Quantum Computer Programming

(Proposal Type: Quantum Algorithm)

A Proposal for ARO Solicitation
DAAD19-02-R-0005

PI: Steven J. Leon
Co-PI: Robert R. Tucci

A. TABLE OF CONTENTS

C. PROJECT ABSTRACT

(Proposal Type: Quantum Algorithm) In Ref. [Tuc99], one of the investigators (Dr. Robert Tucci) has given an algorithm for reducing an arbitrary unitary matrix U into a sequence of elementary operations (i.e., operations that act on one or two qubits only, such as controlled nots and qubit rotations). The algorithm given by Tucci applies recursively a mathematical technique called the CS decomposition. Since 1998, Tucci has made available at his website the C++ code for a computer program called "Qubiter" (covered by US PATENT 6,456,994) that implements his algorithm. The other investigator in the project is Prof. Steven Leon. Leon is an expert on the CS decomposition. His routines for computing the CS decomposition were the first to be made available in public software libraries (the Net-Lib library at Oakridge National Laboratory and the MATLAB User's Toolbox, The Mathworks). Prof. Leon is the author of two textbooks Refs.[LeonLA6][ATLAST] on Linear Algebra. Tucci received his Ph.D in Theoretical Physics and Leon in Mathematics, so they bring complementary skills into this collaboration. Furthermore, they are both experienced and avid programmers in C++ , Matlab, etc. The goal of this project is to enhance the Qubiter software and the theory behind it. Quantum Computing urgently needs new algorithms for performing specific tasks. An enhanced Qubiter would be a general purpose tool that would greatly aid in the search of such algorithms.

D. PROJECT DESCRIPTION

Introduction

In classical digital computation, one deals with sequences of elementary operations (operations such as AND, OR and NOT). These sequences are used to manipulate an array of classical bits. The operations are elementary in the sense that they act on only a few bits (1 or 2) at a time. Henceforth, we will sometimes refer to sequences as products and to operations as operators, matrices, instructions, steps or gates. Furthermore, we will abbreviate the phrase "sequence of elementary operations" by "SEO". In Quantum Computation, one also deals with SEOs (with operations such as controlled-nots and qubit rotations), but for manipulating quantum bits (qubits) instead of classical bits. Quantum SEOs are often represented graphically by qubit circuits.

In Quantum Computation, one is often given a unitary operator *U* that describes the evolution of an array of qubits. One must then find a way to reduce *U* into a SEO. In Ref.[Tuc99], one of the investigators in this project (Tucci) presented a new algorithm, based on a mathematical technique called the CS Decomposition (CSD), for accomplishing this task. In Ref.[Tuc99], Tucci also reported on a computer program called "Qubiter" that implements his algorithm. Qubiter is covered by US PATENT 6,456,994. Its C++ source code is publicly available at www.ar-tiste.com/qubiter.html. We call Qubiter a "quantum compiler" because, like a classical compiler, it produces a SEO for manipulating bits.

Qubiter's algorithm can be applied to any unitary operator *U*.

It is useful to define certain unitary operators U_{N_B} for all $N_B \in \{1,2,3,\ldots\}$, where U_{N_B} is a $2^{N_B} \times 2^{N_B}$ matrix and N_B is the number of bits. Some U_{N_B} are known to be expressible as a SEO whose length (i.e., whose number of elementary quantum operations) is a power (rather than an exponential) of N_B. Two examples are the N_B -bit Hadamard Transform (HT) matrix and the N_B -bit Discrete Fourier Transform (DFT) matrix. The HT matrix is known to be expressible as a SEO of length Order(N_B). The DFT matrix is known to be expressible as a SEO of length Order($N_B{}^2$). Qubiter already achieves both of these SEO-length benchmarks for N_B =2,3,4.

Although Qubiter yields short SEOs for many unitary operators such as the HT and DFT matrices, it does not yield the shortest possible SEO for every unitary operator. Is it possible to enhance Qubiter so that it does? Given a unitary matrix *U*, is it possible to construct a unitary matrix $\hat{U}$ such that: (1) $\hat{U}$ approximates *U* in some sense, and (2) $\hat{U}$ has a SEO that is significantly shorter than the SEO of *U*? These are the types of questions that this project will attempt to answer.

<u>CSD and Qubiter's algorithm</u>

In this section, we will give a brief description of the CSD and Qubiter's algorithm.

