangle + |\chi\rangle)$$

- Distributive law of multiplication

$$(a +_f b)|\psi\rangle = a \cdot |\psi\rangle + b \cdot |\psi\rangle$$
$$a \cdot (|\phi\rangle + |\psi\rangle) = a \cdot |\phi\rangle + a \cdot |\psi\rangle$$

- Associative law of multiplication

$$(a \times_f b) \cdot |\psi\rangle = a \cdot (b \cdot |\psi\rangle)$$

- Role of 0 and 1

$$0 \cdot |\psi\rangle = \mathbf{0} \quad ; \quad 1 \cdot |\psi\rangle = |\psi\rangle$$

Second axiom A complex inner-product space $\mathcal{H}$ is a vector space with a carrier H over the field of complex numbers, equipped with an inner product (also called scalar product or Hermitian scalar product) $\langle\cdot|\cdot\rangle : H \times H \to \mathbb{C}$ satisfying, for any $|\phi\rangle$, $|\psi\rangle$, and $|\chi\rangle \in H$, and any $c_1, c_2 \in \mathbb{C}$, the following properties:

- Hermitian symmetry

$$\langle\psi|\phi\rangle = \langle\phi|\psi\rangle^*$$

- Definite form

$$\langle\psi|\psi\rangle \geq 0 \text{ and } \langle\psi|\psi\rangle = 0 \text{ if and only if } |\psi\rangle = \mathbf{0}$$

- Associative and distributive law

$$\langle\psi|c_1\phi + c_2\chi\rangle = c_1\langle\psi|\phi\rangle + c_2\langle\psi|\chi\rangle$$

The inner product introduces on H the norm $\|\psi\|_H = \sqrt{\langle\psi|\psi\rangle}$ and the distance between ψ and ϕ is $\|\psi - \phi\|$.

Third axiom There are arbitrarily many linearly independent vectors. that is, for each $k = 1, 2, \ldots$, we can specify k such vectors.

Fourth axiom An inner-product space $\mathcal{H}$ is *complete.* That is, if for any sequence $\{|\psi_i\rangle\}_{i=1}^{\infty}$ in $\mathcal{H}$ satisfies the Cauchy convergence criterion (for each $\epsilon > 0$, there exist an $N = N(\epsilon)$, such that $\|\psi_m - \psi_n\| < \epsilon$ for all $m, n \geq N$), then it is convergent. A complete inner-product space is called a Hilbert space.

Fifth axiom $\mathcal{H}$ is separable. That is, there is a sequence $|\psi_1\rangle, |\psi_2\rangle, \ldots$ in $\mathcal{H}$ which is everywhere dense in $\mathcal{H}$.

Note:

To each continuous linear mapping $f : \mathcal{H} \to \mathbb{C}$, the Riesz representation theorem ensure that there exists a unique $\phi_f \in \mathcal{H}$ such that $f(\psi) = \langle \phi_f | \psi \rangle$ for any $\psi \in \mathcal{H}$. The space of all linear mapping (called also functionals) of a Hilbert space $\mathcal{H}$ forms again a Hilbert space, called *dual Hilbert space* (or *conjugate Hilbert space*). The mapping $f_\phi(\psi) = \langle \phi | \psi \rangle$ is a functional for any $\phi \in \mathcal{H}$. Thus a bra-vector $\langle \cdot |$ can be seen as the operator that maps each state ϕ into a functional $\langle \phi |$ such that $\langle \phi | (|\psi\rangle) = \langle \phi | \psi \rangle$ for every state.

A rigorous mathematical treatment for Hilbert space can be found on [9]

2.3 The density matrix

The axioms of quantum mechanics that discussed in the subsection 2.1 provide a perfectly acceptable general formulation of the quantum theory. The trouble is that our axioms are intended to characterize the quantum behavior of the entire universe. In practice, the observations are always limited to a small part of a much larger quantum system. When we limit our attention to just part of a larger system, then:

1. States are *not* rays.
2. Measurements are *not* orthogonal projections.
3. Evolution is *not* unitary.

We can best understand these points by considering the bipartite system or a system that undergoes decoherence. Therefore, states that are not pure are called mixed states.

Consider a mixed state $\rho := (|\psi_1\rangle, |\psi_2\rangle, \ldots, |\psi_n\rangle; p_1, p_2, \ldots, p_n)$ where $\sum_{i=1}^{n} p_i = 1$. Let $\mathbf{A}$ be any operator. Now let $|\psi\rangle$ in $\mathcal{H}$. Then

$$\begin{aligned} \langle A \rangle_\psi = \langle \psi | \mathbf{A} | \psi \rangle &= \sum_{j=1}^{N} \langle \psi | \mathbf{A} | e_j \rangle \langle e_j | \psi \rangle \equiv \sum_{j=1}^{N} \langle e_j | \psi \rangle \langle \psi | \mathbf{A} | e_j \rangle \\ &= \sum_{j=1}^{N} \langle e_j | \boldsymbol{\rho_\psi} \mathbf{A} | e_j \rangle = Tr(\boldsymbol{\rho_\psi} \mathbf{A}) \end{aligned} \tag{7}$$

where $\boldsymbol{\rho_\psi} := |\psi\rangle\langle\psi|$

Now define the *mixed-state operator* $\boldsymbol{\rho}$ by

$$\boldsymbol{\rho} \equiv \sum_i p_i |\psi_i\rangle\langle\psi_i| \tag{8}$$

The operator $\boldsymbol{\rho}$ has the following properties:

1. $\boldsymbol{\rho}$ is self-adjoint: $\boldsymbol{\rho} = \boldsymbol{\rho}^\dagger$
2. $\boldsymbol{\rho}$ is a positive, semi-definite operator: $\langle\psi|\boldsymbol{\rho}|\psi\rangle \geq 0$ for all $|\psi\rangle$
3. $Tr(\boldsymbol{\rho}) = 1$

It follows that $\boldsymbol{\rho}$ can be diagonalized, that the eigenvalues are all real and non-negative, and that the eigenvalues sum up to one. In general, any operator satisfying these three conditions is called a *density matrix.* Note that a pure density matrix has the property $\boldsymbol{\rho}^2 = \boldsymbol{\rho}$. Hence, the density matrix $\boldsymbol{\rho}$ is a "generalization" of the notion of state and it represent *all and only* information which can be learned by sampling the ensemble.

3 Quantum Computation and Information

Quantum computation and quantum information is the study of the information processing tasks that can be accomplished using quantum mechanical systems that we described earlier. Two key problems here are: how to represent information and how to manipulate quantum information.

3.1 Qubits and entanglement

The most fundamental concept of classical computation and classical information is *bit* (binary digit). This is a system that can take on one of two values, such as true and false or 0 and 1. *Qubit* (Quantum bit) is the quantum analog of a bit. Just as a classical bit, it has two states.

