An Extension of the Deutsch-Jozsa Algorithm to Arbitrary Qudits

A Thesis

Submitted To the Faculty of Graduate Studies and Research

In Partial Fulfillment of the Requirements

for the Degree of

Master of Science

In the Department of Physics and Engineering Physics

University of Saskatchewan

by

Peter Marttala

In presenting this thesis in partial fulfilment of the requirements for the degree of Master of Science from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor who supervised my thesis work, Dr. C. Rangacharyulu, or alternatively, by the Head of the Department of Physics and Engineering Physics or the Dean of the College of Graduate Studies and Research. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Physics and Engineering Physics

University of Saskatchewan

Saskatoon, SK

S7N 5E2

Canada

Abstract

Recent advances in quantum computational science promise substantial improvements in the speed with which certain classes of problems can be computed. Various algorithms that utilize the distinctively non-classical characteristics of quantum mechanics have been formulated to take advantage of this promising new approach to computation. One such algorithm was formulated by David Deutsch and Richard Jozsa. By measuring the output of a quantum network that implements this algorithm, it is possible to determine with N – 1 measurements certain global properties of a function $f(x)$, where N is the number of network inputs. Classically, it may not be possible to determine these same properties without evaluating $f(x)$ a number of times that rises exponentially as N increases. Hitherto, the potential power of this algorithm has been explored in the context of qubits, the quantum computational analogue of classical bits. However, just as one can conceive of classical computation in the context of non-binary logic, such as ternary or quaternary logic, so also can one conceive of corresponding higher-order quantum computational equivalents.

This thesis investigates the behaviour of the Deutsch-Jozsa algorithm in the context of these higher-order quantum computational forms of logic and explores potential applications for this algorithm. An important conclusion reached is that, not only can the Deutsch-Jozsa algorithm's known computational advantages be formulated in more general terms, but also a new algorithmic property is revealed with potential practical applications.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CNOT	controlled NOT
EPR	Einstein – Polodosky – Rosen
GHZ	Greenberger – Horne – Zeilinger
PBS	polarizing beam splitter
XOR	exclusive OR

Chapter 1 – Introduction

1.1 Background

Quantum computation provides an entirely new paradigm by which to view the process of computation. From its inception, theoretical computational science has been formulated in terms of the Turing machine. Hitherto, all proposed modes of computation have been shown to be reducible to the Turing model. Quantum computing, in contrast, presents the opportunity to contemplate a non-Turing mode of computation and an associated extension in the range of computational applications.

The discipline of computer science, following a lengthy period characterized by ad hoc experiments and proposals, was placed on a systematic theoretical footing through the work of Alan Turing and Alonzo Church. Turing envisioned an idealized computing machine comprised of a symbolic alphabet $\{a_0, a_1, ..., a_N\}$, a moveable tape consisting of a finite number of non-blank cells each containing one symbol from the alphabetic set, a read/write head, and an internal machine state [1]. The operational details of this Turing machine are not of concern here, except to note that its features of a symbolic instruction set, a stored program, and automated function evaluation are standard in today's computing technology. Despite the vast diversity of ways in which this technology manifests, from embedded microcontrollers to supercomputers, the underlying principles can always be related to the Turing machine. Already by 1936, this observation was generalized in the Church-Turing thesis. In its modern form[1], the thesis states that

[1] Church and Turing did not envision that probabilistic algorithms could solve certain classes of problems more efficiently than any known deterministic algorithms. The recognition of this fact led to a generalization of the original Church-Turing thesis.

A probabilistic Turing machine can simulate any model of computation with at most a polynomial increase in the number of elementary operations required. [2]

In other words, the thesis claims that if an algorithm can be implemented on any computational platform, the same set of solutions produced can be obtained by way of a corresponding algorithm implemented on a Turing machine without substantial degradation in computational complexity. Computational complexity describes the relationship between the size of the input to an algorithm and the number of elementary operations that must be performed to obtain the desired solution. For example, if the number of operations increases according to some polynomial function of the input size, the computational complexity of the algorithm is denoted by $O(x^k)$, where x is the input size and k is the highest order exponent in the polynomial. Multiplying constants and lower order polynomial terms have a less significant effect[2] and hence are not included in this description. If the computational complexity of a given algorithm is of polynomial order or less, for example $O(\log x)$, then that program is said to be efficient. One can also consider algorithms in which the number of required operations grows exponentially with respect to input size. These *superpolynomial* algorithms, described as inefficient, when computed on a Turing machine are only able to reach a desired solution within a reasonable amount of time for small input sizes. For larger input sizes, the required processing time on any Turing machine rapidly becomes impracticably long. Unfortunately, many problems with practical application have a large input size and

[2] Clearly a term like $10{,}000x^2$ cannot be considered insignificant compared to x^3 unless one is employing extraordinarily large input sizes. In practice, however, nearly all multiplying constants encountered do not diverge from unity by more than one or two orders of magnitude and as such, do not substantially influence this rough, but insightful, means of quantifying computational complexity.

cannot be related to any known efficient algorithm. Given current computing technology, such problems cannot be computed within any realistic timeframe[3].

At no time has the Church-Turing thesis been established by means of rigorous proof. Nevertheless, the failure over the course of several decades to provide a counterexample that could not be ultimately demonstrated to be fundamentally equivalent to a Turing machine gave this thesis a wide acceptance amongst computational science researchers. The recent development of quantum computation, however, provides a long sought after exception to the Church-Turing thesis.

1.2 The Emergence of the Quantum Computational Paradigm

The foundations for the discipline of quantum mechanics were laid in the 1920s and 1930s. Until the 1970s however, the major applicative concern of this subject was the description of systems over which little control over individual constituents could be exercised. Superconductors for example, while admitting of a quantum mechanical description in excellent agreement with experiment, are generally manipulated as a bulk sample composed of a huge number of individual atoms. Even particle accelerators exercise only a restricted degree of control over the individual particles involved.

Since then, breakthroughs in experimental techniques have provided the means to exercise very precise control over individual systems. For example, experiments have demonstrated that atoms trapped in a superconducting cavity [3] or ions in a linear radio-

[3] This statement remains true despite the success of the empirical Moore's Law in describing the fact that the peak fabricable number of transistors per integrated circuit doubles every 18 – 24 months without a corresponding increase in unit cost. The increased number of floating-point operations per second this enables only reduces by a constant factor the overall computational time required for a Turing machine to achieve a solution to a particular problem. But the inherent mode of computation involved remains Turing, and as such, an inefficient algorithm remains inefficient.

frequency trap [4] can be manipulated by photons to achieve the fine manipulations necessary to construct quantum computational hardware.

Concurrent with these experimental developments, several researchers laid the theoretical groundwork for quantum computation. This process was slow at first, but gained impetus as the importance of potential quantum computational applications was realized.

Paul Benioff was the first to discuss the possibility of computation with quantum mechanical hardware [5]. Envisioning a grid of spin-1/2 particles, he described how these could be made to simulate classical logical gates, and more generally, realize the function of a Turing machine. Relative to later developments in the field, the mere simulation of a Turing machine is a rather restricted goal. However, prior to the publication of Benioff's paper, some uncertainty existed as to whether the Heisenberg uncertainty principle would induce fluctuations that would make any computer based on quantum mechanical principles even less capable than a Turing machine. Benioff's work not only definitively dispelled this concern, but also inspired others to think about quantum mechanics computationally.

The next step was taken by Richard Feynman, a key figure in the development of quantum electrodynamics. He noted that certain aspects of the dynamical evolution of quantum mechanical systems could not be simulated on a Turing machine without exponentially slower performance [6]. From this observation, he logically surmised that a simulation of a quantum mechanical system on a machine operating under inherently quantum mechanical principles would not run appreciably slower than the actual system being simulated. Feynman's goal was merely the abstract description of a universal

quantum simulator; however, it was later shown that this is fundamentally equivalent to the universal quantum computer[4] described by David Deutsch in 1985.

Deutsch's contribution originated from his consideration of the intimate connection between the process of computation and the physical basis upon which this process is implemented. Given that quantum mechanics is more general than classical mechanics, Deutsch reasoned by analogy that quantum computation may be more general – and hence may involve greater capabilities – than the classical Turing computational model. As will be described in detail in the next chapter, he noted that the uniquely quantum mechanical properties of superposition and entanglement provide computational resources not accessible within the Turing machine paradigm. These properties, combined with the ability to simulate a Turing machine on a quantum computer, suggested to him that quantum computation is a superset of classical computation. Deutsch laid out these considerations in his description of a universal quantum computer [7]. Among its capabilities, such a computer is capable of fully realizing Feymann's universal quantum simulator.

Despite the groundbreaking nature of this work, no applications of recognizable practical interest presenting a great improvement over their classical counterparts were immediately forthcoming. Deutsch supplemented his description of a universal quantum computer with a relatively simple example consisting of a way to determine in one function evaluation what would take a Turing machine two function evaluations to accomplish. A classical computer must evaluate both possible input combinations of a

[4] A "universal computer" is a machine capable of computing all problems that are computable. A Turing machine is a universal computer, despite the fact that it cannot compute all computable problems efficiently. What is fundamental to the definition of "universal" is that a solution can be computed given an arbitrarily large timeframe and other resources such as memory.

single bit binary function in order to determine if the function is constant or balanced[5]. A quantum computer on the other hand can make this determination with a single function evaluation. This increased efficiency becomes even more pronounced when a generalization is made to a binary function with a multiple bit domain. If we know in advance that the function is either constant or balanced, a maximum of $\frac{N}{2}+1$ function evaluations must be made classically in order to make this determination, where N is number of possible input combinations. A more generalized version of this algorithm, first described by Richard Jozsa in collaboration with David Deutsch [8] and known as the Deutsch-Jozsa algorithm, can make this determination within O(log N) steps. Therefore, a substantive difference in computational complexity exists between the classical and quantum versions of this algorithm.

A major breakthrough came in 1994 with Peter Shor's publication of an algorithm for a quantum computer capable of efficiently reducing large numbers to their prime factors [9]. The longstanding failure to provide an efficient classical algorithm capable of performing this task provided cryptographers a seemingly good reason to use the difficulty of factoring large numbers as a basis for a popular cryptosystem used to guarantee the privacy of sensitive communication channels. Another important development emerged with Lov Grover's publication of a quantum algorithm capable of searching for a specified element within an unstructured search space with a quadratic decrease in computational complexity over the fastest known classical algorithm [10]. The quantum algorithm for random walk graph traversal [11] likewise provides an improvement in computational complexity over its fastest classical counterpart.

[5] A balanced function's range consists of two values, both of which appear with equal frequency.

1.3 The Current Status of the Discipline

Despite the promise held by Shor's factoring algorithm and others, certain difficulties still overshadow the entire field of quantum computation. For example, the development of new quantum algorithms capable of solving problems of practical interest is non-intuitive and hence difficult for the human accustomed to classical modes of thinking. Indeed, the range of possible quantum algorithms capable of improving the computational complexity of a given problem with respect to the fastest known classical algorithm is still undefined. In consequence, one cannot determine if the rarity of known quantum algorithms is simply due to their non-intuitive nature or the potential ontological reality that few such algorithms are possible. Given that no known automated procedure exists to generate quantum algorithms, only human creativity will in time tell us whether a large set of as yet unknown such algorithms in fact exists.

Moreover, attempts at practical realizations of quantum computing hardware quickly run into the problem of decoherence. Throughout the entire process of performing such a computation, the system must be guarded against being influenced by unwanted interactions with its surroundings lest the quantum states involved collapse or become entangled with external entities thereby destroying the meaningfulness of the results obtained. Currently, the engineering challenges associated with creating quantum networks satisfying this criterion prevent quantum computers containing more than a few gates from being built. For example, the capability of a very simple quantum computer to factor the number 15 using Shor's algorithm has been demonstrated [12]. However, successful techniques to overcome decoherence must first be established before the

ability to efficiently factor numbers large enough to threaten the security of some currently popular cryptosystems could emerge.

Nevertheless, the enticing possibilities that practical quantum computation offers are sufficient reason to investigate the potentialities of this nascent discipline and seek solutions to the variety of challenges that researchers must face. The potential for efficient computation of classically inefficient algorithms would not only be beneficial to the advancement of theoretical computer science, but may also produce economic benefits. For example, Lov Grover's quantum search algorithm could be implemented as a means to achieve substantial reductions in the time required to query a database. More generally, some portion of the global economic successes and productivity increases over the past few decades can arguably be attributed to the dramatic increase in the power and speed of computation coupled with a corresponding reduction in its cost as described by "Moore's Law".

However, as the size of integrated circuit components continues to decrease, eventually quantum effects will become unavoidable. To avoid these non-classical effects, such as electron tunneling, one must either ensure that all circuit components remain above a certain minimum size or develop new computational technology that employs these effects as a crucial aspect of its functionality[6]. Although the designers of classical integrated circuits can continue to achieve refinements for some time, only the second approach promises substantial improvements over current computing technology.