First, let us state the CSD Theorem. The C and S stand for "cosine" and "sine", respectively. See Ref.[Pai94] for a review of the history of CSD.

Suppose that U is an $N \times N$ unitary matrix, where N is an even number. Then the CSD Theorem states that one can always express *U* in the form:

$$U = \begin{bmatrix} L_0 & 0 \\ 0 & L_1 \end{bmatrix} D \begin{bmatrix} R_0 & 0 \\ 0 & R_1 \end{bmatrix}, \tag{1}$$

where the left and right side matrices L_0, L_1, R_0, R_1 are $\frac{N}{2} \times \frac{N}{2}$ unitary matrices and

$$D = \begin{bmatrix} D_{00} & D_{01} \\ D_{10} & D_{11} \end{bmatrix}, \tag{2}$$

$$D_{00} = D_{11} = \text{diag}(C_1, \quad C_2, \quad \cdots \quad C_{\frac{N}{2}}), \tag{3}$$

$$D_{01} = \text{diag}(S_1, \quad S_2, \quad \cdots \quad S_{\frac{N}{2}}), \tag{4}$$

$$D_{01} = -D_{10}. \tag{5}$$

For all $i \in \left\{1, 2, 3\ldots, \frac{N}{2}\right\}$, $C_i = \cos(\theta_i)$ and $S_i = \sin(\theta_i)$ for some angle θ_i.

Henceforth, we will use the term *D matrix* to refer to any matrix that satisfies Eqs.(2) to (5). If one partitions U into four blocks $U_{i\,j}$ of size $\frac{N}{2}\times\frac{N}{2}$, then

$$U_{i\,j} = L_i\, D_{i\,j}\, R_j \tag{6}$$

for $i, j \in \{1, 0\}$. Thus, $D_{i\,j}$ gives the singular values of $U_{i\,j}$.

Note that if U were a general (not necessarily unitary) matrix, then the four blocks $U_{i\,j}$ would be unrelated. Then to find the singular values of the four blocks $U_{i\,j}$ would require eight unitary matrices (two for each block), instead of the four L_i, R_j. This double use of the L_i, R_j is a key property of the CSD.

Next, we will give a bird's eye view of Qubiter's algorithm. For more details, see Ref.[Tuc99].