Let $\mathcal{H}$ be a two-dimensional quantum system with two orthonormal states, denoted by $|0\rangle$ and $|1\rangle$, that can be considered as forming standard basis of $\mathcal{H}$. A qubit is a quantum state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{9}$$

where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$.

One picture useful in thinking about qubits is the following representation.

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle \tag{10}$$

where θ takes value from 0 to π and φ takes value from 0 to 2π. The number θ and φ define a point on the three-dimensional sphere. One can observe that when $\theta = \pi$ or $\theta = 0$, all values of φ yield the same state. This sphere is often called the Bloch sphere or Poincaré sphere. Mathematically, this can

$|0\rangle$

$|1\rangle$

Figure 1: Bloch sphere representation of a qubit.

be seen as $\mathbb{C}^2/\mathbb{C}^* \cong S^3/S \cong S^2$, where $\mathbb{C}$ is a complex vector space and $\mathbb{S}$ is a spherical group defined $S^{n-1} = \{x \in \mathbb{R}^n | \; \|x\|^2 = x \cdot x = 1\}$.

The most essential property of quantum states when used to encode bits is the possibility of *coherence* and *superposition.* This means is not that the value of a qubit is somewhere between 0 and 1, but rather that the qubit is in a superposition of both states. This phenomena was first shown by Thomas Young in his double-slit experiment.

In order to implement any useful quantum algorithm we need to deal with many qubits in one Hilbert Space, and the appropriate model is a *tensor product* of qubits. Specifically, if we have n qubits, each with a given computational basis in a two-dimensional Hilbert Space $\mathcal{H}$, then the tensor product is a 2^n-dimensional space with a basis consisting of 2^n vectors.

Let $\mathcal{B}$ be a set of standard basis vectors:

$$\mathcal{B} = \{|i\rangle \mid i \in \{0,1\}^n\} \tag{11}$$

or, another notation,

$$\mathcal{B} = \{|i\rangle \mid 0 \leq i < 2^n\} \tag{12}$$

The general qubit state of the n-qubits register is

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle \text{ with } \sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1 \tag{13}$$

n-qubits system might enable us to simultaneously represent all 2^n numbers, and calculate the value of function at all 2^n integers simultaneously. Thus, the basic strategy of quantum algorithm is to take advantage of superpositions, which grows exponentially with the number of qubits.

An n-qubit state that is a *direct product* of the pure states in sub-systems,

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes \ldots \otimes |\psi_n\rangle = \left(\sum_{j=0}^{1} \beta_{1_j} |j\rangle\right) \otimes \left(\sum_{j=0}^{1} \beta_{2_j} |j\rangle\right) \otimes \ldots \otimes \left(\sum_{j=0}^{1} \beta_{n_j} |j\rangle\right) \tag{14}$$

is said to be *separable* (or unentangled). Thus, a state that cannot be decompose into the tensor product of one-qubit states is said to be entangled state.

An important example of entangled states is the *Bell state* or *EPR pair*, i.e.

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{15}$$

This innocuous-looking state is responsible for a remarkable phenomena called non-locality and is at the heart of EPR (Einstein-Podolsky-Rosen) paradox. The set of Bell states ($\Psi^{\pm}$ and $\Phi^{\pm}$) is the key ingredient in quantum teleportation and super-dense coding. A further reading on entanglement can be found in [10], [5], and [3]

3.2 Quantum Gates

Classical computer circuits consist of wires and logic gates. In a similar way, quantum computer have a quantum gates from which quantum computing devices are designed.

Quantum gates on a n-qubit can be described by a 2^n by 2^n matrices. Because of the normalization condition requires $\sum_{i=0}^{2^n-1}|\alpha_i|^2 = 1$, there is a constrain for matrices that can be used as quantum gates. It turns out that the appropriate condition is that the matrix $\mathbf{U}$ describing the quantum gate must be *unitary*, that is $\mathbf{U}^\dagger\mathbf{U} = \mathbf{I}$. The implication of this, is that a unitary operation implies a reversible operation.

A *quantum gate* is specified by a unitary operator $\mathbf{U} : \mathcal{H}_{2^n} \to \mathcal{H}_{2^n}$.

Below are the frequently used quantum gates.

Hadamard	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Pauli X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli Y	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Phase	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
$\frac{\pi}{8}$	$\begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix}$

Controlled-NOT	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Swap	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Controlled-Z	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$
Controlled-phase	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathrm{i} \end{bmatrix}$
Toffoli	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$
Fredkin (controlled-swap)	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

3.3 Quantum Circuits

Quantum circuit is a collection of quantum gates acyclicly[4] connected by "quantum wires"[5]. There are few features allowed in classical circuits that are not present in quantum circuits. First of all, classical circuits allow wires to be 'joined' together, an operation known as FANIN. Secondly, the inverse operation, FANOUT whereby several copies of a bit is produced, is also not allowed in quantum circuits.

The size and the depth of a circuit refer to the number of nodes and depth of the underlying connection graph. The circuit is to be read from the left-to-right. It is conventional to assume that the state input to the circuit is a computational basis state, usually the state consisting of all $|0\rangle$'s. We shall find quantum circuits useful as models of all quantum processes, including but not limited to computation, communication, and even quantum noise.

An interferometer experiment can be cast into a quantum network as follow:

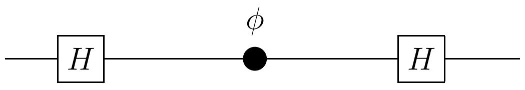


Figure 2: Interferometer circuit

The gate in the middle is the phase shift gate ϕ defined as $|0\rangle \mapsto |0\rangle$ and $|1\rangle \mapsto e^{i\phi}|1\rangle$, or in matrix notation,

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix} \tag{16}$$

It turns out that this simple circuit is the implementation of Deutsch-Jozsa algorithm[6]. A natural extension of this gate, is the controlled-U gate (U is any unitary matrix). It has been shown[7] that the Hadamard gate, phase gate, and the C-NOT, form an infinite universal set of gates[8].

It is possible for a quantum circuit to simulate a classical logic circuit. A simple concatenation of the Toffoli gate and the C-NOT gives a simplified

[4]which means, we don't allow 'loops', that is, feedback from one part of the quantum circuit to another

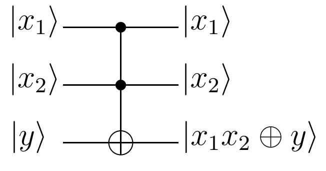
[5]This wire does not necessarily correspond to a physical wire; it may correspond instead to the passage of time, or perhaps by sharing a physical qubit (physical particle), or via field interactions.

[6]which I will discuss in the next subsection

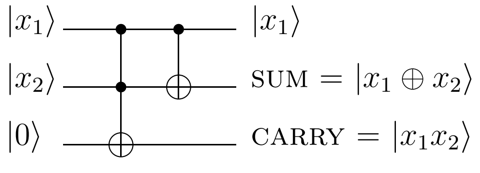
[7]Solovay-Kitaev theorem

[8]*Phys. Rev. A* 52 3457

quantum adder, which is a good starting point for constructing full adders, multipliers and more elaborate arithmetic circuits.