[6] Certain non-classical computational platforms which merely emulate a Turing machine have been proposed and investigated to varying degrees. For example, single electron transistor and "quantum dot" technology offer the potential to construct nanoscale components which replicate the functionality of their classical counterparts. Despite quantum effects being vital to their intended operation, they do not provide any new computational resources. These technologies will not be considered further in this thesis.

Correspondingly, it is this approach which offers the greatest promise for spin-off economic benefits in the long term.

In light of this motivation to investigate the promise of quantum computation, I believe that it is worthwhile to study the behavior of quantum algorithms in the context of higher-order quantum logic. Just as it is possible to study non-binary modes of classical computation that employ, for example, ternary or quaternary logic, so also is it possible to study quantum computation in the context of equivalent higher-order forms of quantum logic. In particular, this thesis will develop a generalization of the Deutsch-Jozsa algorithm in the context of arbitrary order quantum logic and investigate its properties towards the goal of determining those behavioral characteristics that may yield future practical application.

Chapter 2 – Quantum Computation and the Deutsch-Jozsa Algorithm

2.1 Introduction to Quantum Information Theory

Within the context of computation, the basic classical unit of information is the bit. Regardless of whether a bit is physically realized by voltage levels across a transistor, the on and off states of a vacuum tube or by some other means, only two bit states are possible. Moreover, every bit intrinsically constitutes a complete state in and of itself. Bits may be stored in a memory unit or register for future use or to make a datum available for logical or arithmetical manipulation. Clearly, a N bit register can only hold one quantity generated from the combination of the individual states of all N bits at any one time. For example, a byte is uniquely represented by the sequence of the eight bit states of which it consists.

Quantum computation, on the other hand, offers the two additional properties of *superposition* and *entanglement* that are unavailable within the classical paradigm. Together, these two computational resources provide the potential for a quantum computer to outperform any Turing machine in certain circumstances. To illustrate the differences between classical and quantum computation, it is first necessary to define the basic unit of quantum information, the *qubit*.

Like the bit, the qubit is an abstract information unit that has no ontological connection to any one method of physical realization[7]. The qubit exists in a two

[7] To clarify, all information, whether classical or quantum, must possess some physical representation. However, the precise means by which this information is represented does not affect its "message". Hence the qubit, as an abstract unit of quantum information, can be physically realized by several different means, yet each method gives the qubit the same ability to express the same range of information.

dimensional complex linear vector space[8] and may be defined by the selection of two orthogonal states, denoted $|0\rangle$ and $|1\rangle$, which together form the computational basis. If an arbitrary qubit $|\psi\rangle$ is expanded across this basis, the result may be written as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \tag{2.1}$$

where $|\alpha|^2 + |\beta|^2 = 1$. The quantities $|\alpha|^2$ and $|\beta|^2$ give the probabilities that a measurement of $|\psi\rangle$ will yield $|0\rangle$ or $|1\rangle$ respectively. In certain instances, it may prove convenient to parameterize $|\psi\rangle$ by introducing θ and δ to give

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\delta}\sin\frac{\theta}{2}|1\rangle = \begin{pmatrix} \cos\frac{\theta}{2} \\ e^{i\delta}\sin\frac{\theta}{2} \end{pmatrix}, \tag{2.2}$$

where $0 \le \theta \le \pi$ and $0 \le \delta < 2\pi$. As no physical significance is attached to the overall phase of $|\psi\rangle$, one variable has been eliminated from this expression by requiring the $|0\rangle$ coefficient to be real. One can now introduce **P**, termed the polarization vector[9], to elucidate further upon the physical meaning of the qubit. This vector is defined by the expectation values of the Pauli matrices σ_x, σ_y and σ_z taken with respect to $|\psi\rangle$. For example,

[8] For the purposes of quantum computation, the vector spaces in which all quantum information units exist are always dimensionally finite, not nondenumerably infinite Hilbert spaces, as are those required to describe the position or momentum of a free particle.

[9] The use of the term "polarization" here does not necessarily imply any physical actualization of this system in the context of polarized photons. Historically speaking, however, this discussion was first formulated in that context, hence the name.

$$P_x = \langle \psi \, | \sigma_x | \psi \rangle = \left(\cos\frac{\theta}{2}, e^{-i\delta} \sin\frac{\theta}{2} \right) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \cos\frac{\theta}{2} \\ e^{i\delta} \sin\frac{\theta}{2} \end{pmatrix} = \sin\theta \cos\delta \ . \tag{2.3}$$

Likewise,

$$P_y = \sin\theta \sin\delta \ , \tag{2.4}$$

$$P_z = \cos\theta \ . \tag{2.5}$$

These three components are easily recognized as the parameterization of a unit sphere in spherical coordinates, where θ is the polar angle between **P** and the z axis and δ is the azimuthal angle measured from the x axis. Therefore, the state of an arbitrary qubit can be conceptualized as a point on the surface of a unit sphere, which is given the special designation *Bloch sphere* in the context of quantum computation. Consider the special case $\delta = 0$ and $\theta = 0$. In this scenario, $P_x = P_y = 0; P_z = 1$, corresponding to the state $|0\rangle$. Likewise, $\delta = 0$ and $\theta = \pi$ yields $P_x = P_y = 0; P_z = -1$, corresponding to the state $|1\rangle$. These two states are analogous to the only two possible states a classical bit may assume. In contrast, a qubit may exist in any one of the infinite states that correspond to points on the surface of the Bloch sphere[10]. These states, with the exception of the two special cases already noted, are known as *superpositions*.

[10] It should be emphasized, however, that the measurement of a qubit will yield only $|0\rangle$ or $|1\rangle$ according to their associated probability distributions relative to the axis of measurement. The parameters θ and δ can only be precisely determined by conducting an identical measurement procedure on an ensemble of infinitely many identically prepared systems.

All of the possible qubit states discussed thus far are known as pure states[11]. Although a measurement of an arbitrary qubit $|\psi\rangle$ will yield $|0\rangle$ with probability $\cos^2\frac{\theta}{2}$ and $|1\rangle$ with probability $\sin^2\frac{\theta}{2}$, one can always rotate the entire axis of measurement to obtain either outcome with certainty. In contrast, a measurement of a mixed state cannot obtain any single outcome with certainty regardless of the orientation of the measurement apparatus. As its name implies, a mixed state is a combination of at least two differently prepared pure states. Even if the measurement apparatus should happen to be aligned with one of the pure states of which a mixed state is composed, the probability of obtaining that pure state as a measurement outcome is equal to the fractional proportion of that state within the mixed state.

A storage unit containing N qubits is called a quantum register of size N. If the state of each qubit in this register is in a superposition, the register encompasses a total of 2^N states. For example, given two qubits in the state

$$\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right), \tag{2.6}$$

a quantum register that contains those two qubits encompasses the four states described by the superposition

$$\begin{aligned}\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)\otimes\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) &= \frac{1}{2}\left(|0\rangle\otimes|0\rangle + |0\rangle\otimes|1\rangle + |1\rangle\otimes|0\rangle + |1\rangle\otimes|1\rangle\right)\\ &= \frac{1}{2}\left(|00\rangle + |01\rangle + |10\rangle + |11\rangle\right).\end{aligned} \tag{2.7}$$

[11] A pure state is usually defined as an ensemble of identically prepared systems. However, in light of the precise control over individual systems that the physical realization of quantum computation necessitates, this definition will be extended to include the states of singly prepared systems, such as a single atom in a linear ion trap.

Should such a register serve as an input source to some quantum logic gate, its operation is simultaneously performed upon all states. Classically, this effect could only be accomplished through the execution of 2^N separate operations. Hence, quantum algorithms that make advantageous use of this property can transform an $O(2^N)$ algorithm into an O(N) one[12]. Some classically inefficient problems may become practicably solvable using a quantum computational approach.

If each qubit in a quantum register of size N exists in vector space V_i, where $1 \leq i \leq N$, then the entire system held by that register exists in vector space $V = V_1 \otimes V_2 \otimes \ldots \otimes V_N$. Assuming that each qubit in the register constitutes a complete state in and of itself, a property shared by all classical bits, then the register may be represented as

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes |\psi_3\rangle \otimes \ldots \otimes |\psi_N\rangle = |\psi_1 \psi_2 \psi_3 .. \psi_N\rangle, \tag{2.8}$$

where each element $|\psi_i\rangle \in \{|0\rangle, |1\rangle\}$ and $\psi_i \in \{0,1\}$. As natural as this representation may seem however, it is not the most general expression of an arbitrary N qubit register. That expression is given by

$$|\psi\rangle = \sum_{i,j,k,\ldots}^{N} c_{ijk\ldots} |i\rangle \otimes |j\rangle \otimes |k\rangle \otimes \ldots, \tag{2.9}$$

where $c_{ijk} \in \mathbf{R}$ and $\sum_{i,j,k,\ldots}^{N} |c_{ijk\ldots}|^2 = 1$.

To take an example, assume that

[12] The notation O(x) refers to the computational complexity of a particular problem. O(N) refers to the computational time required by the algorithm increasingly linearly with the size N of the input. $O(k^N)$, where k is some constant, refers to the computational time increasing exponentially with the size of the input.

$$|\psi\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle + |10\rangle\right). \tag{2.10}$$

Unlike an expression such as

$$|\phi\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle + |11\rangle\right) = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \otimes |1\rangle, \tag{2.11}$$

(2.10) cannot be written as a tensor product. In this instance, the two qubits constituting $|\psi\rangle$ do not form a product of two self-contained states, but a single joint state, termed an *entangled* state. This entangled state exists in vector space V, which cannot be reduced to a product of vector subspaces.

In general, entanglement is a characteristic associated with any state $|\psi\rangle$ that cannot be reduced to a simple tensor product. This uniquely quantum mechanical property, while offering potential resources unavailable within the classical computing paradigm, is also partly responsible for the non-intuitive nature of the subject. In a seminal paper[13] describing how the fundamental principles of quantum mechanics necessitate the existence of entangled states, Albert Einstein and his two co-authors, Boris Podolsky and Nathan Rosen, argued that this fact exposes a fundamental incompleteness[14] in the formulation of quantum mechanics itself. They based their argument on the supposition that the reality of entanglement would violate the intuitive notion of locality[15]. As they demonstrated in their paper, a measurement conducted upon

[13] Henceforth "EPR" will denote all references to this paper and the argument it sets forth.

[14] Contrary to a common misconception, nowhere does EPR challenge the correctness of quantum mechanics. A theory may be correct in the sense that its predictions exhibit no contradiction with the results of appropriate measurements, but is incomplete in that it fails to describe all the elements of reality present within the system being studied. For example, statistical thermodynamics is a correct theory, but is incomplete because it is ignorant of the states of the individual atoms or molecules present within the system of interest.

[15] *Locality* refers to the property that all interactions occurring within a system propagate via forces or signals that travel at luminal or sub-luminal speeds. David Bohm and others have constructed nonlocal

one member of a widely separated pair of entangled particles would, under the assumption of locality, lead to a contradiction. However, Niels Bohr and others have responded by questioning their assumption of locality.

2.2 Introduction to Quantum Computational Theory

As carriers of information, qubits, like classical bits, must be manipulated in order to obtain useful computational results. To accomplish this goal, various quantum logic gates have been developed which may be linked to form a topological network. Each gate performs a specified unitary operation on a set of input qubits within a fixed period of time [13], a transformation that may be denoted as

$$|\psi\,'\rangle = U|\psi\,\rangle, \tag{2.12}$$

where U is some unitary operator[16]. From this relationship, it follows that quantum computation must be reversible, a constraint which restricts the range of possible quantum computational gates. For example, a classical two input AND logic gate maps three of its four possible input state combinations to the logical output 0. This irretrievable loss of information about the gate's input state means that a classical AND gate has no direct quantum computational equivalent[17].

The simplest class of quantum gates act on a single qubit. Quantum gates in this category may be conceptually visualized as rotating a qubit relative to the Bloch sphere. Of these gates, the simplest is the no-op (no operation), equivalent in function to an ideal

---

interpretations of quantum theory which explicitly deny this presupposition.

[16] Any function U that satisfies the property $U^t U^{-1} = I$, where U^t is the Hermitian conjugate of U, is unitary.

[17] Despite the lack of any direct quantum computational analogue, the logical AND function can be realized as a subset of the quantum Toffoli gate's functionality.

wire[18]. It is represented by the identity matrix. The single qubit NOT gate, equivalent to the Pauli matrix

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \tag{2.13}$$

interchanges the qubit's $|0\rangle$ and $|1\rangle$ components. Unlike these gates, the Hadamard gate lacks any classical analogue as its function is to convert a qubit into a superposition according to the rule

$$|x\rangle \rightarrow \frac{1}{\sqrt{2}}\left((-1)^x|x\rangle + |1-x\rangle\right), \tag{2.14}$$

which may be represented in matrix form (within the basis $\{|0\rangle,|1\rangle\}$) as

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{2.15}$$

The phase-shift gate provides another useful single qubit network building block. Its most general form is represented as

$$\frac{1}{\sqrt{2}}\begin{pmatrix} e^{i\phi_1} & 0 \\ 0 & e^{i\phi_2} \end{pmatrix}, \tag{2.16}$$

where ϕ_1 and ϕ_2 are phase-shift angles. As an example, if $\phi_1 = \frac{\pi}{2}$ and $\phi_2 = 0$, the matrix becomes

$$\frac{1}{\sqrt{2}}\begin{pmatrix} i & 0 \\ 0 & 1 \end{pmatrix}. \tag{2.17}$$

[18] A "wire" refers to any process by which qubits are transferred unchanged from one stage of the network to another. Included here are teleportation of the qubit's state or some form of intermediary storage and transport mechanism, which may involve a significant time delay.

