

Computational Synthesis of any N -Qubit Pure or Mixed State

Lin Song and Colin P. Williams*
Mail Stop 126-347, Jet Propulsion Laboratory,
California Institute of Technology, Pasadena, CA 91109-8099

ABSTRACT

Future quantum information processing devices will require the use of exotic quantum states, such as specially crafted entangled states, to achieve certain desired computations on demand. Thus far, synthesis schemes for such states have been devised on a case-by-case basis using ad hoc techniques. In this paper we present a systematic method for finding a quantum circuit that can synthesize any pure or mixed n -qubit state. We then give examples of the use of our algorithm for finding synthesis pathways for especially exotic quantum states such as maximal mixed states. It is not known how to prepare general instances of such states by other means. Thus our quantum state synthesis algorithm should be of use not only in quantum information processing, but also in experimental quantum physics.

Keywords: pure, mixed, state synthesis, quantum circuit, quantum network, quantum computing

1. INTRODUCTION

Future quantum information processing devices will require the use of exotic quantum states, such as specially crafted entangled states, to achieve certain desired computations on demand. For example, linear optics/projective measurement quantum computing requires certain entangled states to be prepared offline in order to achieve a *CNOT* gate on demand¹. Even if these entangled states can only be generated inefficiently then, provided a quantum memory register is available, they can be created offline, stored, and then brought (or teleported) into the computational stream as needed to obtain deterministic quantum logic operations on demand². Moreover, the ability to synthesize arbitrary pure states provides a direct method of creating a source of true randomness with any desired bias built in. Such states, in conjunction with quantum measurements, would overcome the negative effects of hidden correlations that can bedevil high-dimensional Monte-Carlo integration on a classical computer using a pseudo-random source³. It is therefore a question of general interest to quantum computing as to how arbitrary n -qubit pure states, including exotic entangled states, might be prepared.

Besides pure state preparation, in recent years physicists have been developing more general techniques for synthesizing various types of mixed states. So far, mixed states have found fewer applications in quantum information processing than pure states⁴. In part this is because far less is known about the properties of mixed states than pure states. What is known has come from detailed studies of few-qubit systems. For example, for 2-qubit mixed states, the “tangle” and “linear entropy” were introduced to quantify their simultaneous degrees of entanglement and mixedness respectively. Numerical studies have revealed that there is a sharp boundary in the “tangle”/“linear entropy” plane dividing a region of physically-possible mixed states from a region of physically impossible mixed states⁵. With this new understanding, it was recognized that no-one had ever synthesized a certain type of “maximal” mixed state which lies in the region of the tangle/linear entropy plane between the Werner states and this boundary. Only recently was a particular maximal state synthesized optically by Kwiat, White et al.⁶. However, the question of how to synthesize an *arbitrary* mixed state remains open.

In this paper we provide a solution to the problem of synthesizing an arbitrary pure or mixed state defined on n -qubits. Our approach treats state-preparation as a quantum computational problem, by first computing a unitary operator, U , capable of deterministically synthesizing the desired pure or mixed state starting from the state $|00\dots0\rangle$, and then decomposing U into an equivalent quantum circuit comprising only 1-qubit and 2-qubit quantum logic gates. Unlike previous synthesis schemes which were all non-deterministic⁷, case-specific⁸ and hardware dependent⁹, our approach is deterministic, general, and not tied to any particular physical hardware scheme. The quantum circuit abstraction provides

* Email: Colin.P.Williams@jpl.nasa.gov, Tel: (818) 393 6998.

a unified description language for a synthesis pathway that can be specialized to one of several quantum hardware contexts by, for example, adjusting the choice of 2-qubit gate to best fit the desired hardware scheme¹⁰.

The paper is organized as follows: in Sections 2 and 3 we describe our scheme for synthesizing an arbitrary pure state defined on n -qubits, and give an example of its use for synthesizing a particular 2-qubit pure state. Section 4 uses this pure-state synthesis technique as a sub-routine in our more general mixed state synthesis procedure. In Section 5 we demonstrate use of this technique to make a particular “maximal” mixed state. Such states are highly non-classical and lie at the boundary of the possible/impossible regions of the tangle/linear entropy plane. The synthesis algorithms developed in Sections 2 and 4 both exploit a scheme for decomposing an arbitrary unitary operator into an equivalent quantum circuit¹¹. We describe the details of this decomposition scheme in Section 6. Finally, in Section 7, we conclude with a discussion of a possible application of arbitrary pure state synthesis to as a means of entering data into a quantum computer in order to perform quantum signal, quantum data, and quantum image processing tasks.