Toffoli gate



Quantum adder

Another important quantum circuit is the Quantum Fourier Transformation (QFT) circuit. The Fourier transform

$$\sum_x f(x)|x\rangle \to \sum_y \left(\frac{1}{\sqrt{N}} \sum_x e^{2\pi i x y/2^N} f(x) \right) |y\rangle \tag{17}$$

is multiplication by an $N \times N$ unitary matrix. With quantum parallelism, we can write a Quantum Fourier Transform defined in the computational basis as the unitary operator

$$|x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_y e^{2\pi i x y/2^N} |y\rangle \tag{18}$$

A given phase $\phi_x = 2\pi x/2^N$ can be encoded by a QFT. In this process the information about ϕ_x is distributed between states of a register. An important observation is that the QFT takes each computational basis state to an unentangled state.

An important observation is that the QFT of x, is unentangled, and in fact can be factorized as

$$(|0\rangle + e^{i\phi_x}|1\rangle)(|0\rangle + e^{i2\phi_x}|1\rangle)\dots(|0\rangle + e^{i2^{n-1}\phi_x}|1\rangle) \tag{19}$$

The QFT can be implemented with Hadamard gates and the controlled phase shift in between (see figure 3).

The QFT network operating on n qubits contains n Hadamard gates H and $n(n-1)/2$ phase shifts. This remarkably simple circuit is the heart of the ground-breaking Shor's algorithm.

3.4 Quantum Algorithms and their complexity

Suppose we are challenged to distinguish between two different classes of functions f which map $\{0,1\} \to \{0,1\}$. We have exactly four such functions.

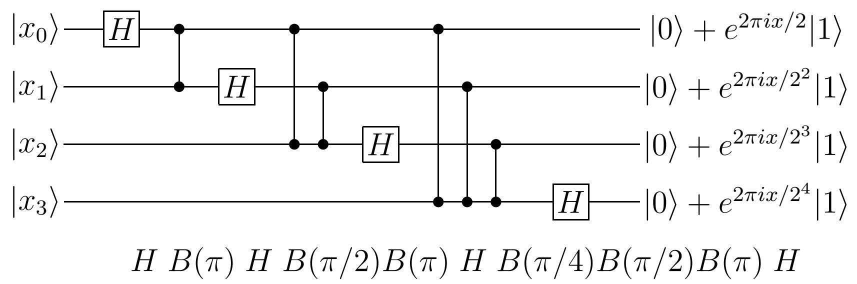

$|x_0\rangle$
$|x_1\rangle$
$|x_2\rangle$
$|x_3\rangle$
H
$|0\rangle + e^{2\pi i x/2}|1\rangle$
$|0\rangle + e^{2\pi i x/2^2}|1\rangle$
$|0\rangle + e^{2\pi i x/2^3}|1\rangle$
$|0\rangle + e^{2\pi i x/2^4}|1\rangle$
$H\ B(\pi)\ H\ B(\pi/2)B(\pi)\ H\ B(\pi/4)B(\pi/2)B(\pi)\ H$

Figure 3: Quantum Fourier Transformation circuit

Two constant function:

$$\begin{array}{ccc} f(0) = 0 & & f(0) = 1 \\ & or & \\ f(1) = 0 & & f(1) = 1 \end{array} \tag{20}$$

and two balances functions:

$$\begin{array}{ccc} f(0) = 0 & & f(0) = 1 \\ & or & \\ f(1) = 1 & & f(1) = 0 \end{array} \tag{21}$$

Suppose that we have a "black box" (also called "oracle") that computes a Boolean function. The task is to determine whether the function by the oracle is balanced or constant, or equivalently, to determine whether $f(0) \oplus f(1) = 0$ or 1.

In classical computation, it clear that we need to query the oracle twice to solve the problem. We shall see that we can solve this problem with a single query, by employing an algorithm that has the same mathematical structure as the interferometer.

Let us take two qubits, prepare the first qubit in the state $|0\rangle$ and the second in state $|1\rangle$. Let us follow the states along the circuit.

The input state

$$|\psi_0\rangle = |01\rangle \tag{22}$$

is sent through two Hadamard gates and give

$$|\psi_1\rangle = \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right] \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \tag{23}$$

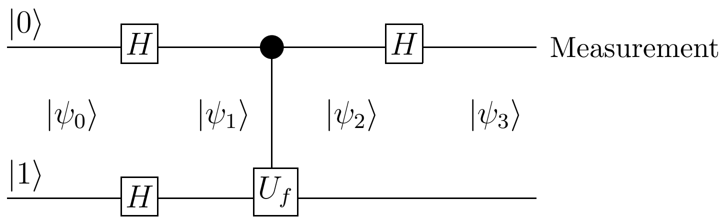


Figure 4: Quantum network solving Deutsch's Problem

Let U_f be the unitary mapping of $|x, y\rangle$ into $|x, y \oplus f(x)\rangle$. Thus, by applying U_f to $|\psi_1\rangle$ we have

$$\begin{aligned} U_f : \quad & \left(\tfrac{1}{\sqrt{2}}\left[|0\rangle + |1\rangle\right]\right)\left(\tfrac{1}{\sqrt{2}}\left[|0\rangle - |1\rangle\right]\right) \rightarrow \\ & \left(\tfrac{1}{\sqrt{2}}\left[(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right]\right)\left(\tfrac{1}{\sqrt{2}}\left[|0\rangle - |1\rangle\right]\right) = |\psi_2\rangle \end{aligned} \tag{24}$$

The final Hadamard gate on the first qubit gives us

$$|\psi_3\rangle = |f(0) \oplus f(1)\rangle \left(\frac{1}{\sqrt{2}}\left[|0\rangle - |1\rangle\right]\right) \tag{25}$$

so by measuring the first qubit, we may determine $f(0) \oplus f(1)$. The quantum circuit has given us ability to extract *global information* of the function, namely $f(0) \oplus f(1)$ using only one evaluation of $f(x)$. This example highlights the difference between quantum parallelism and classical algorithm.

This Deutsch's algorithm was subsequently generalized to cover "black boxes" computing Boolean functions $f : \{0, 1\}^n \mapsto \{0, 1\}$ (this algorithm is called Deutsch-Jozsa algorithm). This quantum computation exhibits "*massive* quantum parallelism" There are many algorithms follow the same pattern as Deutsch's algorithm: the Hadamard transform, a unitary function evaluation, the Hadamard transform (H-f-H sequence). We recognize it as a generic interference pattern.