This operation, also known as the S gate, applies a phase shift of i to any input $|0\rangle$, but leaves $|1\rangle$ unaffected. Likewise, one can define a T gate as follows:

$$\frac{1}{\sqrt{2}}\begin{pmatrix} -1 & 0 \\ 0 & -i \end{pmatrix}. \tag{2.18}$$

Another commonly encountered gate is the rotation gate. It takes the general form

$$\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}, \tag{2.19}$$

where θ is the rotation angle. One particularly useful special case is the L gate, obtained by taking $\theta = -\frac{\pi}{4}$, is represented by the matrix

$$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}. \tag{2.20}$$

Its counterpart, the R gate, is obtained by taking $\theta = \frac{\pi}{4}$, leading to

$$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}. \tag{2.21}$$

Just as a special quantum gate, like the Hadamard gate, is required to take advantage of the potential to create superpositions, one needs another uniquely quantum gate to generate entangled states. Clearly such a gate must receive at least two qubits as input. Although many such "entangling gates" are conceivable, the controlled NOT (CNOT) gate is most commonly encountered. One of its inputs serves as a 'trigger' such that the state of the other input qubit is flipped if the trigger assumes a certain value,

typically $|1\rangle$. Otherwise no operation is performed. The two input CNOT gate triggered on the first qubit may be represented as

$$|a,b\rangle \rightarrow |a,a \oplus b\rangle, \tag{2.22}$$

where $a \oplus b$ is the exclusive-or (XOR) operation. In matrix form, this becomes

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.23}$$

See Figure 2.1 for the standard symbols used to represent these gates schematically in quantum network layouts.

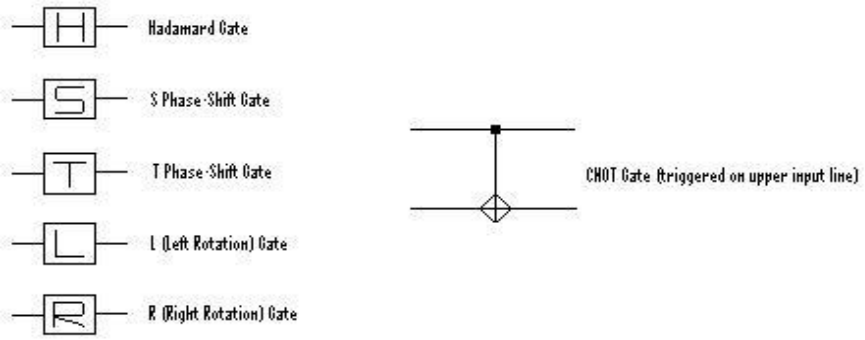

Figure 2.1: Schematic representations of some common quantum gates.

For more comprehensive and in-depth coverage of these and other issues related to quantum information and computation, various texts are available focusing on both theoretical and experimental aspects. Nielsen and Chuang [14] is quite comprehensive and is widely considered a standard introduction to the discipline. Other texts include Berman, Doolen, Mainieri and Tsifrinovich [15] as well as Lo, Popescu and Spiller [16] and Benenti, Casati and Strini [17].

2.3 Generalization to Higher-Order Logic

As the discipline of classical computation science matured, widespread agreement was achieved to standardize the representation of data in binary form. Given the inherently dual state nature of vacuum tubes and transistors, this choice was perhaps inevitable in retrospect. However, considerable experimentation with tri-state logic and other alternative means of representing data occurred throughout the nascent stage of the discipline. While these efforts largely remain a historical footnote in the history of classical computational science, they may offer a useful lesson for the future development of quantum computational hardware. As noted in the previous chapter, a variety of physical realizations of quantum computational technology have been proposed, not all of which intrinsically favour the quantum computational equivalent of binary logic. For example, an atom capable of being reliably energized to each of two excited states on demand offers, along with its ground state, the physical basis for representing a tri-state quantum bit, known as a qutrit[19]. As no wide agreement yet exists on what physical basis would most practicably and economically serve as a platform for quantum computation, it is worthwhile to investigate how quantum computation in general, and specific quantum algorithms in particular, behave in the context of higher-order quantum logic.

The basic quantum unit of information, or qudit, of an arbitrary logical order M exists in a vector space consisting of M orthogonal elements[20], denoted collectively as $\{|0\rangle, |1\rangle, |2\rangle, \ldots, |M-1\rangle\}$. Any number of these elements may exist in a superposition:

[19] A four-state quantum bit is referred to as a ququad. A five-state quantum bit is a ququint, a six-state quantum bit is a qusext, a seven-state quantum bit is a qusept, and an eight-state quantum bit is a quoct. A generalized quantum bit is referred to as a qudit when no specific logical state is implied.

[20] Although one could mathematically describe a generalized higher dimensional analogue to the Bloch sphere described above, little conceptual advantage could be gained from such an exercise.

$$\alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle + \ldots = \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \vdots \end{pmatrix}, \tag{2.24}$$

where $|\alpha|^2 + |\beta|^2 + |\gamma|^2 + \ldots = 1$. A register can be used to store an arbitrary number of qudits. The description of such a register is identical to the special qubit case, except that now each element $|x_i\rangle \in \{|0\rangle, |1\rangle, |2\rangle, \ldots, |M-1\rangle\}$, where $x_i \in \{0,1,2,\ldots,M-1\}$. If a register holds N qudits of logical type M, then that register can contain up to M^N unique states. For example, a register containing N qutrits holds up to 3^N different states.

The logical gates required to manipulate qubits may be generalized to accommodate higher-order qudits. For example, the Hadamard gate generalizes to

$$H = \frac{1}{\sqrt{M}} \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \sigma^{(M-1)} & \sigma^{2(M-1)} & \sigma^{3(M-1)} & \cdots & \sigma^{(M-1)(M-1)} \\ 1 & \sigma^{(M-2)} & \sigma^{2(M-2)} & \sigma^{3(M-2)} & \cdots & \sigma^{(M-1)(M-2)} \\ 1 & \sigma^{(M-3)} & \sigma^{2(M-3)} & \sigma^{3(M-3)} & \cdots & \sigma^{(M-1)(M-3)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \sigma & \sigma^{2} & \sigma^{3} & \cdots & \sigma^{(M-1)} \end{pmatrix}, \tag{2.25}$$

where $\sigma = e^{\frac{i2\pi}{M}}$ and M is the logical order [18]. Given that $e^{i2\pi} = 1$, (2.25) may be simplified to the symmetric form

$$H = \frac{1}{\sqrt{M}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 1 \\ 1 & \sigma^{(M-1)} & \sigma^{2(M-1)} & \cdots & \sigma^{2} & \sigma \\ 1 & \sigma^{(M-2)} & \sigma^{2(M-2)} & \cdots & \sigma^{4} & \sigma^{2} \\ 1 & \sigma^{(M-3)} & \sigma^{2(M-3)} & \cdots & \sigma^{6} & \sigma^{3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & \sigma & \sigma^{2} & \cdots & \sigma^{(M-2)} & \sigma^{(M-1)} \end{pmatrix}. \tag{2.26}$$

The CNOT gate may also be generalized to arbitrary qudits. In this instance however, the generalized CNOT can no longer be said to perform the function of controlled negation. Rather, the generalized CNOT becomes a controlled-summation gate, modulo M.

2.4 Deutsch Algorithm

As mentioned in the introductory chapter, one of the earliest proposed quantum algorithms was the Deutsch algorithm for determining in one evaluation whether a function is constant or balanced. The quantum network that implements[21] this algorithm, shown in Figure 2.2, takes two inputs, $|0\rangle$ and $|1\rangle$, respectively. Each input is separately subjected to a Hadamard operation, resulting in the overall system state

$$H|0\rangle \otimes H|1\rangle = \frac{1}{2}\left[\left(|0\rangle + |1\rangle\right) \otimes \left(|0\rangle - |1\rangle\right)\right] = \frac{1}{2}\left[|0\rangle \otimes \left(|0\rangle - |1\rangle\right) + |1\rangle \otimes \left(|0\rangle - |1\rangle\right)\right]. \quad (2.27)$$

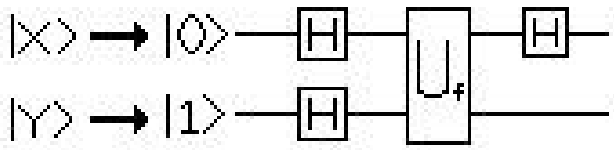

Figure 2.2: A schematic diagram of a quantum network that implements Deutsch's algorithm.

Operator U_f performs the exclusive-OR transformation,

$$|x, y\rangle \rightarrow |x, y \oplus f(x)\rangle, \quad (2.28)$$

where *f*(x) is a definable function with a purely binary mapping, $f : \{0,1\} \rightarrow \{0,1\}$. Depending upon how *f*(0) and *f*(1) are defined, this gives four possible combinations, two

[21] Throughout the remainder of this chapter, the discussion will presuppose quantum binary logic and the use of qubits to represent data.

of which are constant $\left(f(0) = f(1)\right)$ and two of which are balanced $\left(f(0) = \overline{f(1)}\right)$. Given that $\left|y \oplus \left(f(x) = 0\right)\right\rangle = \left|y\right\rangle$ and $\left|y \oplus \left(f(x) = 1\right)\right\rangle = \left|\overline{y}\right\rangle$, four possible outcomes result from the operation $U_f\left(H\left|0\right\rangle \otimes H\left|1\right\rangle\right)$:

(a) $f(0) = f(1) = 0$ $\quad \frac{1}{2}\left(\left|0\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right) + \left|1\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right)\right)$,

(b) $f(0) = f(1) = 1$ $\quad -\frac{1}{2}\left(\left|0\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right) + \left|1\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right)\right)$,

(c) $f(0) = 0, f(1) = 1$ $\quad \frac{1}{2}\left(\left|0\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right) - \left|1\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right)\right)$,

(d) $f(0) = 1, f(1) = 0$ $\quad -\frac{1}{2}\left(\left|0\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right) - \left|1\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right)\right)$.

These four outcomes can be summarized as follows:

$$U_f\left(H\left|0\right\rangle \otimes H\left|1\right\rangle\right) = \frac{1}{\sqrt{2}} \sum_{x=\{0,1\}} (-1)^{f(x)} \left|x\right\rangle \otimes \left(\frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}}\right). \tag{2.29}$$

Of these four possible outcomes, (a) differs from (b), as does (c) from (d), only by a measurably indistinguishable global phase factor. Therefore only two unique outcomes exist. Applying the final Hadamard gate to $\left|x\right\rangle$ yields the final state

$$\frac{1}{2}\left[(-1)^{f(0)}\left(\left|0\right\rangle + \left|1\right\rangle\right) + (-1)^{f(1)}\left(\left|0\right\rangle - \left|1\right\rangle\right)\right] \otimes \left(\frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}}\right). \tag{2.30}$$

If a measurement is taken at the output corresponding to the $\left|x\right\rangle$ input, four possibilities must again be considered:

(a) $f(0) = f(1) = 0 \Rightarrow \left|0\right\rangle$,

(b) $f(0) = f(1) = 1 \Rightarrow -\left|0\right\rangle$,

(c) $f(0) = 0, f(1) = 1 \Rightarrow |1\rangle$,

(d) $f(0) = 1, f(1) = 0 \Rightarrow -|1\rangle$.

A single measurement, therefore, will yield $|0\rangle$ if function *f*(x) is constant and $|1\rangle$ if it is balanced.

2.5 Deutsch-Jozsa Algorithm

Following the publication of Deutsch's description of the aforementioned algorithm, Richard Jozsa explained how it could be generalized to an arbitrary number N of qubits [8]. His proposed generalization is schematically outlined in Figure 2.3 [19].