2. PURE STATE SYNTHESIS

Suppose we wish to synthesize the state $|\psi\rangle = \sum_{i=0}^{2^n-1} c_i |i\rangle$ with $\sum_{i=0}^{2^n-1} |c_i|^2 = 1$. Without loss of generality, we may assume that $c_0 \neq 0$. Otherwise, if $c_0 = 0$ we simply change to a basis $\{|i'\rangle\}$ such that $\sum_{i=0}^{2^n-1} c_i |i\rangle \equiv \sum_{i'=0}^{2^n-1} c'_i |i'\rangle$ with $c'_0 \neq 0$. One can think of the state $|\psi\rangle$ as being equivalent to a column vector of complex amplitudes. Our algorithm for synthesizing $|\psi\rangle$ starts with this column vector of amplitudes, embeds it as the leftmost column of a larger matrix having linearly independent columns, and applies the Gram-Schmidt procedure to construct a set of columns orthonormal to the column vector representation of $|\psi\rangle$. Specifically, our **SynthesizePureState** algorithm works as follows:

Algorithm: **SynthesizePureState**

Step 1: Map the state to be synthesized into a basis such that $c_0 \neq 0$ in this basis.

Step 2: Construct the padded matrix M defined as:

$$M = \begin{pmatrix} c_0 & | & & & \\ c_1 & & 1 & & \\ \vdots & & & 1 & \\ \vdots & & & & 1 \\ c_{2^n-1} & & & & 1 \end{pmatrix}$$

Step 3: Compute the matrix U , obtained by applying the Gram-Schmidt orthogonalization procedure M .

Step 4: As the first column of M is properly normalized, and (since $c_0 \neq 0$) the columns of M are guaranteed to be linearly independent, and the matrix U will always be unitary. Devising a quantum circuit for U will provide a constructive method for synthesizing the state $|\psi\rangle$ starting from the state $|00\dots 0\rangle$, i.e., $U \cdot |00\dots 0\rangle = |\psi\rangle$.

The mapping between a unitary matrix and an equivalent quantum circuit is explained in Section 6 and elsewhere¹¹.

3. EXAMPLE: SYNTHESIS OF A PURE STATE

Suppose we wish to synthesize the pure state $|\psi\rangle = \frac{1}{2}|00\rangle - i\frac{\sqrt{3}}{2}|11\rangle$. We can use **SynthesizePureState** to compute a synthesis pathway for this state. The column vector of amplitudes for $|\psi\rangle$ is embedded as the leftmost column of a matrix, and our goal is to find an orthonormal set of column vectors sufficient to make this matrix unitary. The matrix is padded down the diagonal with 1s, and the Gram-Schmidt procedure applied. This yields the matrix U , shown below:

$$M = \begin{pmatrix} \frac{1}{2} & | & 0 & 0 & 0 \\ 0 & & 1 & 0 & 0 \\ 0 & & & 0 & 1 & 0 \\ -i\frac{\sqrt{3}}{2} & & & 0 & 0 & 1 \end{pmatrix} \xrightarrow{\text{Gram-Schmidt Procedure}} U = \begin{pmatrix} \frac{1}{2} & 0 & 0 & -i\frac{\sqrt{3}}{2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -i\frac{\sqrt{3}}{2} & 0 & 0 & \frac{1}{2} \end{pmatrix}$$

Devising a quantum circuit that implements U is then guaranteed to provide a constructive, deterministic, method for synthesizing $|\psi\rangle = \frac{1}{2}|00\rangle - i\frac{\sqrt{3}}{2}|11\rangle$ starting from the state $|00\rangle$. The entire procedure, including mapping the constructed unitary matrix into an equivalent quantum circuit, is implemented in a *Mathematica* program called “QCD” (for

“Quantum Circuit Designer”). A sample run of the program for this problem is shown below together with the resulting quantum circuit QCD found.

```

Clear[u]; u = StateSynthesizer[ $\frac{1}{2}$  ket[0, 0] -  $\frac{i\sqrt{3}}{2}$  ket[1, 1]]


$$\begin{pmatrix} \frac{1}{2} & 0 & 0 & -\frac{i\sqrt{3}}{2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{i\sqrt{3}}{2} & 0 & 0 & \frac{1}{2} \end{pmatrix}$$


ColumnVectorToKet[u.KetToColumnVector[ket[0, 0]]]

 $\frac{1}{2}$  ket(0, 0) -  $\frac{i}{2}\sqrt{3}$  ket(1, 1)

circ = MatrixToQuantumCircuit[u] // CompactifyQuantumCircuit

{direct(phase( $\frac{\pi}{2}$ ), i(2)), direct(rotz(- $\frac{\pi}{2}$ ), i(2)), direct(i(2), roty(- $\pi$ )), cnot(1, 2, 2), cnot(2, 1, 2),
direct(roty(1.0472), i(2)), cnot(2, 1, 2), direct(roty(-1.0472), i(2)), direct(i(2), roty( $\pi$ )), cnot(1, 2, 2), direct(rotz(- $\frac{\pi}{2}$ ), i(2))}

QuantumCircuitToDiagram[circ]



- Graphics -