There are three classes of quantum algorithms which provide an advantage over known classical algorithms. The first class is algorithms based upon quantum Fourier transformation. The Deutsch-Jozsa algorithm and Shor's algorithm are example of this type of algorithm. The second class of algorithms is quantum search algorithms. The third class is quantum simulation, whereby a quantum computer is used to simulate a quantum system (just like Feynman's original proposal of the usage of quantum computer). We now briefly describe Shor's algorithm.

Shor's algorithm evolved from the order finding problem, which was one of the application of optimal phase estimation algorithm. States of the form (19) are form by function evaluation in quantum computers. Suppose that U is any unitary transformation on n qubits and $|\psi\rangle$ is an eigenvector of U with eigenvalue $e^{i\phi}$. We do not explicitly know U or $|\psi\rangle$ or $e^{i\phi}$, but instead we are given devices that perform $C-U$, $C-U^2$, until $C-U^{2^{n-1}}$. Our goal is to obtain an n-bit estimator of ϕ. We start by constructing the following network,

The phase estimation algorithm works as follow:

Inputs: A black box which performs a controlled-U^j operation, for integer j, and eigenstate $|\psi\rangle$ of U with eigenvalue $e^{i\phi_\psi}$, and $t = n + \lceil \log(2+\frac{1}{2\epsilon}) \rceil$ qubits initialized to $|0\rangle$.

Outputs: An n-bit approximation $\widetilde{\phi_\psi}$ to ϕ_ψ.

Runtime: $O(t^2)$ operations. Succeeds probability at least $1-\epsilon$.

Procedure: 1. $|0\rangle|\psi\rangle$ initial state

2. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|\psi\rangle$ create superposition

3. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle U^j|\psi\rangle$ apply black box
$= \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} e^{ij\phi_\psi}|j\rangle|\psi\rangle$ result of black box

4. $\rightarrow |\widetilde{\phi_\psi}\rangle|\psi\rangle$ apply inverse Fourier transform

5. $\rightarrow \phi_\psi$ measure first register

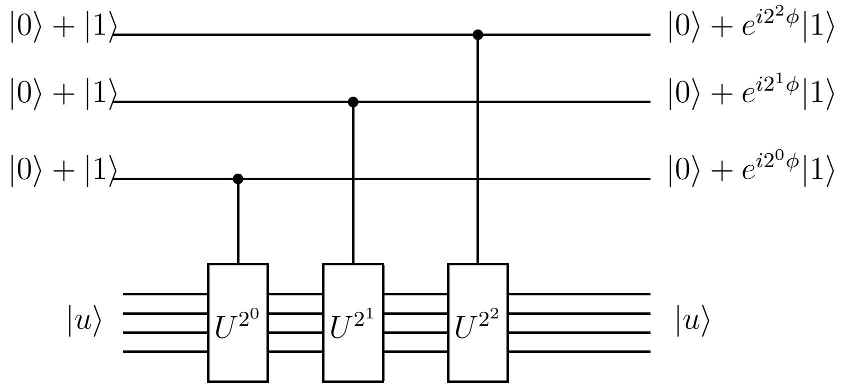


Figure 5: Phase estimation circuit

This algorithm can be applied to make quantum order-finding algorithm, which is the heart of Shor's factoring algorithm.

The quantum order-finding algorithm work as follow:

Inputs: A black-box $U_{x,N}$ which performs the transformation $|j\rangle|k\rangle \rightarrow |j\rangle|x^j k \bmod N\rangle$, for x co-prime to L-bit numbers N, $t = 2L + 1 + \lceil \log(2 + \frac{1}{2\epsilon}) \rceil$ qubits initialized to $|0\rangle$, and L qubits initialized to the state $|1\rangle$.

Outputs: The least integer $r > 0$ such that $x^r = 1(\bmod N)$.

Runtime: $O(L^3)$ operations. Succeeds with probability $O(1)$.

Procedure:

1. $|0\rangle|1\rangle$ initial state
2. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|1\rangle$ create superposition
3. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|x^j \bmod N\rangle$ apply $U_{x,N}$
   $\approx \frac{1}{\sqrt{r2^t}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} e^{isj/r} |j\rangle|\psi_s\rangle$
4. $\rightarrow \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |\widetilde{s/r}\rangle|\psi_s\rangle$ apply inverse Fourier transform to first register
5. $\rightarrow \widetilde{s/r}$ measure first register
6. $\rightarrow r$ apply continued fractions algorithm

It has been shown that a factoring problem can be reduced to order-finding problem. The Shor's algorithm can be summarized as follow:

Inputs: A composite number N

Outputs: A non-trivial factor of N

Runtime: $O(\log^3 N)$ operations. Succeeds with probability $O(1)$.

Procedure:

1. If n is even, return the factor 2.
2. Determine whether $N = a^b$ for integers $a \geq 1$ and $b \geq 2$, and if so return the factor a
3. Randomly choose x in the range 1 to $N - 1$. If $\gcd(x, N) > 1$ then return the factor $\gcd(x, N)$.
4. Use order-finding subroutine to find the order r of x modulo N.

5. If r is even and $x^{r/2} \neq -1 \pmod N$ then compute $\gcd(x^{r/2}-1, N)$ and $\gcd(x^{r/2}+1, N)$, and test to see if one of these is a non-trivial factor, returning that factor if so. Otherwise, the algorithm fails.

More of quantum algorithms can be found in [4], [13], [15], and [6]. Web resources on Quantum information can be found in: http://www.qubit.org and http://www.physics.nus.edu.sg/ phyohch/hyperqc2.htm.

4 Wavelet

4.1 General characteristic of wavelets

Data as a bit (binary digit) is just a mathematical representation for the computation. But in our daily life, what we see is continuous wave. Therefore we need to make this into a digital signal and analyse it. A wave is usually defined as an oscillating function of time or space, such as a sinusoid.

There are three ways to transform (or analyse) this signals:

1. by Fourier transform. This method expands signals in terms of cosine waves, which has proven to be extremely important for periodic, time-invariant, or stationary phenomena. The practicality problem of this method is that, it need many frequencies for a high-fidelity signal.

2. into short time Fourier transform. Short segments of the signals are transformed separately. In each segment, the signal is expanded into cosine wave as before. The disadvantage of this method is there are sudden breaks between segments ("blocking effect").

3. into wavelets. A wavelet is a "small wave" that start and stop. It has energy concentrated in time, which useful to analyze transient, non-stationary, or time-varying phenomena. All wavelets come from one basic wavelet $\psi(t)$.