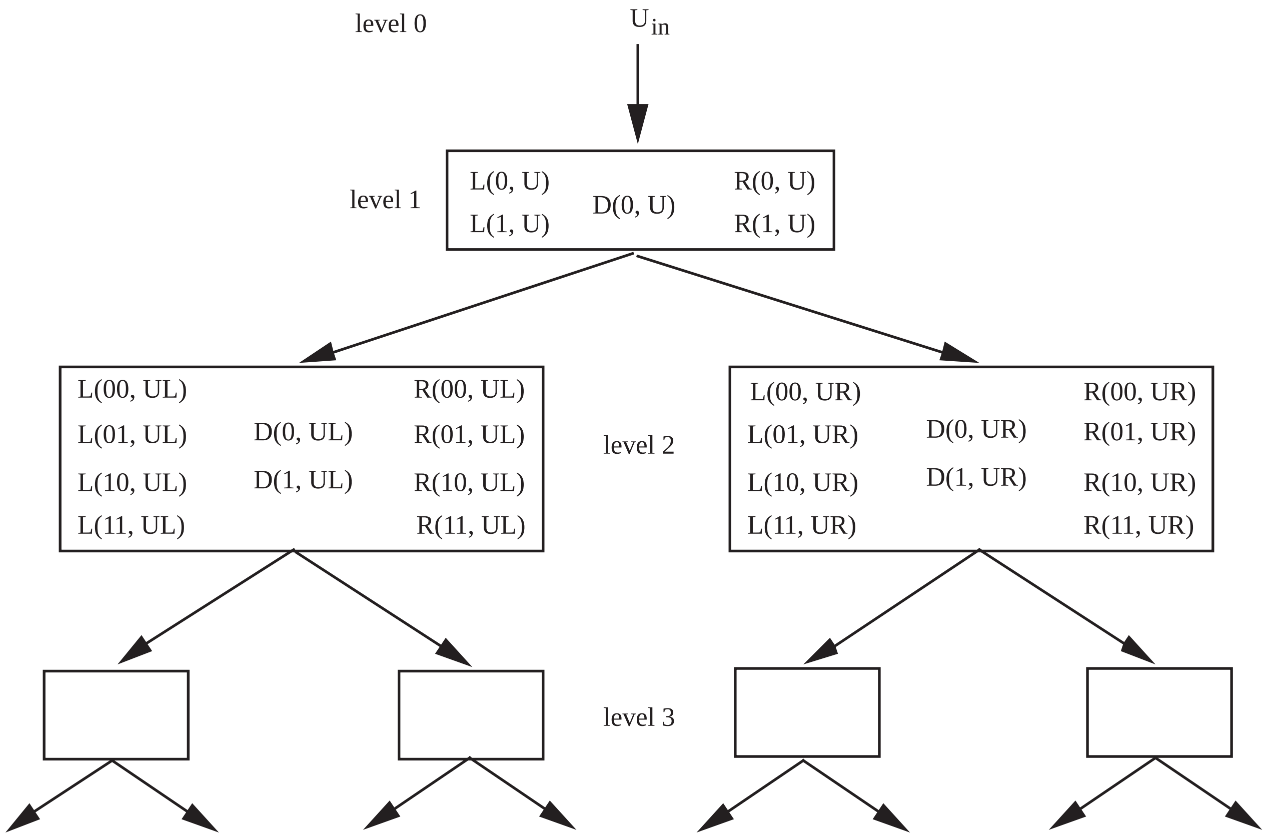

Figure 1. A CSD binary tree

Consider Fig.1. We start with an initial unitary matrix U_{in} at horizontal level 0. Without loss of generality, we can assume that the dimension of U_{in} is 2^{N_B} for some $N_B \geq 1$. (If initially U_{in}'s dimension is not a power of 2, we replace U_{in} by a direct sum $U_{in} \oplus \text{diag}(1,1,\ldots 1)$ whose dimension is a power of two.) We apply the CSD method to U_{in}. This yields for level 1 a *D* matrix $D(0,U)$, two unitary matrices $L(0,U)$ and $L(1,U)$ on the left side, and two unitary matrices $R(0,U)$ and $R(1,U)$ on the right side. Then we apply the CSD method to each of the 4 matrices $L(0,U)$, $L(1,U)$, $R(0,U)$ and $R(1,U)$ that were produced in the previous step. Then we apply the CSD method to each of the 16 *R* and *L* matrices that were produced in the previous step. And so on. At level N_B, the *L*'s and *R*'s are 1×1 dimensional---i.e., just unit-modulus complex numbers.

Call a *central matrix* either (1) a single *D* matrix, or (2) a direct sum $D_1 \oplus D_2 \oplus \cdots \oplus D_r$ of *D* matrices, or (3) a diagonal unitary matrix. From Fig.1 it is clear that the initial matrix U_{in} can be expressed as a product of central matrices, with each node of the tree providing one of the central matrices in the product. In Ref.[Tuc99], we show how to decompose any central matrix into a SEO.

<u>Related Work</u>

Qubiter's algorithm is not the only algorithm that has been proposed for decomposing an arbitrary unitary matrix into a SEO. Another algorithm for doing this was first proposed by Barenco et al in Ref.[Bar95], and it was later repeated by Cybenko in Ref.[Cyb01]. Like Qubiter's algorithm, their algorithm can be applied to any unitary operator *U*. However, it is very unlikely that their algorithm will be efficient at producing short SEOs unless further optimizations are added to it. And such optimizations, if they exist, have not been specified by anyone. Furthermore, as far as we know, there is no publicly available software that implements their algorithm. Qubiter's algorithm is significantly different from theirs. Theirs is based on a mathematical technique described in Ref.[Mur58], whereas Qubiter's algorithm is based on a mathematical technique called the CS Decomposition (CSD). Qubiter's algorithm applies recursively the CSD to build a ***binary*** tree of matrices whose product, in some order, equals the original matrix *U*. Furthermore, the CSD gives ***two*** same-sized *R* matrices (resp. *L* matrices) on the right side (resp. left side) of a central matrix. Because of these binary symmetries, Qubiter's algorithm is a natural tool to use for analyzing two state systems (qubits). By contrast, the Barenco et al algorithm does not have any explicit binary symmetries—it seems to lack the natural symmetry of the problem. It would be very interesting to study how Qubiter's algorithm is related to the one by Barenco et al, and whether they can be merged.