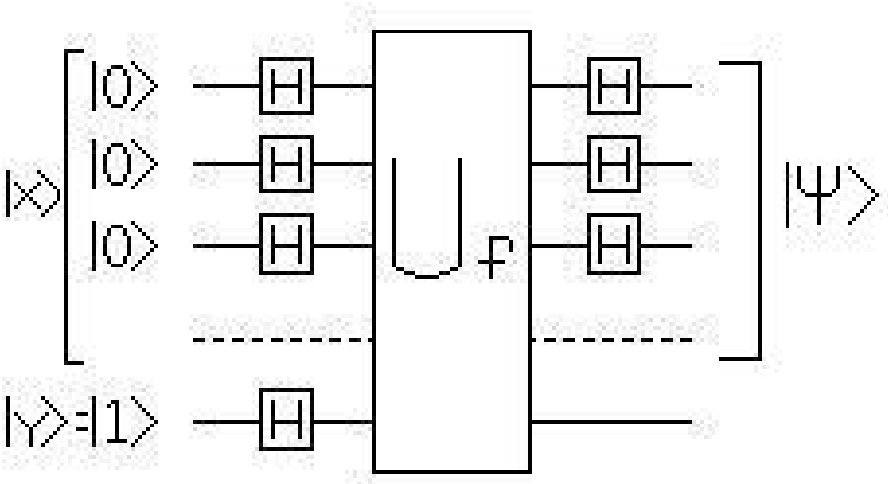

Figure 2.3: A schematic diagram of a quantum network that implements the Deutsch-Jozsa algorithm. The N – 1 inputs collectively denoted $|x\rangle$ are all set to $|0\rangle$ and the final input $|y\rangle$ is set to $|1\rangle$. The final state $|\psi\rangle$ represents the state of a subset of the system immediately prior to its being measured.

As before, a Hadamard operation acts upon each input prior to the evaluation of the function. The overall action of N simultaneous Hadamard operations shall be denoted as

$$H^{\otimes N} = H \otimes H \otimes \ldots \otimes H . \tag{2.31}$$

These operations, applied to the input state $|0\rangle^{\otimes N-1} \otimes |1\rangle$, yield the state

$$\frac{1}{\sqrt{2^{N-1}}} \sum_{x=0}^{2^{N-1}-1} |x\rangle \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right). \tag{2.32}$$

Operator U_f produces the same exclusive-OR transformation

$$|x, y\rangle \rightarrow |x, y \oplus f(x)\rangle, \tag{2.33}$$

encountered previously, except that *f*(x) now maps

$$f : \{0, 1\}^N \rightarrow \{0, 1\} . \tag{2.34}$$

Application of U_f yields

$$U_f \left(H^{\otimes N} |0\rangle^N \otimes |1\rangle \right) = \frac{1}{\sqrt{2^{N-1}}} \sum_{x=0}^{2^{N-1}-1} (-1)^{f(x)} |x\rangle \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right). \tag{2.35}$$

The final set of Hadamard operations yields

$$\sum_{z=0}^{2^{N-1}-1} \sum_{x=0}^{2^{N-1}-1} \frac{(-1)^{x \cdot z + f(x)} |z\rangle}{2^{N-1}} \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right). \tag{2.36}$$

Assuming that one knows in advance that function *f*(x) is either constant or balanced, a single sequence of measurements to determine $|\psi\rangle$ will again yield this information.

Chapter 3 – Deutsch's Algorithm and the Deutsch-Jozsa Algorithm Generalized to Arbitrary Qudits

3.1 Motivation

As an emerging discipline, quantum computing offers the investigator many exciting prospects for further development. If one attempts to glean from the history of classical computational science possible historical parallels relevant to the future direction that quantum computing may take, two points become evident. Not only were several decades, the attention of many participants and a considerable process of trial and error required for the discipline to attain a solid theoretical foundation, but also the sundry breakthroughs and advances that ultimately shaped the field could not have been predicted by the early investigators. Given the lack of predictability inherent in any nascent discipline, it is worthwhile to investigate many different avenues of possible future development with the intention of giving others the data they will need to give greater fixity to their thinking regarding which approaches will most satisfactorily reach the desired goals.

One such promising avenue of future development involves extending the description of already known quantum algorithms to qudits of an arbitrary logical order. Given the ubiquitous nature of binary data processing in classical computational science, it is not surprising that a strong favoritism exists towards the formulation of quantum algorithms within the context of binary quantum logic, using qubits as the quantum information carriers. As previously noted however, there presently exists an ambiguity as to how quantum computation would be most practically and economically realized. With

certain physical systems being naturally suited to higher-order logic, one cannot discount the possibility that higher-order quantum logic will ultimately play a much more significant role than its classical counterpart does. With these considerations in mind, it is evidently worthwhile to explore how known quantum algorithms would behave in the context of higher-order logic. Moreover, it is possible that the higher-order analogues of known quantum algorithms will solve certain classes of problems even more efficiently than can their counterparts formulated only in terms of quantum binary logic.

3.2 General Approach

To determine the characteristics of Deutsch's algorithm and the Deutsch-Jozsa algorithm[22] in the context of higher-order logic, I chose to implement a quantum network level simulation of these algorithms using the programming language GNU Octave, version 2.1.42, generously provided free of charge by the GNU Foundation from the website www.octave.org. To validate the results obtained thereby, I derived a generalized version of Equation 2.35 that holds for arbitrary qudits and wrote a short function in Octave to compute it. Using these two parallel methods, I obtained a set of data from which I sought general patterns regarding the behavior of these algorithms in the context of higher-order logic, paying special attention to those which may have future practical quantum computational applications.

In the course of this work, I assumed that all quantum logic gates and other network elements are ideal and that the surrounding environment introduces no undesirable interactions with respect to the system of interest.

[22] As Deutsch's algorithm can be viewed as simply a special case of the Deutsch-Jozsa algorithm, only the latter term will be used henceforth except when the former is explicitly intended.

3.3 Quantum Network Simulation of the Generalized Deutsch-Jozsa Algorithm

For arbitrary qudits, the Deutsch-Jozsa algorithm can continue to be represented by the same schematic shown in Figure 2.3, except that now the range of possible inputs is $\{|0\rangle,|1\rangle,\ldots,|M-1\rangle\}$ where M is the logical order. To simulate this quantum system, I chose to represent the entire system as a matrix describing the overall tensor product of all elements in the quantum network at a particular stage in the computation. Beginning from a column vector representing the tensor product of all input states, these quantum data are acted upon by a series of successive transformations representing the initial series of Hadamard operations, followed by the application of the operator U_f and the final series of Hadamard transformations. Additional details and a worked out example of this process are included in Appendix 1. Source code for the quantum simulator is included in Appendix 3.

3.4 Mathematical Analysis of the Generalized Deutsch-Jozsa Algorithm

To complement the simulation of a quantum network implementing the generalized Deutsch-Jozsa algorithm, it is desirable to derive a mathematical expression from which one can obtain identical results for arbitrary qudits.

To begin with, it was necessary for me to examine the behaviour of the N simultaneous Hadamard operations $H^{\otimes N} = H \otimes H \otimes \ldots \otimes H$ in the context of arbitrary qudits. From (2.25), one can see that for qutrits, the action of the Hadamard gate on the states $|0\rangle$, $|1\rangle$ and $|2\rangle$, respectively, yields

$$H|0\rangle = \frac{1}{\sqrt{3}}\left(|0\rangle + |1\rangle + |2\rangle\right), \tag{3.1}$$

$$H|1\rangle = \frac{1}{\sqrt{3}}\left(|0\rangle + e^{\frac{i4\pi}{3}}|1\rangle + e^{\frac{i2\pi}{3}}|2\rangle\right),$$

$$H|2\rangle = \frac{1}{\sqrt{3}}\left(|0\rangle + e^{\frac{i2\pi}{3}}|1\rangle + e^{\frac{i4\pi}{3}}|2\rangle\right).$$

These transformations can be summarized by the expression

$$H|x\rangle = \frac{1}{\sqrt{3}}\sum_{z=0}^{2} e^{\frac{-i2\pi}{3}(xz)}|z\rangle. \tag{3.2}$$

One can easily confirm by using (2.25) that (3.2) generalizes to

$$H|x\rangle = \frac{1}{\sqrt{M}}\sum_{z=0}^{M-1} e^{\frac{-i2\pi}{M}(xz)}|z\rangle, \tag{3.3}$$

for higher-order qudits. The overall action of two Hadamard gates applied simultaneously (as in Figure 2.2) can be expressed as

$$H \otimes H|x\rangle = \frac{1}{M}\left(\sum_{z_1=0}^{M-1} e^{\frac{-i2\pi}{M}(x_1 z_1)}|z_1\rangle\right) \otimes \left(\sum_{z_2=0}^{M-1} e^{\frac{-i2\pi}{M}(x_2 z_2)}|z_2\rangle\right). \tag{3.4}$$

Upon working through the tensor product, this expression can be reduced to

$$H \otimes H|x\rangle = \frac{1}{M}\sum_{z_1,z_2=0}^{M^2-1} e^{\frac{-i2\pi}{M}(x_1 z_1 + x_2 z_2)}|z_1 z_2\rangle. \tag{3.5}$$

In general,

$$H^{\otimes N}|x\rangle = \frac{1}{\sqrt{M^N}}\sum_{z_1,z_2,\ldots,z_N=0}^{M^N-1} e^{\frac{-i2\pi}{M}(\bar{x}\cdot\bar{z})}|z_1 z_2 \ldots z_N\rangle, \tag{3.6}$$

where

$$\bar{x}\cdot\bar{z} = x_1 z_1 + x_2 z_2 + \ldots + x_N z_N. \tag{3.7}$$

To simplify the notation, I will write $|z_1 z_2 \ldots z_N\rangle = |\bar{z}\rangle$ and $z_1, z_2, \ldots, z_N = \bar{z}$ in subsequent equations.

Without in any way diminishing the utility of the Deutsch-Jozsa algorithm[23], the simplifying assumption that $|x\rangle$ shown in Figure 2.3 are set to $|0\rangle$ and $|y\rangle = |M-1\rangle$ will be made. Accordingly, (3.6) becomes

$$H^{\otimes N-1}|0\rangle \otimes H|M-1\rangle = \frac{1}{\sqrt{M^{N-1}}}\left(\sum_{\bar{z}=0}^{M^{N-1}-1}|\bar{z}\rangle\right)\otimes\left(\frac{1}{\sqrt{M}}\sum_{z'=0}^{M-1}e^{\frac{-i2\pi}{M}((M-1)z')}|z'\rangle\right). \tag{3.8}$$

The effect of the transformation U_f: $|x,y\rangle \rightarrow |x,y\oplus f(x)\rangle$ on (3.6) can now be derived. To begin with an example, assume that the function *f*(x) is defined as $f(0)=0; f(1)=0; f(2)=1$ for M = 3 and N = 2. Under these circumstances,

$$U_f\left(H|0\rangle\otimes H|2\rangle\right) = \frac{1}{3}\left[\left(|0\rangle+|1\rangle\right)\otimes|y'\rangle+|2\rangle\otimes\left(|1\rangle+e^{\frac{i2\pi}{3}}|2\rangle+e^{\frac{i4\pi}{3}}|0\rangle\right)\right], \tag{3.9}$$

where

$$|y'\rangle = \frac{1}{\sqrt{3}}\sum_{z=0}^{2}e^{\frac{-i2\pi}{3}(2z)}|z'\rangle. \tag{3.10}$$

Equation (3.9) can be simplified to

$$U_f\left(H|0\rangle\otimes H|2\rangle\right) = \frac{1}{3}\left[\left(|0\rangle+|1\rangle+e^{\frac{i4\pi}{3}}|2\rangle\right)\otimes|y'\rangle\right]. \tag{3.11}$$

Similarly, if $f(0)=0; f(1)=0; f(2)=2$, keeping M and N constant, then

$$U_f\left(H|0\rangle\otimes H|2\rangle\right) = \frac{1}{3}\left[\left(|0\rangle+|1\rangle+e^{\frac{i2\pi}{3}}|2\rangle\right)\otimes|y'\rangle\right]. \tag{3.12}$$

One can compute the effect that U_f has on the other combinations of *f*(0), *f*(1) and *f*(2) to arrive at the more general expression

[23] This statement will justified in Section 3.5.

$$U_f\left(H|0\rangle \otimes H|2\rangle\right) = \frac{1}{3}\left[\left(e^{\frac{-i2\pi}{3}f(0)}|0\rangle + e^{\frac{-i2\pi}{3}f(1)}|1\rangle + e^{\frac{-i2\pi}{3}f(2)}|2\rangle\right) \otimes |y'\rangle\right]. \qquad (3.13)$$

Generalized to arbitrary numbers of qudits, (3.13) can be expressed in the more general form

(3.14)

$$U_f\left(H|0\rangle \otimes H|M-1\rangle\right) = \frac{1}{\sqrt{M^{N-1}}}\left(\sum_{x=0}^{M^{N-1}-1} e^{\frac{-i2\pi}{M}f(x)}|x\rangle\right) \otimes \left(\frac{1}{\sqrt{M}}\sum_{z=0}^{M-1} e^{\frac{-i2\pi}{M}((M-1)z)}|z\rangle\right).$$

No further transformations will be applied to $|y'\rangle$. Moreover, only the output "lines" corresponding to the $|x\rangle$ inputs will be measured. Therefore, $|y'\rangle$ will not be considered further in this discussion in order to direct our focus to the final sequence of Hadamard transformations leading to $|\psi\rangle$, the state upon a measurement will be taken, as depicted in Figure 3.1. From (3.6), these transformations produce the overall effect

$$H^{\otimes N-1}|x\rangle = \frac{1}{\sqrt{M^{N-1}}}\sum_{\bar{z}=0}^{M^{N-1}-1} e^{\frac{-i2\pi}{M}(\bar{x}\cdot\bar{z})}|\bar{z}\rangle, \qquad (3.15)$$

resulting in

$$|\psi\rangle = \frac{1}{M^{N-1}}\sum_{\bar{z}=0}^{M^{N-1}-1}\sum_{\bar{x}=0}^{M^{N-1}-1} e^{\frac{-i2\pi}{M}(\bar{x}\cdot\bar{z}+f(x))}|\bar{z}\rangle, \qquad (3.16)$$

where

$$\bar{x}\cdot\bar{z} = x_0z_0 + x_1z_1 + \ldots + x_{N-1}z_{N-1}, \qquad (3.17)$$

and M is the logical order and N gives the number of inputs into the circuit implementing the Deutsch-Jozsa algorithm. The state $|\psi\rangle$ is directly measured, as shown in Figure 3.1.