A signal $f(t)$ can be expressed as a linear decomposition by

$$f(t) = \sum_l a_l \psi_l(t) \tag{26}$$

The series representation of f in (26) is called a *wavelet series.* If the expansion (26) is unique, the set is called a *basis* for the class of functions. If the basis is orthogonal, meaning

$$\langle \psi_k(t) | \psi_l(t) \rangle = \int \psi_k(t) \psi_l(t) = 0 \qquad k \neq l \tag{27}$$

then the coefficients can be calculated by the *inner product*

$$a_k = \langle f(t) | \psi_k(t) \rangle = \int f(t) \psi_k(t) dt. \tag{28}$$

For *wavelet expansion*, a two-parameter system is constructed such that (26) becomes

$$f(t) = \sum_k \sum_j a_{j,k} \psi_{j,k}(t) \tag{29}$$

The $\psi_{j,k}(t)$ are the wavelet expansion functions that usually form an orthogonal basis. The set of expansion coefficients $a_{j,k}$ are called the *discrete wavelet transform* (DWT) of $f(t)$.

The wavelet expansion set is not unique, but all seem to have the following general characteristics:

1. A wavelet system is a set of *building block* to construct or represent a signal or function. It is a two-dimensional expansion set for some class of one-(or higher) dimensional signals.

2. The wavelet expansion gives a time-frequency *localization* of the signal. This means most of the energy of the signal of the signal is well represented by a few expansion coefficients.

3. The calculation of the coefficients from the signal can be done *efficiently*. It turns out that many wavelet transforms can be calculated with $O(N)$ operations. More general wavelet transforms require $O(N \log(N))$ operations, the same as for the FFT.

4. All wavelet systems are generated from a single scaling function or wavelet by simple *binary scaling*(i.e. dilation by 2^j) and *dyadic translation* (of $k/2^j$). The two-dimensional parameterization is achieved from the generating wavelet $\psi(t)$ by

$$\psi_{j,k}(t) = 2^{j/2} \psi(2^j t - k) \qquad j, k \in \mathbb{Z} \tag{30}$$

5. Almost all useful wavelet systems also satisfy the *multi-resolution* conditions. This means that if a set of signals can be represented by a weighted sum of $(t - k)$, then a larger set (including the original) can be represented by a weighted sum of $\varphi(2t - k)$.

6. The lower resolution coefficients can be calculated from higher resolution coefficients by a tree-structured algorithm called a *filter bank*. Filter bank is a set of linear time-invariant operator.

Before going any further, we will see the connection between filter banks and wavelets, and you will see that the high-pass filter leads to $\psi(t)$ and the low-pass filter leads to scaling function $\varphi(t)$.

4.2 Filter banks

Filter bank is a set of filters. The analysis bank often has two filters, low-pass and high-pass. They separate input signal into frequency bands. Those sub-signals can be compressed much more efficiently than the original signal. A filter is a linear time-invariant operator that acts on input vector $\mathbf{x}$ and gives an output vector $\mathbf{y}$ which is the convolution of $\mathbf{x}$ with a fixed vector $\mathbf{h}$. The vector $\mathbf{h}$ contains the filter coefficients.

$$\mathbf{y}(n) = \sum_k \mathbf{h}(k)\mathbf{x}(n-k) = \mathbf{h} * \mathbf{x} \quad \text{(convolution in the time domain)} \tag{31}$$

4.2.1 Low-pass Filter / Moving Average

We go forward by introducing the simplest low-pass filter. Low-pass filter has its output at time $t = n$ as the average of the input $\mathbf{x}(n)$ and the input $\mathbf{x}(n-1)$:

$$\mathbf{y}(n) = \frac{1}{2}\mathbf{x}(n) + \tfrac{1}{2}\mathbf{x}(n-1) \tag{32}$$

The filter coefficients are $\mathbf{h}(0) = \frac{1}{2}$ and $\mathbf{h}(1) = \frac{1}{2}$. It is a moving average, because the output averages the current component with the previous one.

4.2.2 High-pass Filter / Moving Difference

High-pass filter has its output at time $t = n$ as the difference of the input $\mathbf{x}(n)$ and the input $\mathbf{x}(n-1)$:

$$\mathbf{y}(n) = \tfrac{1}{2}\mathbf{x}(n) - \tfrac{1}{2}\mathbf{x}(n-1) \tag{33}$$

The filter coefficients are $\mathbf{h}(0) = \frac{1}{2}$ and $\mathbf{h}(1) = -\frac{1}{2}$. It is a moving difference.

4.3 Scaling function and wavelets

Corresponding to the low-pass filter, there is a continuous-time scaling function $\varphi(t)$. The dilation equation for the scaling function $\varphi(t)$ is

$$\varphi(t) = 2\sum_{k=0}^{N} \mathbf{h}(k)\varphi(2t-k) \tag{34}$$

Corresponding to the high-pass filter, there is a continuous-time scaling function wavelet $\psi(t)$. It is a direct equation that gives $\psi(t)$ immediately and explicitly from $\varphi(t)$:

$$\psi(t) = 2 \sum \mathbf{h}_1(k)\phi(2t-k) \tag{35}$$

4.4 Haar wavelet

In our example, $\varphi(t)$ is a box function and its dilations $\varphi(2t-k)$ are half boxes, then the wavelet is:

$$\psi(t) = 2 \sum \mathbf{h}_1(k)\phi(2t-k) \tag{36}$$

Explicitly, $\psi(t) = 1$ for $0 \leq t <\frac{1}{2}$ and $\psi(t) = -1$ for $\frac{1}{2} \leq t < 1$. This is the *Haar wavelet*.

Length 2 Haar wavelet can be represented as:

$$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{37}$$

A sequence with length $\gg 2$, can be calculated faster with fast wavelet transform (FWT). Fast wavelet transform is a tree-structured filter bank. The algorithm of FWT can be found on the appendix[9].

A more rigorous mathematical approach can be found on [16], [17], [18], and [19]. More resource on wavelet can be found on the web: http://www.amara.com/current/wavelet.html.

5 Quantum Haar Wavelet

Transition for "classical" wavelet transform to quantum wavelet transform can be approached by factoring the classical operators for the transformation into direct sums, direct products, and dot products of unitary matrices. In doing so, we will find that permutation matrices play a vital role.

Two fundamental permutation matrices for quantum haar transform are the perfect shuffle Π_{2^n} and the bit reversal P_{2^n}.

Description of the matrix Π_{2^n} in terms of its element Π_{ij} can be given as

$$\Pi_{ij} = \begin{cases} 1 & \text{if } j = 1/2 \text{ and } i \text{ is even, or if } j = (i-1)/2 + 2^{n-1} \text{ and } i \text{ is odd} \\ 0 & \text{otherwise} \end{cases} \tag{38}$$

[9] fwt.m

As noted by Hoyer, a quantum description of Π_{2^n} can be given by

$$\Pi_{2^n} : |a_{n-1}a_{n-2}\cdots a_1a_0\rangle \mapsto |a_0a_{n-1}a_{n-2}\cdots a_1\rangle \tag{39}$$

Here we can see that a swap gate Π_4 is a special case of permutation matrix.