Goals of Project

The goal of this project is to enhance the Qubiter software and the theory behind it. This general goal comprises the following four sub-goals:

GOAL 1. Find "good approximations" of a unitary matrix: Let $\boldsymbol{U(N_B)}$ be the set of $2^{N_B} \times 2^{N_B}$ unitary matrices. For any non-negative integer α, let $\boldsymbol{U(N_B, \alpha)}$ contain all matrices U in $\boldsymbol{U(N_B)}$ for which there exists a SEO whose length is less than or equal to $(N_B)^{\alpha}$. For any $\varepsilon > 0$, and any matrix U in $\boldsymbol{U(N_B)}$, find the smallest α for which there exists an approximation matrix $\hat{U}$ in $\boldsymbol{U(N_B, \alpha)}$ such that $\|U - \hat{U}\| < \varepsilon$, for some metric $\| . \|$. For any non-negative integer α, and any matrix U in $\boldsymbol{U(N_B)}$, find an approximation matrix $\hat{U}$ in $\boldsymbol{U(N_B, \alpha)}$ that minimizes $\|U - \hat{U}\|$. Find an algorithm (based on the CSD) for constructing such a $\hat{U}$. Write Matlab and C++ code that implements such an algorithm, and incorporate the code into Qubiter. We believe a quantum computer can perform many useful calculations that are not too sensitive to the replacement of one of the steps, represented by a U in $\boldsymbol{U(N_B)}$, by an approximation $\hat{U}$ in $\boldsymbol{U(N_B, \alpha)}$. The importance of approximations of a unitary matrix in Quantum Computing is already evident in the case of the Quantum Discrete Fourier Transform (Ref.[Cop94]), where some gates with a very small phase can be omitted with impunity.

GOAL 2. Find ways of exploiting non-uniqueness of Qubiter algorithm to optimize SEO-length: There are many situations when Qubiter's algorithm is non-unique. For example, when the angles θ_i of the D matrix are not all distinct, the matrices L_0, L_1, R_0, R_1, given by the CSD are not unique. A second example: the decomposition of the previously defined central matrices into SEO's is highly non-unique. Qubiter has to resolve such non-uniqueness by making particular choices. It can make good choices that yield nearly the shortest possible SEO. Or, it can make bad choices, in which case it might end up decomposing a matrix which belongs to $\boldsymbol{U(N_B, \alpha)}$ into a SEO whose length is exponential in N_B . For example, in decomposing the N_B-bit HT and DFT matrices, Qubiter can resolve the non-uniqueness in a way that causes most of the

CSD tree for these matrices to degenerate into a single branch. (This is possible because when all the R (resp., L) matrices of a node are equal to the identity matrix, then that node will not have any child on its right (resp., left) side). Or, if Qubiter makes bad choices in resolving the non-uniqueness, it will end up with a full CSD tree and a SEO whose length is exponential in N_B.

GOAL 3. Handle Special Matrices Differently: An important way to improve Qubiter efficiency is by treating differently certain special types of unitary matrices. For example, to decompose a deterministic unitary matrix into a SEO, there might be methods from classical digital circuit theory (Karnaugh diagrams, etc.) that are more efficient than CSD. Another example: we might discover that for a special type of unitary matrix, the Barenco et al method is more efficient than CSD. Such algorithms could be incorporated into Qubiter.

GOAL 4. Study and enhance stability of Qubiter code: As with any numerical Linear Algebra software, it is of paramount importance to study and improve the stability of the code. One wants to minimize round-off errors, and to minimize the impact of those errors on the final answer.

Three Year Research Plan

We consider GOAL 1 to be the most difficult one, but the one with the largest payoff if it can be achieved. We therefore propose to work constantly, all three years, on GOAL 1. The other 3 goals will be tackled mainly at the rate of one per year. Thus, we would try to follow this plan:

Year 1: Work on GOAL 1 and GOAL 2.

Year 2: Work on GOAL 1 and GOAL 3.

Year 3: Work on GOAL 1 and GOAL 4. Wrap up the project.

Impact of Research

Quantum Computing urgently needs new algorithms for performing specific tasks. An enhanced Qubiter would be a general purpose tool that would greatly aid in the search of such algorithms.