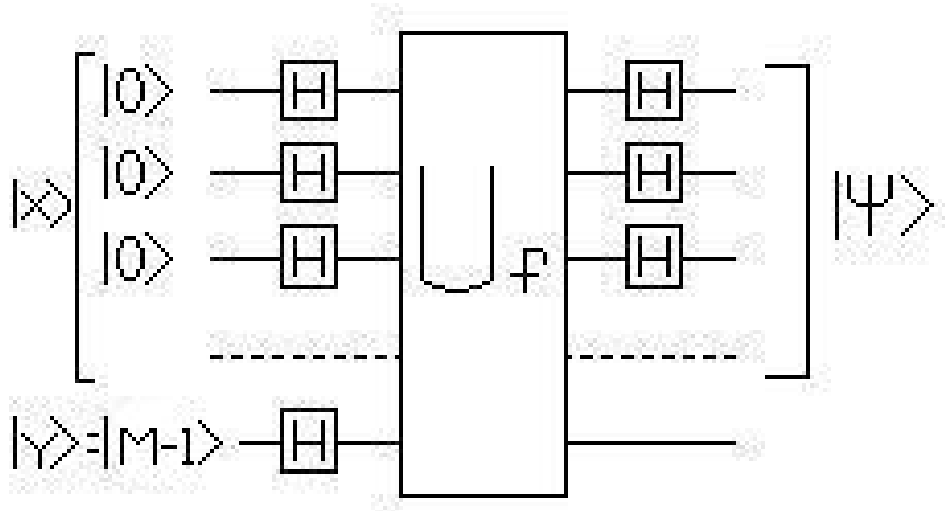

Figure 3.1: A schematic diagram of a quantum network that implements the generalized Deutsch-Jozsa algorithm. The N – 1 inputs collectively denoted $|x\rangle$ are all set to $|0\rangle$ and the final input $|y\rangle$ is set to $|M - 1\rangle$. The final state $|\psi\rangle$ represents the state of a subset of the system immediately prior to its being measured. This network implements (3.16).

As a convenient way to compute (3.16) for multiple combinations of M, N and *f*(x), I wrote a special function, DeutschEquation, the source code for which is in Appendix 3. The results obtained thereby were identical with those computed using the simulation method.

3.5 Results

As a first test of both the simulator and the mathematical method, I confirmed that these two methods yield the same, well-documented, results for the qubit version of the

Deutsch-Jozsa algorithm. For $|x\rangle = |0\rangle$ and $|y\rangle = |1\rangle$, these results are given in Appendix 2 for both N = 2 and N = 3. As is well known, if the function *f*(x) is promised to be either constant or balanced, a measurement of $|\psi\rangle$ will, with probability one, yield $|0\rangle$ if N = 2, $|00\rangle$ if N = 3 and so on. On the other hand, each mapping of *f*(x) that yields a balanced condition will, with certainty, yield a specific measurement outcome other than what is obtained in the constant case for the same N. If the function *f*(x) is neither constant nor balanced, $|\psi\rangle$ becomes a superposition of possible outcomes, hence one cannot obtain any particular outcome with certainty.

I could not locate any discussion of the Deutsch-Jozsa algorithm in the literature for M > 2; therefore, I used my simulator and mathematical functions to confirm each other's output, coupled with extensive hand calculations. The results I obtained all mutually agreed and are presented in Appendix 2 for M = 3 (qutrits), N = 2, $|x\rangle = |0\rangle$ and $|y\rangle = |2\rangle$ in their entirety. For the same values of M and N, selected results are presented for $|x\rangle = |0\rangle$, $|y\rangle = |1\rangle$ and $|x\rangle = |1\rangle$, $|y\rangle = |2\rangle$. Likewise, I have presented those results for M = 4 (ququads) and 5 (ququints) with N = 2 and $|x\rangle = |0\rangle$, $|y\rangle = |M - 1\rangle$ that will help develop the interpretations to be discussed below. I also did much experimentation with N > 2 for these values of M, but I have not included most of these results in Appendix 2. In the following discussion, it will be assumed that $|x\rangle = |0\rangle$ for N = 2, $|x\rangle = |00\rangle$ for N = 3, etc. and $|y\rangle = |M - 1\rangle$.

For all values of M and N, if f(x) is constant,

$$|\psi\rangle = e^{-\frac{i2\pi}{M}f(0)}|0\rangle, \tag{3.18}$$

and a measurement will clearly yield $|0\rangle$. In contrast, the balanced case of the qubit Deutsch-Jozsa algorithm does not generalize as easily. Evidently, if M is odd, there cannot be a balanced case. However, for M even, balanced combinations of f(x) yield definite outcomes only in certain instances. For example, a Deutsch-Jozsa circuit with N = 2 and M = 4 yields the definite outcome $|2\rangle$ only in the following four cases:

$$\begin{aligned}
&f(0) = 0; \quad f(1) = 2; \quad f(2) = 0; \quad f(3) = 2, \\
&f(0) = 2; \quad f(1) = 0; \quad f(2) = 2; \quad f(3) = 0, \\
&f(0) = 1; \quad f(1) = 3; \quad f(2) = 1; \quad f(3) = 3, \\
&f(0) = 3; \quad f(1) = 1; \quad f(2) = 3; \quad f(3) = 1.
\end{aligned} \tag{3.19}$$

All other balanced combinations of f(x) yield a superposition of potential measurement outcomes.

For N = 2, a more generally useful result is obtained if $f(0) = 0, f(1) = 1, \ldots, f$(M – 1) = M – 1. In this instance, a measurement of $|\psi\rangle$ will yield $|M-1\rangle$ with certainty. This also holds true for any permutation of this mapping of f(x) within the schema

$$\begin{aligned}
&f(0) = 1; \quad f(1) = 2; \quad \ldots \quad f(M-2) = M-1; \quad f(M-1) = 0, \\
&f(0) = 2; \quad f(1) = 3; \quad \ldots \quad f(M-2) = 0; \quad f(M-1) = 1, \\
&\quad \vdots \\
&f(0) = M-1; \quad f(1) = 0; \quad \ldots \quad f(M-2) = M-3; \quad f(M-1) = M-2.
\end{aligned} \tag{3.20}$$

For the reverse mapping, i.e. $f(0) =$ M - 1, $f(1) =$ M - 2, …, f(M – 2) = 1, f(M – 1) = 0 and its permutations

$$\begin{aligned} &f(0) = M - 2; \quad f(1) = M - 3; \quad \ldots \quad f(M-2) = 0 \quad f(M-1) = M - 1, \\ &f(0) = M - 3; \quad f(1) = M - 4 \quad \ldots \quad f(M-2) = M - 1; \quad f(M-1) = 0, \\ &\quad \vdots \\ &f(0) = 0; \quad f(1) = M - 1; \quad \ldots \quad f(M-2) = 2; \quad f(M-1) = 1, \end{aligned} \tag{3.21}$$

the measurement result $|1\rangle$ is yielded with certainty. Other combinations involving a one-to-one mapping of *f*(x) do not yield any definite outcome with probability one.

For N > 2, the considerations stated above concerning *f*(x) defined as a constant function apply without modification. The special balanced cases mentioned above also generalize. For example, if M = 4 and

$$f(x_1x_2x_3\ldots x_{N-1}) = \begin{cases} 0, x_{N-1}\ even \\ 2, x_{N-1}\ odd \end{cases}, \tag{3.22}$$

a measurement of $|\psi\rangle$ will yield $|02\rangle$ with certainty. Precisely the same result is obtained for the following three mappings of function *f*(x), again with M = 4.

$$f(x_1x_2x_3\ldots x_{N-1}) = \begin{cases} 2, x_{N-1}\ even \\ 0, x_{N-1}\ odd \end{cases}, \tag{3.23}$$

$$f(x_1x_2x_3\ldots x_{N-1}) = \begin{cases} 1, x_{N-1}\ even \\ 3, x_{N-1}\ odd \end{cases},$$

$$f(x_1x_2x_3\ldots x_{N-1}) = \begin{cases} 3, x_{N-1}\ even \\ 1, x_{N-1}\ odd \end{cases}.$$

In the above statements, if x_{N-1} is replaced by x_i, where $1 \le i \le N-1$, as the determiner of the value of *f*(x), a definite measurement result will again be obtained, but it may not be $|02\rangle$.

One can also generalize the statement made concerning the special one-to-one mappings of *f*(x) for N = 2 to arbitrary N. Given $f(x_1x_2x_3\ldots x_{N-1})$, if as any x_i varies

consecutively from 0 to M – 1, where $i = \{0,1,\ldots,N-1\}$, the function itself varies in exact correspondence to that x_i, then exactly one measurement outcome is guaranteed to occur with certainty. In other words,

$$f(x_1x_2x_3\ldots x_{N-1}) = x_i, \quad 0 \le x_i \le M-1 \text{ and } 1 \le i \le N-1. \tag{3.24}$$

Equation (3.24) can be further generalized according to the permutations described by (3.20) and (3.21).

Thus far, I have assumed that $|x_1x_2x_3\ldots x_{N-1}\rangle = |000...0\rangle$ and $|y\rangle = |M-1\rangle$. If these parameters are adjusted, no new potential computational resources are made available. For $x_1 = x_2 = x_3 = \ldots = x_{N-1} = y$, the same $|\psi\rangle$ is obtained for all mappings of f(x) making this an undesirable combination of inputs. I have included selected results in Appendix 2 for M = 3 (qutrits) and N = 2 with the input combinations $|x\rangle = |0\rangle$; $|y\rangle = |1\rangle$ and $|x\rangle = |1\rangle$; $|y\rangle = |2\rangle$. To summarize, the set of all outcomes previously obtained with certainty remains the same, except that the specific measurement outcome obtained may differ. No mappings of f(x) which did not previously yield a definite outcome do so with these alternative inputs. Hence, the assumption that $|x_1x_2x_3\ldots x_{N-1}\rangle = |000\ldots0\rangle$ and $|y\rangle = |M-1\rangle$ is justifiable in the sense that keeping these parameters constant will not cause the neglect of any potentially important results.

Chapter 4: Conclusion

The results of my experimentation with the Deutsch-Jozsa algorithm laid out in Chapter 3 clearly yield promising results for the future development of quantum computational science. The direct generalization of the behaviour of the constant case in the qubit version of this algorithm to higher-order logic certainly bodes well for the utility of this algorithm. Moreover, the ability of the generalized algorithm to yield a definitive, measurable result in certain instances (for N = 2) where $f(x)$ is one-to-one implies that, for specific problems, one may be able to devise a test that will yield, with a single set of measurements, the answer to a question concerning whether $f(x)$ is a one-to-one function or not. Some of the other mappings of $f(x)$ that yield a single possible measurement outcome may have a less obvious practical utilization; however, any classes of problems that can be formulated in terms of these mappings of $f(x)$ will certainly be able to be solved much more efficiently given that only N – 1 measurements need be made after a single run of the Deutsch-Jozsa algorithm has occurred to arrive at the desired solution. Hence, the computational complexity of this algorithm is O(N-1), which is a dramatic improvement over the classical computational complexity of $O(M^N)$. Even for relatively small values of M and N for which a classical solution can be obtained without a great deal of computational resources being required, the quantum approach may be viable if a large number of repeated function evaluations are necessary.

The challenge, of course, for anyone who desires to use this approach to compute some problem of practical or theoretical interest is to relate that problem to some mapping of $f(x)$ that yields a definite outcome upon measurement. There is much room

for creative flexibility here; however, one especially promising such mapping involves the periodic definition of f(x) described by (3.24). Such a mapping may prove congenial to those problems involving the discrete sampling of a periodic function, such as the result of a Fourier transform. To give one simple example using the result described by (3.24), suppose that one has an unknown function f(x) within the context of N ququads (M = 4). With only N – 1 measurements, one can determine whether that function satisfies

$$f(x_1 x_2 \ldots x_{N-1}) = x_{N-1}, \quad 0 \leq x_{N-1} \leq 3 . \tag{4.1}$$

Hence, one can easily determine if this function possesses this form of periodicity. Classically, one may need to evaluate the function using every possible input to make this same determination. Such a procedure would require up to 4^{N-1} function evaluations.