Description of the matrix P_{2^n} in terms of its element P_{ij} can be given as

$$\Pi_{ij} = \begin{cases} 1 & \text{if } j \text{ is bit reversal of } i \\ 0 & \text{otherwise} \end{cases} \tag{40}$$

A quantum description of P_{2^n} can be given by

$$P_{2^n} : |a_{n-1}a_{n-2}\cdots a_1a_0\rangle \mapsto |a_0a_1\cdots a_{n-2}a_{n-1}\rangle \tag{41}$$

P_{2^n} can be factorized in terms of Π_{2^i} and Π_{2^i} can be factorized in term of Π_4. From previous section we know that the swap gate can be made by 2 CNOT gates. Therefore the whole permutation matrices can be implement on the quantum computer.

Based on recursive definition of Haar matrices, we can factorize H_{2^n} as

$$\begin{aligned} H_{2^n} = (I_{2^{n-1}} \otimes W)\cdots(I_{2^{n-i}} \otimes W \oplus I_{2^n-2^{n-i+1}})\cdots(W \oplus I_{2^n-2}) \times \\ (\Pi_4 I_{2^n-4})\cdots(\Pi_{2^i} I_{2^n-2^i})\cdots(\Pi_{2^{n-1}} I_{2^{n-1}})\Pi_{2^n} \end{aligned} \tag{42}$$

where W is the Hadamard matrix.

More on the Quantum wavelet transform can be found on [1] and [2].

6 Logistic Mapping

Chaotic dynamics was made popular by the computer experiments of Robert May and Mitchell Feigenbaum on a mapping known as the logistic map. The remarkable feature of the logistic map is in the simplicity of its form (quadratic) and the complexity of its dynamics. It is the simplest model that shows chaos. Through logistic map Feigenbaum number was discovered. It is the limit of the ratio between one bifurcation and the next, in logistic mapping; the same number turns up in various chaotic systems, it is about 4.669.

The logistic map is the simplest model in population dynamics that incorporates the effects of both birth and death rates. It is given by the formula:

$$x_{n+1} = f(x_n) = \lambda \cdot x_n \cdot (1 - x_n) \tag{43}$$

where the function f is called the logistic mapping and the parameter λ models the effective growth rate. The population size, (x_n) at the nth year, is defined relative to the maximum population size the ecosystem can sustain and is therefore a number between 0 and 1. The parameter λ is also restricted between 0 and 4 to keep the system bounded and therefore the model will make physical sense.

The logistic equation gives the rule for determining the relative population x_{n+1} at the (n+1)th year in terms of the population in the nth year. To get a physical understanding of the terms in the the logistic equation, we can think of the $\lambda \cdot x_n$ term as a positive feedback term in the sense that as x_n increases so does the value of b x_n. This is same as saying that the population size in the next year (x_{n+1}) is determined by the product of the previous population size x_n and the rate λ at which the population grows. Similarly, the term $(1 - x_n)$ can be thought of as a negative feedback, since increasing x_n will decrease $(1 - x_n)$ and therefore $(1 - x_n)$ can be thought of as population decline due to over population and scarce resources. Logistic mapping is the simplest one dimensional, nonlinear (x squared term), single parameter λ model that shows an amazing variety of dynamical response.

The graph corresponding to the logistic function $y = \lambda \cdot x \cdot (1 - x)$ is a parabola which passes through the points (0,0) and (1,0) independent of the choice of the parameter λ. The maxima of the parabola, which is always located at $x = 0.5$, is $0.25 \cdot \lambda$. There is a nice graphical visualization of the iteration process of logistic map via what is called the graphical iteration plot which shows how the iterates x_0, x_1, x_2, $\ldots$ can be obtained graphically.

It is clear that for values of λ between [0,1], if we start iterating the equation with any value of x the value of x will settle down to 0. This can be understood from the fact that the the logistic equation is a product of three numbers, namely, λ, x_n and $(1 - x_n)$ which are all between [0,1], x at the next time step must always be smaller than what is at the current time step. The point zero is called the fixed point of the system and is stable for $\lambda = [0, 1]$.

For values of λ between (1,3) the iterates instead of being attracted to zero, get attracted to a different fixed point. A fixed point is a point which when fed back into the map gives back the same point. Mathematically, this is expressed by the condition $x = \lambda \cdot x \cdot (1 - x)$. The long term behavior of the logistic equation when $\lambda \geq 3$, show that the system settles down to a period ≥ 2 limit cycle. Figure 6 shows that the initial x attracted to a single non-zero fixed point.

The existence of n period limit cycle is established by real solutions to

Figure 6: Iteration with $x_0 = 0.2$, and $\lambda = 2.8$

the equation $x = f^n(x)$, provided the solutions lie between 0 and 1.

On further increasing λ the period 2 limit cycle becomes unstable (at about $\lambda = 3.5$), and we get a period 4 cycle (see figure 7 and 8). The rate at which this doubling occurs increases, and by $\lambda = 3.56$ there is a period 8 cycle, by $\lambda = 3.567$ there is a 16-cycle. This continues infinitely, but happens so quickly that by $\lambda = 3.58$ it has finished. At this point the mapping becomes chaotic. There are more period doubling cascades to come though. At $\lambda = 3.835$ there is a period 3 cycle, doubling to 6, 12, 24 etc as λ is increased very slightly. There is a period 5 attractor at $\lambda = 3.739$, which again forms a period doubling cascade (5, 10, 20, 40,...). The simple logistic mapping produces this extremely complex mixture of chaos and order. This period doubling is called bifurcation.

The behavior of the logistic mapping can be described graphically by a bifurcation diagram. Figure 9 is a graph which plots the value of λ across, against the attractors of the sequence vertically. The single branch at the beginning represents the steady state, which branches into two. This is the period 2 cycle. Now we can graphically see the period doubling cascade, as the branches each divide into two, until there is chaos. Amidst the chaos, new branches spring up, and then double themselves - these are the new period doubling cascades.

The source code to make the iteration plot[10] can be found on the appendix One of the possible application is to study the chaotic behavior of the logistic mapping when λ is a critical value

7 Applications

Logistic mapping exhibit an interesting structure around the critical λ ($3.57 < \lambda < 3.58$). A fourier transform of the logistic mapping cannot tell much about the multi-level structure of the plot. The "self-similarity" on the logistic mapping is best studied using wavelet transform.

On our simulation, we vary the parameter λ to see several different structure. The program `run.m` will generate:

1. Logistic mapping for specific λ (using `logistic.m`)

[10]iteration.m

Figure 7: Iteration with $x_0 = 0.5$, and $\lambda = 3.2$

Figure 8: Iteration with $x_0 = 0.4$, and $\lambda = 3.52$

Figure 9: Bifurcation diagram

2. Haar transformed of the logistic mapping (using `haar.m`)
3. The average energy distribution across different resolution on linear scale (using `energy.m`)
4. The average energy distribution across different resolution on logarithmic (dB) scale.