Our work would build bridges between two communities: (1) developers of Linear Algebra software (e.g., authors of Matlab, Lapack, etc.) and (2) Quantum Computing researchers . Currently these two communities rarely interact. We believe community (1) has much to contribute to Quantum Computing.

Investigators, Students

There will be two investigators: Robert R. Tucci and Steven J. Leon, dividing the work-load about equally.

Dr. Tucci has considerable experience in quantum computers and quantum information theory. He is the author of 15 papers (all available at the arXiv eprint library) on this subject. He has also written 3 substantial computer programs in the field: (1) Quantum Fog (patented), a quantum computer simulator (2) Qubiter(patented), a quantum compiler (3) Causa Común (described in Ref.[Tuc01]), a program for calculating entanglement of formation .

Prof. Steven Leon is an expert on the CS decomposition. He was one of the first to work extensively with this factorization. His subroutines for the CS decomposition and the Generalized Singular Value Decomposition were the first to be included in public software libraries. (The Net-Lib library, Oakridge National Lab, and the MATLAB User Toolbox, The Mathworks, Natick, MA). Prof. Leon is the author of two Linear Algebra textbooks Refs.[LeonLA6][ATLAST] .

Tucci has a PhD in Theoretical Physics and Leon in Mathematics, so they bring complementary skills into this collaboration. Furthermore, they are both experienced and avid programmers in C++ , Matlab, etc.

Additionally there will be one graduate student assisting Professor Leon and Dr. Tucci on the project. The graduate student will be selected by Dr. Leon from the most qualified candidates in the Physics, Mathematics, Engineering, or Computer Sciences programs at UMass Dartmouth. The project will introduce the graduate student to Quantum Computing, and train him on how to conduct original research and work in a research team. It will be a valuable educational experience for the student, and it will recruit the student to work in a field where there is a critical need for trained researchers.

Resources-Existing & Planned

This project will not require the use of any existing or planned laboratory equipment, other than personal computers. The project will, however, use certain existing resources in intellectual property. In fact, the investigators of this project will enjoy during this project an advantage that other investigators working on a similar topic may not enjoy, namely, free, and unfettered access to the following privately owned intellectual property:

- existing Qubiter software
- US PATENT 6,456,994 covering Qubiter and Qubiter-like software.

Budget Requirements

For each of the three years, as financial support for Dr. Leon's research, we are asking for 20% of his salary during the academic year and for 2 months salary during the summer.

Each year Robert Tucci will devote 75% of his work time to this project. As financial support for Dr. Tucci, we are requesting 75% of his yearly salary .

We are also requesting yearly support for one graduate student.

Additionally we are requesting three personal computers, software, and travel costs.

F. BIBLIOGRAPHY

- [ATLAST] S. Leon, G. Herman, R. Faulkenberry, *ATLAST Computer Exercises for Linear Algebra* (Prentice Hall, 1997) .
- [Bar95] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J.H. Smolin, H. Weinfurter, Physical Review A **52** , 3457 (1995).
- [Cop94] D. Coppersmith, "An Approximate Fourier Transform Useful in Quantum Factoring", IBM Research Rep. No. 19642, 1994. Also available as quant-ph/0201067
- [Cyb01] G. Cybenko, "Reducing Quantum Computations to Elementary Unitary Operations". Comput. Sci. Eng. 3 (2): 27-32 Mar. – Apr. 2001.
- [Golub96] Gene H. Golub and Charles F. Van Loan, *Matrix Computation, 3rd Edition,* The Johns Hopkins University Press (1996).
- [LeonLA6] S. Leon, *Linear Algebra with Applications, 6th Edition* (Prentice Hall, 2002)
- [Mur58] F. D. Murnaghan, *The Orthogonal and Symplectic Groups* (Institute for Advanced Studies, Dublin, 1958).
- [Pai94] C.C. Paige, M. Wei, "History and Generality of the CS Decomposition", *Linear Algebra And Its Applications* **208**, 303 (1994).
- [Tuc99] R.R. Tucci, "A Rudimentary Quantum Compiler (2cnd Ed.)", quant-ph/9902062
- [Tuc01] R.R. Tucci, "Relaxation Method For Calculating Quantum Entanglement", quant-ph/ 0101123
- [Van85] C.F. Van Loan, "Computing the CS and Generalized Singular Value Decomposition", *Numer. Math,* **46** (1985).