Clearly the methodologies presented in Chapter 3 do not exhaust the range of possible avenues of exploration concerning this topic. One needs only to think of the questions that must be resolved prior to actual experiments implementing the generalized Deutsch-Jozsa algorithm can take place and what the results described above may imply within the context of other quantum algorithms to obtain a glimpse of the vast array of issues that still remain for consideration.

Perhaps the most logical extension of this work is an examination of the behaviour of the Deutsch-Jozsa algorithm in the presence of non-ideal conditions. Despite the assumptions of ideal operating conditions as explained in Chapter 3, all attempts to actualize quantum computational hardware must confront the challenges posed by non-ideal hardware and the ever present possibility of the surrounding environment introducing unwanted perturbations and entanglements into the system.

This fact has long been recognized and has led to much discussion of quantum error correction procedures and fault-tolerant quantum computation. Prior to any attempt to translate the generalized Deutsch-Jozsa algorithm into an experimental reality, it would be wise to examine how one could ensure that the quantum network implementing the algorithm is robust against some common forms of noise.

Furthermore, a fairly direct extension of this discussion of the generalized Deutsch-Jozsa algorithm concerns the behaviour of related quantum algorithms in the context of higher-order logic. A few such quantum "oracle" algorithms[24] are relatively similar in structure to the Deutsch-Jozsa algorithm and could be generalized rather easily to higher-order logic. Although less similar to the Deutsch-Jozsa algorithm, Shor's factoring algorithm or Grover's search algorithm are of great practical importance and as such, should be investigated within the context of higher-order logic.

Whether or not the extension of the Deutsch-Josza algorithm to higher-order bit states ultimately finds practical application is not the only consideration relevant to the ultimate worth of this proposal. Even if practical quantum computers in the future, like their classical counterparts, generally do not use higher-order bit types, the very investigation of these possibilities will help define the boundaries of the discipline as it continues to emerge and help ensure that no potential breakthroughs are overlooked and that, of all the possible avenues of future development, only the most promising are actually followed.

[24] In this context, an "oracle" is a quantum algorithm in which a single measurement or set of measurements is made in order to determine the results of a "black-box" operation with the goal of answering a specific question. The Deutsch-Jozsa algorithm is one example.

Appendix 1 – Deutsch-Jozsa Quantum Network Simulator

As noted in Chapter 3, I implemented a quantum network level simulation of the Deutsch-Jozsa algorithm using the programming language GNU Octave, version 2.1.42. As the version I obtained does not come bundled with any function libraries pertaining to quantum computation, I had to write some subsidiary functions to support the Deutsch-Jozsa simulation.

The first such subsidiary function receives the logical order M as input and returns a matrix representing the corresponding Hadamard gate (see Equation 2.25). The second subsidiary function receives the logical order M as input and returns the sum, modulo M, of its other two inputs (for M = 2, this is equivalent to the exclusive-OR operation). However, the majority of the computation necessary to simulate the Deutsch-Jozsa algorithm occurs within a single function, called DeutschJozsa. This function does not receive any external inputs; instead, the logical order M, the quantum network inputs and the definition of $f(x)$ are all adjustable parameters within the function itself. A printout of the source code of this function has been included in Appendix 3.

The operation of the Deutsch-Jozsa simulator may be best illustrated through an example. Assume that the logical order M = 3 (qutrits) and that the quantum network is fed by two input lines, corresponding to the situation depicted in Figure 3.1. Assume further that the function $f(x)$ is defined by the mapping

$$\begin{aligned} f(0) &= 2 \\ f(1) &= 0 \\ f(2) &= 1 \end{aligned} . \qquad \text{(A1.1)}$$

The overall input state may be represented by the tensor product

$$|0\rangle \otimes |2\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{A1.2}$$

Each element in this 9 element column matrix can be uniquely identified by an ordered pair $|x, y\rangle$. In this instance $|0,2\rangle = 1$; all other elements are 0. A Hadamard operation acts upon each input as depicted in Figure 3.1, resulting in the overall state

$$H|0\rangle \otimes H|2\rangle = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & \sigma^2 & \sigma \\ 1 & \sigma & \sigma^2 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 & 1 \\ 1 & \sigma^2 & \sigma \\ 1 & \sigma & \sigma^2 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 1 \\ \sigma \\ \sigma^2 \\ 1 \\ \sigma \\ \sigma^2 \\ 1 \\ \sigma \\ \sigma^2 \end{pmatrix}. \tag{A1.3}$$

Here, the Hadamard gate for qutrits (Equation 2.25) has been used and $\sigma = e^{\frac{i2\pi}{3}}$. Given that U_f performs the transformation $|x, y\rangle \rightarrow |x, y \oplus f(x)\rangle$, the following possible mappings are realizable given the definition of *f*(x):

x	y	f(x)	$y \oplus f(x)$
0	0	2	2
0	1	2	0
0	2	2	1
1	0	0	0
1	1	0	1
1	2	0	2
2	0	1	1
2	1	1	2
2	2	1	0

Table A1.1: Tri-state logic table for $y \oplus f(x)$.

Therefore the transformation U_f effectively rearranges the state $H|0\rangle \otimes H|2\rangle$ giving the result

$$U_f\left(H|0\rangle \otimes H|2\rangle\right) = \frac{1}{3}\begin{pmatrix} \sigma \\ \sigma^2 \\ 1 \\ 1 \\ \sigma \\ \sigma^2 \\ \sigma^2 \\ 1 \\ \sigma \end{pmatrix} = \frac{1}{\sqrt{3}}\begin{pmatrix} \sigma \\ 1 \\ \sigma^2 \end{pmatrix} \otimes \frac{1}{\sqrt{3}}\begin{pmatrix} 1 \\ \sigma \\ \sigma^2 \end{pmatrix} = |x'\rangle \otimes |y'\rangle . \qquad \text{(A1.4)}$$

Finally, a single Hadamard operation is applied to $|x'\rangle$, giving the result

$$|\psi\rangle = H|x'\rangle = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & \sigma^2 & \sigma \\ 1 & \sigma & \sigma^2 \end{pmatrix}\begin{pmatrix} \sigma \\ 1 \\ \sigma^2 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} \sigma + 1 + \sigma^2 \\ \sigma + \sigma^2 + 1 \\ \sigma + \sigma + \sigma \end{pmatrix} = \sigma\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = e^{\frac{i2\pi}{3}}|2\rangle . \qquad \text{(A1.5)}$$

A measurement of $|\psi\rangle$ will yield $|2\rangle$ with probability one (the global phase factor $e^{\frac{i2\pi}{3}}$ has no effect on the measurement outcome).

Appendix 2 – Selected Results for the Deutsch-Jozsa Algorithm Generalized to Arbitrary Qudits

A tabulation of selected results from the Deutsch-Jozsa algorithm are collated in this Appendix for various function mappings and values of the parameters N (number of input/output "lines") and M (logical order). Each result corresponds to the state $|\psi\rangle$ as it is defined in Figure 3.1. For the interpretation of these results, please consult Section 3.5 in this thesis.

M = 2 (qubits); N = 2 $\|x\rangle = \|0\rangle \quad \|y\rangle = \|1\rangle$		$\|\psi\rangle$
***f*(0)**	***f*(1)**	
0	0	$\|0\rangle$
0	1	$\|1\rangle$
1	0	$-\|1\rangle$
1	1	$-\|0\rangle$

M = 2 (qubits); N = 3 $\|x\rangle = \|00\rangle \quad \|y\rangle = \|1\rangle$				$\|\psi\rangle$
***f*(00)**	***f*(01)**	***f*(10)**	***f*(11)**	
0	0	0	0	$\|00\rangle$
0	0	0	1	$0.5\left(\|00\rangle + \|01\rangle + \|10\rangle - \|11\rangle\right)$
0	0	1	0	$0.5\left(\|00\rangle - \|01\rangle + \|10\rangle + \|11\rangle\right)$
0	0	1	1	$\|10\rangle$
0	1	0	0	$0.5\left(\|00\rangle + \|01\rangle - \|10\rangle + \|11\rangle\right)$
0	1	0	1	$\|01\rangle$
0	1	1	0	$\|11\rangle$
0	1	1	1	$0.5\left(-\|00\rangle + \|01\rangle + \|10\rangle + \|11\rangle\right)$
1	0	0	0	$0.5\left(\|00\rangle - \|01\rangle - \|10\rangle - \|11\rangle\right)$
1	0	0	1	$-\|11\rangle$
1	0	1	0	$-\|01\rangle$
1	0	1	1	$0.5\left(-\|00\rangle - \|01\rangle + \|10\rangle - \|11\rangle\right)$
1	1	0	0	$-\|10\rangle$
1	1	0	1	$0.5\left(-\|00\rangle + \|01\rangle - \|10\rangle - \|11\rangle\right)$
1	1	1	0	$0.5\left(-\|00\rangle - \|01\rangle - \|10\rangle + \|11\rangle\right)$
1	1	1	1	$-\|00\rangle$

M = 3 (qutrits); N = 2 $\|x\rangle = \|0\rangle \quad \|y\rangle = \|2\rangle$			$\|\psi\rangle$
***f*(0)**	***f*(1)**	***f*(2)**	
0	0	0	$\|0\rangle$
0	0	1	$\frac{1}{3}\left[\left(2+e^{\frac{i4\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|1\rangle+\left(1+2e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
0	0	2	$\frac{1}{3}\left[\left(2+e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i4\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
0	1	0	$\frac{1}{3}\left[\left(2+e^{\frac{i4\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
0	1	1	$\frac{1}{3}\left[\left(1+2e^{\frac{i4\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
0	1	2	$\|2\rangle$
0	2	0	$\frac{1}{3}\left[\left(2+e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(1+2e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
0	2	1	$\|1\rangle$
0	2	2	$\frac{1}{3}\left[\left(1+2e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
1	0	0	$\frac{1}{3}\left[\left(2+e^{\frac{i4\pi}{3}}\right)\|0\rangle+e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|1\rangle+e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
1	0	1	$\frac{1}{3}\left[\left(1+2e^{\frac{i4\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i4\pi}{3}}\right)\|1\rangle+e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
1	0	2	$e^{\frac{i4\pi}{3}}\|1\rangle$
1	1	0	$\frac{1}{3}\left[\left(1+2e^{\frac{i4\pi}{3}}\right)\|0\rangle+e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(1+2e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
1	1	1	$e^{\frac{i4\pi}{3}}\|0\rangle$
1	1	2	$\frac{1}{3}\left[e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|0\rangle+e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i4\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
1	2	0	$e^{\frac{i4\pi}{3}}\|2\rangle$

1	2	1	$\frac{1}{3}\left[e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|1\rangle+e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
1	2	2	$\frac{1}{3}\left[e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i4\pi}{3}}\right)\|1\rangle+\left(1+2e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
2	0	0	$\frac{1}{3}\left[\left(2+e^{\frac{i2\pi}{3}}\right)\|0\rangle+e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|1\rangle+e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
2	0	1	$e^{\frac{i2\pi}{3}}\|2\rangle$
2	0	2	$\frac{1}{3}\left[\left(1+2e^{\frac{i2\pi}{3}}\right)\|0\rangle+e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(1+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
2	1	0	$e^{\frac{i2\pi}{3}}\|1\rangle$
2	1	1	$\frac{1}{3}\left[e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(1+2e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
2	1	2	$\frac{1}{3}\left[e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|0\rangle+e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
2	2	0	$\frac{1}{3}\left[\left(1+2e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i2\pi}{3}}\right)\|1\rangle+e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
2	2	1	$\frac{1}{3}\left[e^{\frac{i2\pi}{3}}\left(2+e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|1\rangle+e^{\frac{i2\pi}{3}}\left(1+2e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
2	2	2	$e^{\frac{i2\pi}{3}}\|0\rangle$

M = 3 (qutrits); N = 2 $\|x\rangle = \|1\rangle \quad \|y\rangle = \|2\rangle$			$\|\psi\rangle$
***f*(0)**	***f*(1)**	***f*(2)**	
0	0	0	$\|2\rangle$
0	0	1	$\frac{1}{3}\left[\left(2+e^{\frac{i4\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
0	0	2	$\frac{1}{3}\left[\left(1+2e^{\frac{i4\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
0	1	0	$\frac{1}{3}\left[\left(1+2e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i4\pi}{3}}\right)\|2\rangle\right]$
0	1	2	$\|1\rangle$
0	2	1	$\|0\rangle$
1	0	2	$e^{\frac{i4\pi}{3}}\|0\rangle$
1	1	1	$e^{\frac{i4\pi}{3}}\|2\rangle$
1	2	0	$e^{\frac{i4\pi}{3}}\|1\rangle$
2	0	1	$e^{\frac{i2\pi}{3}}\|1\rangle$
2	1	0	$e^{\frac{i2\pi}{3}}\|0\rangle$
2	2	2	$e^{\frac{i2\pi}{3}}\|2\rangle$