The fixed parameter that we use are:

- $n = 10 \rightarrow N = 2^n = 1024$. We have 1024 data point.
- $L = 10 \rightarrow 2^L = 1024$. We iterate f 1024 times.

Here we present the plots with ten different λ.

7.1 Logistic

1. $\lambda = 1$

Figure 10: Logistic

Figure 11: Haar transformed

Figure 12: Linear scale

Figure 13: Logaritmic scale

2. $\lambda = 2.8$

Figure 14: Logistic

Figure 15: Haar transformed

Figure 16: Linear scale

Figure 17: Logaritmic scale

3. $\lambda = 3.1$

Figure 18: Logistic

Figure 19: Haar transformed

Figure 20: Linear scale

Figure 21: Logaritmic scale

4. $\lambda = 3.54$

Figure 22: Logistic

Figure 23: Haar transformed

Figure 24: Linear scale

Figure 25: Logaritmic scale

5. $\lambda = 3.57$

Figure 26: Logistic

Figure 27: Haar transformed

Figure 28: Linear scale

Figure 29: Logaritmic scale

6. $\lambda = 3.575$

Figure 30: Logistic

Figure 31: Haar transformed

Figure 32: Linear scale

Figure 33: Logaritmic scale

7. $\lambda = 3.5925721$

Figure 34: Logistic

Figure 35: Haar transformed

Figure 36: Linear scale

Figure 37: Logaritmic scale

8. $\lambda = 3.6785735$

Figure 38: Logistic

Figure 39: Haar transformed

Figure 40: Linear scale

Figure 41: Logaritmic scale

9. $\lambda = 3.7$

Figure 42: Logistic

Figure 43: Haar transformed

Figure 44: Linear scale

Figure 45: Logaritmic scale

10. $\lambda = 4$

Figure 46: Logistic

Figure 47: Haar transformed

Figure 48: Linear scale

Figure 49: Logaritmic scale

To proceed to a further analysis, we choose one λ ($\lambda = 3.57$) and lengthen the number of points to $2^{12} = 4096$. Since the haar transform that we have is combination of different scales and the average energy of coarser scale is much higher than finer level, we need to plot different scale separately.

7.2 Multi-Resolution

Figure 50: First scale

Figure 51: Second scale

Figure 52: Third scale

Figure 53: Fourth scale

Figure 54: Fifth scale

Figure 55: Sixth scale

Figure 56: Seventh scale

Figure 57: Eighth scale

Figure 58: Ninth scale

Figure 59: 10th scale

Figure 60: 11th scale

The best representation of these multi-resolution haar transform is by stack them one after another, and make an image representation of it. The brightness is related to the activity in that region (the mod square of the haar transformed signal).

Figure 61: Image representation of multi-resolution Haar transform

The structure of the logistic map around critical lambda are interesting. But in the "sea of chaos" there are λ that give a nice structure. For example the "island of period three" can be found at $\lambda = 3.835$. As we can see that the number of points is grow exponentially when we go down to the finer scale, and as λ is bigger 3.57 we need more iteration to get a better resolution. Therefore, to study the structure at these region, we need a powerful computer to do the computation.

At this situation quantum computer can give a better algorithm to solve the problem. Before going any further, we will explain how quantum mechanic do the computation.

Assume that we have a system with 16 energy states, which the probability to measure one of the energy state is proportional to energy of the state. The real energy distribution of the system (see figure 62) is unknown to us.

Therefore we need to do an experiment to find out the statistical distribution of the energy. The act of measuring the system will collapse the wave function of the system into one of its energy states. Here we have two sets of data from 100 measurements (figure 63 and 64) and two sets of data from 1000 measurements (figure 65 and 66). Here we can see that by hundred measurements we can get a good estimation of the energy distribution.

In the context of our simulation, this simulated histogram can be used to the level of activity across different scale, since the energy distribution is related to the haar transformed signal.

Other application of wavelet transforms in the field of science can be found on [12].

8 Conclusion

Quantum Haar transform implementation on quantum computer enable us to analyze fine scale structure with a high resolution, which our current classical

Figure 62: The real energy distribution

Figure 63: First set of experiment with 100 times measurement

Figure 64: Second set of experiment with 100 times measurement

computer takes enormous computational resource to do it.

9 Acknowledgment

I wish to thank my supervisor, Assoc. Prof. Wayne Lawton for giving me such a broad UROPS topic, and for his guidance throughout the whole year. I am grateful to A/P Lawton for introducing me to “new” mathematics fields. There are also others that have contributed and helped me directly in my project. In particular, I would like to thanks Dr. Kuldip Singh and A/P Kwek LC (NIE) for teaching me the essence of quantum mechanics, Mr. Christian Lee for helping me on making the bifurcation diagram, and Mr. Teo Kai Meng for helping me “hunting for bugs” in my programs..

Then there those assistance help me, without which this paper will not take on its shape. Here I express my gratitude to Mr. Daniel Kuan Li Oi (Oxford), Mr. Lim Kim Yong and Mr. Alexander Ling for helping me learning LaTeX, and then to Mr. You Sean Chung, Mr. Liang Yeong Chern and Mr. Chang Kelken for read proof some of my draft. I would like to express my gratitude to Dr. Chan Onn, Prof. Artur K. Ekert (Oxford) and Prof. Sandu Popescu (Bristol) for their encouragement and support. Many thanks to Open Source community that make “GNU Octave”, “GNU/Linux”, and “TeX” available freely, without which I will have to buy MatLab and had a hard-time to type in Microsoft Word.

And last but not least, I would like to thank all my friends, especially Adele Lim Tzu-Lin, that have accompany me for meals and have given me constant encouragements.

Figure 65: Third set of experiment with 1000 times measurement

Figure 66: Fourth set of experiment with 1000 times measurement

Appendix

A M-files source-code:

A.1 logistic.m

```
function y = logistic(lambda,L,x)
% function y = logistic(lambda,L,x)
%
% Darwin Gosal 2 June 2001
%
% Inputs:
% x = sampled values over [0,1]
% L = integer >= 0
% lambda = parameter in [0,4]
%?
% Ouputs:
% y = f^N(x) where N = 2^L
%
N = 2^L;
y = x;
for j = 1:N
   y = lambda*y.*(1-y);
end
temp=sqrt(sum(y.^2));
y = y/temp;
figure(1)
xlabel('x')
ylabel('y')
title('{\ity} = {\itf}^{N}({\itx})')
plot(x,y)
grid
xlabel('x')
ylabel('y')
```

A.2 haar.m

```
function [ha] = haar(a)
%
```

```
% function [ha] = haar(a)
%
% Darwin Gosal 7 June 2001
%
% Inputs:
% a = row array of length N=2^L
% L = integer >= 0
%
% Ouputs:
% ha = Haar transform of a
%  low-frequencies/course scale on left
%
N = size(a,2);
L = log(N)/log(2);
ha = a;
for j = L:-1:1
   M = 2^j;
   e = ha(1:2:M);
   o = ha(2:2:M);
   ha(1:M) = (1/sqrt(2))*[o+e o-e];
end
```