M = 3 (qutrits); N = 2 $\|x\rangle = \|0\rangle \quad \|y\rangle = \|1\rangle$			$\|\psi\rangle$
***f*(0)**	***f*(1)**	***f*(2)**	
0	0	0	$\|0\rangle$
0	0	1	$\frac{1}{3}\left[\left(2+e^{\frac{i2\pi}{3}}\right)\|0\rangle+\left(1+2e^{\frac{i4\pi}{3}}\right)\|1\rangle+\left(2+e^{\frac{i2\pi}{3}}\right)\|2\rangle\right]$
0	1	2	$\|1\rangle$
0	2	1	$\|2\rangle$
1	0	2	$e^{\frac{i2\pi}{3}}\|2\rangle$
1	1	1	$e^{\frac{i2\pi}{3}}\|0\rangle$
1	2	0	$e^{\frac{i2\pi}{3}}\|1\rangle$
2	0	1	$e^{\frac{i4\pi}{3}}\|1\rangle$
2	1	0	$e^{\frac{i4\pi}{3}}\|2\rangle$
2	2	2	$e^{\frac{i4\pi}{3}}\|0\rangle$

M = 4 (ququads); N = 2 $\|x\rangle = \|0\rangle \quad \|y\rangle = \|3\rangle$				$\|\psi\rangle$
***f*(0)**	***f*(1)**	***f*(2)**	***f*(3)**	
0	0	0	0	$\|0\rangle$
1	1	1	1	$-i\|0\rangle$
2	2	2	2	$-\|0\rangle$
3	3	3	3	$i\|0\rangle$
0	0	1	1	$\frac{1}{2}\left((1-i)\|0\rangle + \|1\rangle\right)$
0	0	2	2	$\frac{1}{2}\left((1-i)\|1\rangle + (1+i)\|3\rangle\right)$
0	1	0	1	$\frac{1}{2}\left((1-i)\|0\rangle + (1+i)\|2\rangle\right)$
0	1	1	0	$\frac{1}{2}\left((1-i)\|0\rangle + i\|1\rangle + \|3\rangle\right)$
0	1	2	3	$\|3\rangle$
0	1	3	2	$\frac{1}{2}\left(-i\|1\rangle + (1+i)\|2\rangle + \|3\rangle\right)$
0	2	0	2	$\|2\rangle$
0	2	1	3	$\frac{1}{2}\left(i\|1\rangle + (1-i)\|2\rangle + \|3\rangle\right)$
0	2	2	0	$\frac{1}{2}\left((1+i)\|1\rangle + (1-i)\|3\rangle\right)$
0	2	3	1	$\frac{1}{2}\left(\|1\rangle + (1+i)\|2\rangle - i\|3\rangle\right)$
0	3	0	3	$\frac{1}{2}\left((1+i)\|0\rangle + (1-i)\|2\rangle\right)$
0	3	1	2	$\frac{1}{2}\left(\|1\rangle + (1-i)\|2\rangle + i\|3\rangle\right)$
0	3	2	1	$\|1\rangle$
1	0	1	0	$\frac{1}{2}\left((1-i)\|0\rangle - (1+i)\|2\rangle\right)$
1	0	3	2	$\|1\rangle$
1	2	3	0	$\|3\rangle$
1	3	1	3	$\|2\rangle$

2	0	2	0	$\|2\rangle$
2	1	0	3	$\|1\rangle$
2	3	0	1	$\|3\rangle$
3	0	1	2	$\|3\rangle$
3	1	3	1	$\|2\rangle$
3	2	1	0	$\|1\rangle$

M = 5 (ququints) ; N = 2 $\|x\rangle = \|0\rangle \quad \|y\rangle = \|4\rangle$					$\|\psi\rangle$
***f*(0)**	***f*(1)**	***f*(2)**	***f*(3)**	***f*(4)**	
0	0	0	0	0	$\|0\rangle$
1	1	1	1	1	$e^{-\frac{i2\pi}{5}}\|0\rangle$
2	2	2	2	2	$e^{-\frac{i4\pi}{5}}\|0\rangle$
3	3	3	3	3	$e^{-\frac{i6\pi}{5}}\|0\rangle$
4	4	4	4	4	$e^{-\frac{i8\pi}{5}}\|0\rangle$
0	1	2	3	4	$\|4\rangle$
1	2	3	4	0	$e^{-\frac{i2\pi}{5}}\|4\rangle$
2	3	4	0	1	$e^{-\frac{i4\pi}{5}}\|4\rangle$
3	4	0	1	2	$e^{-\frac{i6\pi}{5}}\|4\rangle$
4	0	1	2	3	$e^{-\frac{i8\pi}{5}}\|4\rangle$
4	3	2	1	0	$e^{-\frac{i8\pi}{5}}\|1\rangle$
3	2	1	0	4	$e^{-\frac{i6\pi}{5}}\|1\rangle$
2	1	0	4	3	$e^{-\frac{i4\pi}{5}}\|1\rangle$
1	0	4	3	2	$e^{-\frac{i2\pi}{5}}\|1\rangle$
0	4	3	2	1	$\|1\rangle$

Appendix 3: Source Code

(1) Hadamard Gate Generation Function

Given input logical_type = M, this function constructs and returns a matrix representing the corresponding Hadamard gate for that value of M according to (2.25).

```
function generalized_Hadamard_gate = Hadamard(logical_type)

  sigma = exp(i*2*pi / logical_type);

  generalized_Hadamard_gate = [];
  for i=1:logical_type
    for j=1:logical_type
      if (i == 1)
        generalized_Hadamard_gate(i,j) = 1;
      else
        generalized_Hadamard_gate(i,j) = sigma^((logical_type - i + 1) *
(j - 1));
      endif
    endfor
  endfor

endfunction
```

(2) Generalized XOR Function

Computes $y \oplus f(x)$ for input values of y and *f*(x).

```
## Performs the function y XOR f(x) for some given inputs y, f(x) and
## specified logical type (e.g. qubits, qutrits, ququads).

function generalized_XOR_output = generalized_XOR(logical_type, y, fx)

  generalized_XOR_output = y + fx;
  if (generalized_XOR_output >= logical_type)
    generalized_XOR_output = generalized_XOR_output - logical_type;
  endif

endfunction
```

(3) Deutsch-Jozsa Simulator

This function simulates the operation of a circuit implementing the Deutsch-Jozsa algorithm (Figure 3.1) using the input parameters specified within the function body.

```
function DeutschJozsa_output = DeutschJozsa

  ## Logical Order M: Set to '2' for qubits, '3' for qutrits, etc.

  logical_type = 3;

  ## Describe the input.  Each successive term to the right describes
  ## the input along a single 'bit' line, going from top to bottom as
  ## seen in a schematic in which the input is on the left side and the
  ## output on the right.  The permissible range of values is {0,1,...M
  ## - 1}, where M is the logical order.

  circuit_input = [0 0 2];

  ## Describes the mapping of f(x).  In the example presented below, the
  ## first value in the function matrix corresponds to f(00) = 0.
  ## Subsequent values are f(01) = 0, f(02) = 0, f(10) = 1,…, f(22) = 2.

  fx = [0 0 0 1 1 1 2 2 2];

  ## Convert the variable circuit_input into a column matrix.

  if (size(circuit_input)(1) == 1)
    circuit_input = circuit_input';
  endif

  circuit_input_matrix = [];
  for i=1:size(circuit_input)(1)
    circuit_input_column = [];
    for j=1:logical_type
      if (circuit_input(i) == j - 1)
        circuit_input_column = [circuit_input_column;1];
      else
        circuit_input_column = [circuit_input_column;0];
      endif
    endfor
    if (i == 1)
      circuit_input_matrix = circuit_input_column;
    else
      circuit_input_matrix(:,i) = circuit_input_column;
    endif
  endfor

  ## Apply Hadamard gate to each input line and take tensor product of
  ## resultant.

  if (size(circuit_input)(1) == 1)
    input_to_unitary_function = Hadamard(logical_type) *
circuit_input_matrix;
```

```
  elseif (size(circuit_input)(1) == 2)
    input_to_unitary_function = tensor_multiply(Hadamard(logical_type) *
circuit_input_matrix(:,1), Hadamard(logical_type) *
circuit_input_matrix(:,2));
  elseif (size(circuit_input)(1) > 2)
    input_to_unitary_function = tensor_multiply(Hadamard(logical_type) *
circuit_input_matrix(:,1), Hadamard(logical_type) *
circuit_input_matrix(:,2));
    for i=3:size(circuit_input)(1)
      input_to_unitary_function =
tensor_multiply(input_to_unitary_function, Hadamard(logical_type) *
circuit_input_matrix(:,i));
    endfor
  endif

  for i=1:size(input_to_unitary_function)(1)
    unitary_function_output(i) = 0;
  endfor

  ## Compute the transformation |x, y> --> |x, y XOR f(x)>

  counter = 1;
  left_ket = 0;
  for i=1:size(input_to_unitary_function)(1)
    y_XOR_fx = generalized_XOR(logical_type, counter - 1, fx(left_ket +
1));
    unitary_function_output(left_ket * logical_type + y_XOR_fx + 1) =
unitary_function_output(left_ket * logical_type + y_XOR_fx + 1) +
input_to_unitary_function(i);
    if (counter == logical_type)
      counter = 1;
      left_ket++;
    else
      counter++;
    endif
  endfor

  ## The "transpose" operator ' in GNU Octave actually returns the
  ## Hermitian adjoint (matrix transpose + complex conjugate).  To get
  ## around this to perform a simple matrix transpose, I needed to use
  ## the conj (complex conjugate) function after using GNU Octave's
  ## "transpose" operator.

  unitary_function_output = conj(unitary_function_output');

  ## Apply Hadamard gate to each output line (except |y'>) and take
  ## tensor product of resultant.

  if (size(circuit_input)(1) == 2)
    Deutsch_output_operator = Hadamard(logical_type);
  elseif (size(circuit_input)(1) == 3)
    Deutsch_output_operator = tensor_multiply(Hadamard(logical_type),
Hadamard(logical_type));
  elseif (size(circuit_input)(1) > 3)
    Deutsch_output_operator = tensor_multiply(Hadamard(logical_type),
Hadamard(logical_type));
    for i=4:size(circuit_input)(1)
```

```
      Deutsch_output_operator = tensor_multiply(Deutsch_output_operator,
Hadamard(logical_type));
    endfor
  endif

  ## Finishes computing the final system matrix representing the whole
  ## system just before measuring |x'>.

  DeutschJozsa_output = tensor_multiply(Deutsch_output_operator,
no_op(logical_type)) * unitary_function_output /
logical_type^(size(circuit_input)(1) - 1);

  ## Determines the outcome probabilities for each |x'> output.

  for i=1:logical_type
    probability_list(i,1) = round(i) - 1;
    probability_list(i,2) = abs(DeutschJozsa_output((i - 1) *
logical_type + 1)) ^ 2;
  endfor

endfunction
```

(4) Deutsch-Jozsa Equation

The following function computes (3.16) for the parameters set within the function body.

```
function closed_form_output = DeutschEquation

  ## Logical Order M: Set to '2' for qubits, '3' for qutrits, etc.

  logical_type = 4;

  ## Describe the input.  Each successive term to the right describes
  ## the input along a single 'bit' line, going from top to bottom as
  ## seen in a schematic in which the input is on the left side and the
  ## output on the right.  The permissible range of values is {0,1,...M
  ## - 1}, where M is the logical order.

  circuit_input = [0 3];

  ## Describes the mapping of f(x).  In the example presented below, the
  ## first value in the function matrix corresponds to f(0) = 0.
  ## Subsequent values are f(1) = 1, f(2) = 2 and f(3) = 3.

  fx = [0 1 2 3];

  ## Convert the variable circuit_input into a column matrix.

  if (size(circuit_input)(1) == 1)
    circuit_input = circuit_input';
  endif

  circuit_input_matrix = [];
```

```
  for i=1:size(circuit_input)(1)
    circuit_input_column = [];
    for j=1:logical_type
      if (circuit_input(i) == j - 1)
        circuit_input_column = [circuit_input_column;1];
      else
        circuit_input_column = [circuit_input_column;0];
      endif
    endfor
    if (i == 1)
      circuit_input_matrix = circuit_input_column;
    else
      circuit_input_matrix(:,i) = circuit_input_column;
    endif
  endfor

## Create two-dimensional array of consecutive integers in binary format
## from 0 to M^(N-1) where M is the logical order and N is the number of
## input qubits into the circuit.  These will be used later as inputs to
## the XOR computation.

for i=1:logical_type^(size(circuit_input)(1) - 1)
  closed_form_output(i) = 0;
  for j=1:size(circuit_input)(1) - 1
    binary_array(i, j) = 0;
  endfor
endfor

for i=1:logical_type^(size(circuit_input)(1) - 1) - 1
  for j=1:size(circuit_input)(1) - 1
    binary_array(i + 1, size(circuit_input)(1) - j) =
mod(floor(i/logical_type^(j-1)), logical_type);
  endfor
endfor

## Evaluate closed-form Deutsch-Jozsa expression for the given inputs.

for i=1:size(binary_array)(1)
  for j=1:size(binary_array)(1)
    temp_sum = 0;
    for k=1:size(circuit_input)(1) - 1
      temp_sum = generalized_XOR(logical_type, temp_sum,
binary_array(i,k) * binary_array(j,k));
    endfor
    XOR_sum = temp_sum + fx(j);
    closed_form_output(i) = closed_form_output(i) + exp(-1 * sqrt(-1) *
2 * pi * XOR_sum / logical_type);
  endfor
endfor
closed_form_output = conj(closed_form_output') /
logical_type^(size(circuit_input)(1) - 1);


endfunction
```

Appendix 4: Experimental Demonstration of Five Photon Entanglement

The generation of entangled pairs is an obvious requisite to any experiments involving manipulations of GHZ states[1]. The traditional approach to generating such states has involved the decay of an unstable particle into identical fragments with correlated spin directions, a realization of Bohm's *gedankenexperiment* [20]. However, it has recently been demonstrated [21] that independent sources can generate EPR[2] effects. Implementations using this approach possess the potential to overcome a serious limitation of the older method, that of its inability to generate a large number of particles sharing the same entangled state.

Using photons generated by independent sources, an experimentally realized generator of EPR states, as presented in its five photon form in a paper published in *Nature* [22] by Zhao et al., can be suitably modified to generate arbitrary N-photon entanglement. The five photon case will be described first, followed by its generalization and some practical considerations regarding actual implementations of this type of design.

[1] A GHZ state, named after Daniel Greenberger, Michael Horne and Anton Zeilinger, is any maximally entangled state involving N qubits, where N > 2. If N = 2, then the four maximally entangled states that may be generated are called Bell states.

[2] EPR is an acronym referring to Albert Einstein, Boris Podolsky and Nathan Rosen, coauthors of the seminal paper "Can the Quantum Mechanical Description of Physical Reality Be Considered Complete" which appeared in *Phys. Rev*. 47 (1935), pages 777-780. EPR effects refer to non-local (spacelike) correlations between separated particles prepared in an entangled state.

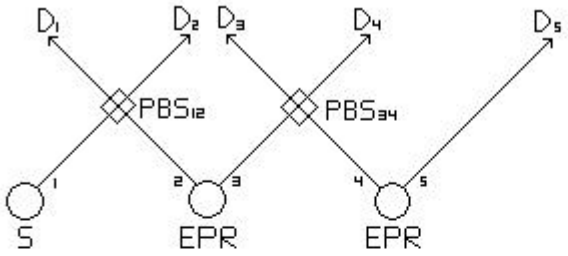

Figure A4.1: The apparatus used to achieve five photon entanglement, where D_i refers to photon detector i, PBS_{ij} refers to the polarizing beam splitter that acts upon beams i and j, S refers to a monochromatic photon source and EPR refers to a source of photons generated in a Bell state.

The detectors are only present for the purpose of verification; clearly they would not be present if the photons were to be employed in some way following the generation of their entangled state.

The physical basis of the orthogonal polarizations $\{|H\rangle,|V\rangle\}$ will be employed throughout. The monochromatic photon source S generates photons in the state $\frac{1}{\sqrt{2}}(|H\rangle + |V\rangle)$ and the two EPR sources produce entangled photons in the Bell state

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{A4.1}$$

Once produced, four of the photons interact[3] with the two polarizing beam splitters. Horizontally polarized photons pass through unimpeded and vertically polarized photons are reflected. In order to achieve the desired coincidence, such that

[3] In order to achieve five-fold coincidence, any experimental implementation of this design must ensure that close overlap in time occurs in regards to the interactions with both beam splitters and that the wavelengths of all photons involved closely match.

precisely one photon is vectored toward each detector, all five detectors must register either horizontally or vertically polarized photons. The desired state to be produced therefore is

$$\left|\Phi\right\rangle_{12345} = \frac{1}{\sqrt{2}}\left(\left|H_1\right\rangle\left|H_2\right\rangle\left|H_3\right\rangle\left|H_4\right\rangle\left|H_5\right\rangle + \left|V_1\right\rangle\left|V_2\right\rangle\left|V_3\right\rangle\left|V_4\right\rangle\left|V_5\right\rangle\right). \tag{A4.2}$$

The underlying principle behind this approach is one of chain entanglement. That is, photons 1 and 2 are entangled by virtue of a polarizing beam splitter. But photons 2 and 3 were already entangled through being generated in the same Bell state. Hence, photons 1 and 3 come to share the same state. This argument can be extended to include photons 4 and 5, and in fact, any number of photons N that one may wish to entangle through a generalization of this approach. Such generalization is very easy to achieve, as the following diagram demonstrates:

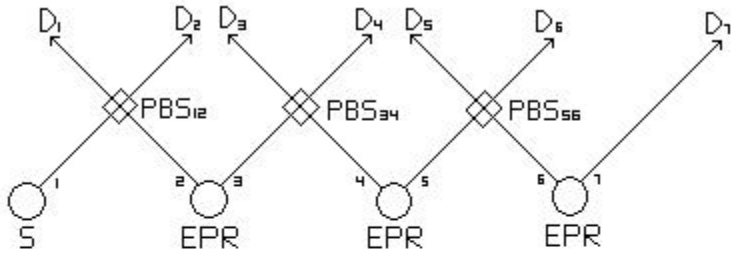

Figure A4.2: An extension of the five-fold entanglement approach to six or seven-fold entanglement. (If only six-fold entanglement is desired, photon 7 may simply be ignored.)

In regards to practical implementations of this approach, economical considerations certainly are a factor that the experimentalist should consider. One potential economization is a reduction of the number of detectors necessary to verify that

the desired N-fold coincidence has indeed occurred. The exact number of detectors necessary varies depending upon the assumed properties of the detectors employed. Clearly, very basic detectors only capable of distinguishing whether or not they have observed one or more photons provide less useful information to the experimenter than do detectors distinguishing between whether they have encountered one or two photons, the polarization state of the detected photons, or both.

S	EPR_1	EPR_2	EPR_3	EPR_4		D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9
H	H	H	H	H		H	H	H	H	H	H	H	H	H
H	H	H	H	V		H	H	H	H	H	H	-	H+V	V
H	H	H	V	H		H	H	H	H	-	H+V	H+V	-	H
H	H	H	V	V		H	H	H	H	-	H+V	V	V	V
H	H	V	H	H		H	H	-	H+V	H+V	-	H	H	H
H	H	V	H	V		H	H	-	H+V	H+V	-	-	H+V	V
H	H	V	V	H		H	H	-	H+V	V	V	H+V	-	H
H	H	V	V	V		H	H	-	H+V	V	V	V	V	V
H	V	H	H	H		-	H+V	H+V	-	H	H	H	H	H
H	V	H	H	V		-	H+V	H+V	-	H	H	-	H+V	V
H	V	H	V	H		-	H+V	H+V	-	-	H+V	H+V	-	H
H	V	H	V	V		-	H+V	H+V	-	-	H+V	V	V	V
H	V	V	H	H		-	H+V	V	V	H+V	-	H	H	H
H	V	V	H	V		-	H+V	V	V	H+V	-	-	H+V	V
H	V	V	V	H		-	H+V	V	V	V	V	H+V	-	H
H	V	V	V	V		-	H+V	V	V	V	V	V	V	V
V	H	H	H	H		H+V	-	H	H	H	H	H	H	H
V	H	H	H	V		H+V	-	H	H	H	H	-	H+V	V
V	H	H	V	H		H+V	-	H	H	-	H+V	H+V	-	H
V	H	H	V	V		H+V	-	H	H	-	H+V	V	V	V
V	H	V	H	H		H+V	-	-	H+V	H+V	-	H	H	H
V	H	V	H	V		H+V	-	-	H+V	H+V	-	-	H+V	V
V	H	V	V	H		H+V	-	-	H+V	V	V	H+V	-	H
V	H	V	V	V		H+V	-	-	H+V	V	V	V	V	V
V	V	H	H	H		V	V	H+V	-	H	H	H	H	H
V	V	H	H	V		V	V	H+V	-	H	H	-	H+V	V
V	V	H	V	H		V	V	H+V	-	-	H+V	H+V	-	H
V	V	H	V	V		V	V	H+V	-	-	H+V	V	V	V
V	V	V	H	H		V	V	V	V	H+V	-	H	H	H
V	V	V	H	V		V	V	V	V	H+V	-	-	H+V	V
V	V	V	V	H		V	V	V	V	V	V	H+V	-	H
V	V	V	V	V		V	V	V	V	V	V	V	V	V

Table A4.1: A tabulation for the nine-fold entanglement generator of the photon number and polarization state received by each detector for all possible input combinations. H and V refer to horizontally and vertically polarized photons, respectively. The "-" represents no photon and H+V represents the vectoring of two orthogonally polarized photons toward a single detector.

It can be inductively demonstrated, as can be directly observed for the particular case of nine-fold entanglement generation presented in Table 1, that if the detectors cannot distinguish anything except whether they have received some input of photons or not, all detectors must be present if N is even. Otherwise, the final detector D_N is superfluous. If the detectors can distinguish between horizontally and vertically polarized photons or if they can enumerate the number of photons that they receive, all even numbered detectors are redundant, as is D_N for odd N.

A related practical concern is that of production efficiency. The desired N-fold entangled state is only generated when an N-fold coincidence is achieved, the efficiency of which, as Table 1 would suggest, becomes rather low as N increases. In fact, precisely two combinations of input produce the desired entanglement state regardless of the value N assumes. In general, the production efficiency, assuming all other experimental considerations are ideal, is given by $2^{-Floor\left(\frac{N}{2}\right)}$, where the floor function rounds to the nearest integer less than or equal to its argument.

Bibliography

[1] Turing, A. **"**On Computable Numbers, with an Application to the Entscheidungsproblem." *Proc. London Math. Soc.*, Vol. 2, No. 42. (1936), 230-265.

[2] Benenti, G., G. Casati and G. Strini. *Principles of Quantum Computation and Information*. World Scientific: Singapore. (2004), 26.

[3] Domokos, P., J. M. Raimond, M. Brune and S. Haroche. "Simple Cavity-QED Two-Bit Universal Quantum Logic Gate: The Principle and Expected Performances." *Phys. Rev. A* 52 (1995), 3554–3559.

[4] Steane, A. "The Ion Trap Quantum Information Processor." *Appl.Phys. B 64* (1997), 623.

[5] Benioff, P. "The Computer as a Physical System: A Microscopic Quantum Mechanical Hamiltonian Model of Computers as Represented by Turing Machines." *J. Stat. Phys.* 22 (1980), 563.

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

[7] Deutsch, D. "Quantum Theory, the Church-Turing principle and the Universal Quantum Computer." *Proc. Roy. Soc.Lond A* 400 (1985), 97-117.

[8] Deutsch, D. and R. Jozsa. "Rapid Solution of Problems by Quantum Computation." *Proceedings of the Royal Society of London, Series A* 439 (1992), 553.

[9] Shor, P. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer." 1994 <quant-ph/9508027>

[10] Grover, L. "Quantum Mechanics Helps in Searching for a Needle in a Haystack." *Phys. Rev. Lett.* 79(2) (1997), 325-328.

[11] Childs, A.M., R. Cleve, E. Deotto, E. Farhi, S. Gutmann and D.A. Spielman. "Exponential Algorithmic Speedup by Quantum Walk." (2002) <quant-ph/0209131>

[12] Lieven, M., K. Vandersypen, M. Steffen, G. Breyta, C.S. Yannoni, M. Sherwood and I. Chuang. "Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance." *Nature* 414 (2001), 883-887.

[13] Vedral, V., A. Barenco and A. Ekert. "Quantum Networks for Elementary Arithmetic Operations." *Phys. Rev. A* 54 (1996), 147.

[14] Nielsen, M. and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press: Cambridge (2000).

[15] Berman, G., G. Doolen, R. Mainieri and V. Tsifrinovich. *Introduction to Quantum Computers*. World Scientific: Singapore (1998).

[16] Lo, H.K., S. Popescu and T. Spiller. *Introduction to Quantum Computation and Information*. World Scientific: Singapore (1998).

[17] Benenti, G., G. Casati and G. Strini. *Principles of Quantum Computation and Information*. World Scientific: Singapore. (2004), 26.

[18] Fujii, K. "Quantum Optical Construction of Generalized Pauli and Walsh-Hadamard Matrices in Three Level System." (2003), equation 90. < quant-ph/0309132>

[19] Nielsen, M. and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press: Cambridge (2000), 34-37.

[20] Bohm, D. *Quantum Theory*. Prentice-Hall: Englewood Cliffs, NJ (1951), 615-619.

[21] Yurke, B. and D. Stoler. "Einstein-Podolsky-Rosen Effects From Independent Particle Sources." *Phys. Rev. Lett.* 68 (1992), 1251-1254.

[22] Zhao, Z., Y. Chen, A. Zhang, T. Yang, H. Briegel and J. Pan. "Experimental Demonstration of Five-Photon Entanglement and Open-Destination Teleportation." *Nature* 430 (2004), 54-58.