A.3 haar.m

```
function E = energy(ha)
%
% function E = energy(ha)
%
% Darwin Gosal 8 June 2001
%
% Inputs:
% ha = the signal after Haar transform
%
% Ouputs:
% E = energy level distribution across different level
%
k = log2(max(size(ha)))-1;

for i=1:k
```

```
 E(i)=0;
  for j=2^i+1:2^(i+1)
   E(i) = E(i) + ha(j)^2;
  end
 E(i)=E(i)/2^i;
end
Et = sum(E);
E = E/Et;
```

A.4 run.m

```
function [y, ha, E] = run(n,lambda,L)
%
% function run(n,lambda,L)
%
% Darwin Gosal 8 June 2001
%
% Inputs:
% n = the power of sample size N (N=2^n)
% L = number of iterations integer >= 0
% lambda = the variable of the logistic mapptinf
%
% Ouputs:
% It will make:
% y = which is the logistic mapping to the corresponding paramaters
% ha = which is the haar transform or y
% E = energy distribution of the mapping.
%

x = 0:1/(2^n-1):1;
y = logistic(lambda,L,x);
ha = haar(y);
figure(2);
title('Haar transform of the signal');
plot(5:2^n,ha(5:2^n));
E = energy(ha);
figure(3);
title('Energy distribution on dB scale');
plot(log10(E));
```

```
title('Energy distribution on linear scale');
figure(4);
plot(E);
```

A.5 mkimg.m

```
function img = mkimg(ha,l)
%
% function img = mkimg(ha,l)
%
% Darwin Gosal 11 June 2001
%
% Inputs:
% ha = the haar transform that going to be displayed as image
% l  = number of pixels per level
%
n = log2(size(ha,2));

for i=1:n-1
    c=2^(n-i-1);
    for j = 2^i+1:2^(i+1)
 k=j-2^i-1;
 img(i,k*c+1:(k+1)*c)=ha(j);
    end
end

temp=img;

for i=1:n-1
    for j=l*(i-1)+1:l*i
 img(j,:)=temp(i,:);
    end
end
```

A.6 simulate.m

```
function B = simulate(n,s)
```

```
% function B = simulate(n,s);
%?
% Darwin Gosal 10 June 2001
%
% Inputs:
% n = number of possible energy state
% s = number of points (degree of precision)
%
% Output:
% B = Energy distribution

A = rand(1,n);
A = A/sum(A);
A = round(A*s);
A(n)=A(n)+s-sum(A);
C = cumsum(A);A
B = ones(1,C(1));
for i=1:n-1
    for m=C(i)+1:C(i+1)
 B(m)=i+1;
    end
end

size(B)
hist(B,n)
```

A.7 iteration.m

```
function iteration(x0,a);
% function iteration(x0,lambda);
% Darwin Gosal 12 June 2001
% x0 = starting point
% lambda = growth rate
%
X = 0:1/(2^10-1):1;
for i=1:2^10
    F(i) = a*X(i)*(1-X(i));
    G(i) = X(i);
end
```

```
clear x;

for i=0:2:100
if i==0
 x(i+1)=x0;
 y(i+1)=a*x(i+1)*(1-x(i+1));
 x(i+2)=y(i+1);
 y(i+2)=y(i+1);
else
 x(i+1)=y(i-1);
 y(i+1)=a*x(i+1)*(1-x(i+1));
 x(i+2)=y(i+1);
 y(i+2)=y(i+1);
endif
end

plot(x,y)
hold
plot(X',F)
plot(X',G)
```

A.8 fwt.m

```
function h = fwt(m,l)

% function h = fwt(m,l)
%
% Dariwn Gosal, June 6, 2001
%
% Inputs:
% m = input size
% l = level
%
% Output:
% h = l level haar decomposition


if(nargin != 2)
```

```
 error("fwt:invalid number of arguments");
endif

n=log2(m);

if(l > n)
 error("fwt: 2nd parameter must be less the the log2 of the first parameter");
endif

h=eye(m);

for i=1:l

    B=[1 1];
    C=[1 -1];
    D=[]; E=[];
    k=m/2^i;
    for j=1:k
 D=blkdiag(D,B);
 E=blkdiag(E,C);
    end
    F=(1/sqrt(2))*[D zeros(k,m-2*k); E zeros(k,m-2*k);
                   zeros(m-2*k,2*k) eye(m-2*k)];

h = F*h;

end
```

References

[1] Andreas Klappenecker, *Wavelets and Wavelet Packets on Quantum Computers*. quant-ph/9909014

[2] Amir Fijany and Colin P. Williams, *Quantum Wavelet Transforms: Fast Algorithms and Complete Circuits*. quant-ph/9809004

[3] Michael A. Nielsen and Isaac L. Chuang, *Quantum computation and quantum information*. Cambridge University Press, 2000.

[4] Arthur O. Pittenger, *An introduction to quantum computing algorithms*. Birkhauser, 2000.

[5] Dirk Bouwmeester, Artur K. Ekert, and Anton Zeilinger, *The physics of quantum information : quantum cryptography, quantum teleportation, quantum computation*. New York, Springer, 2000.

[6] Jozef Gruska, *Quantum computing*. London, McGraw-Hill, 1999.

[7] Gennady P. Berman (et al.), *Introduction to quantum computers*. World Scientific, 1998.

[8] Chris J. Isham, *Lectures on quantum theory : mathematical and structural foundations*. Imperial College Press, London, 1995.

[9] John von Neumann, *Mathematical Foundations of Quantum Mechanics*. Princeton University Press, 1983.

[10] Macchiavello C., Palma G.M., Zeilinger A., *Quantum Computation and Quantum Information Theory*. World Scientific, 2000.

[11] Cabello A., *Bibliographic guide to the foundations of quantum mechanics and quantum information*. quant-ph/0012089

[12] Dremin I.M., Ivanov O.V., Nechitailo V.A., *Wavelets and Their Use*. hep-ph/0101182

[13] Richard Jozsa, *Quantum Algorithms and the Fourier Transform*. quant-ph/9707033

[14] John Preskill, *The Future of Quantum Information Science*. A talk at the NSF Workshop on Quantum Information Science, 28 October 1999.

[15] Ekert A., Hayden P., and Inamori H., *Basic Concepts in Quantum Computation*. quant-ph/0011013

[16] Gilbert Strang and Truong Nguyen, *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.

[17] Burrus C.S., Gopinath R.A., Guo H., *Introduction to Wavelets and Wavelet Transforms: A Primer*. Prentice-Hall, 1998.

[18] Charles K. Chui, *An Introduction to Wavelets*. Academic Press, 1992.

[19] Chan A.K. and Liu S.J., *Wavelet Toolware: Software for Wavelet Training*. Academic Press, 1998.