```

4. MIXED STATE SYNTHESIS

The foregoing discussion assumed we wished to synthesize an arbitrary pure state on n -qubits. But what if, instead, we wished to synthesize an arbitrary mixed state, ρ , on n -qubits? We proceed as follows:

```

Algorithm: SynthesizeMixedState
Step 1: Compute the spectral decomposition of  $\rho$  as  $\rho = \sum_i p_i |i\rangle\langle i|$ 
Step 2: Compute a family of unitary operators,  $\{U_i\}$  such that  $U_i|0\rangle = |i\rangle$ 
Step 3: Devise a quantum circuit for performing the direct sum of these operators, i.e.,  $U_1 \oplus U_2 \oplus \dots$ 
Step 4: Compute a synthesis pathway for the “loaded dice” state  $|\phi\rangle_C = \sum_i \sqrt{p_i} |i\rangle$ 
Step 5: Synthesize the input (pure) state  $|\psi\rangle = |\phi\rangle_C \otimes \underbrace{|00\dots 0\rangle}_n$ 
Step 6: Push this state through  $U$  and trace over the control qubits (“C”), i.e., compute  $\text{Tr}_C(U|\psi\rangle\langle\psi|U^\dagger) = \rho$ . The result will be the desired mixed state,  $\rho$ .

```

The scheme works by using the spectral decomposition of the desired mixed state to identify a set of unitary matrices sufficient to synthesize the necessary eigenvectors found in the spectral decomposition. Then, these unitary matrices are combined, via their direct sum, into a conditional quantum logic circuit. Finally, a loaded-dice pure state is synthesized weighted in proportion to the representation of each eigenvector in the mixed state. Our mixed state synthesis scheme exploits the correspondence between a unitary matrix built from the direct sum of smaller unitary matrices, and an equivalent conditional quantum logic circuit. For example, the unitary matrix shown in Fig. 1 is equivalent to the conditional quantum logic circuit shown in Fig. 2.

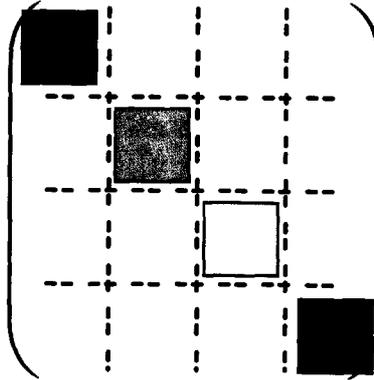


Fig. 1 Unitary matrix built from the direct sum of smaller unitary matrices.

This matrix is equivalent to the conditional quantum logic circuit shown in Fig. 2. Here a white (black) circle means that the associated gate acts if and only if the control qubit is $|0\rangle$ ($|1\rangle$).

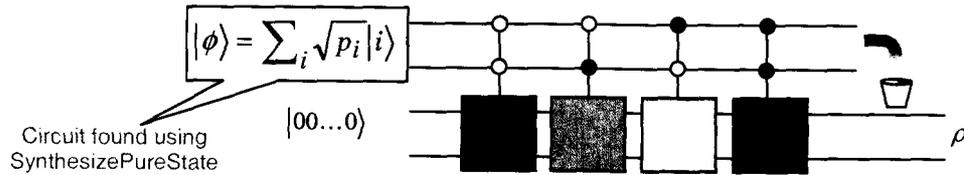


Fig. 2 Conditional quantum circuit for generating the matrix shown in Fig. 1.

5. EXAMPLE: SYNTHESIS OF A MAXIMAL STATE

As we mentioned in the introduction, Kwiat et al. have recently performed a valuable numerical study of the possible types of 2-qubit mixed states⁵, especially with respect to their allowed joint values of entanglement and mixedness (measured by their tangle and linear entropy respectively). The resulting diagram is repeated in Fig. 3 to aid comprehension:

$$\rho_{\text{Test}} = \begin{pmatrix} \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{3} \end{pmatrix}$$

SpectralDecomposition[\rho_{Test}]

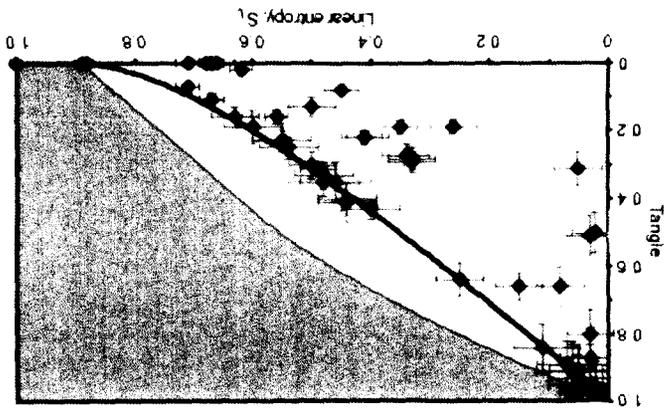
$$\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{3} \end{pmatrix} \begin{pmatrix} \text{ker}(1,1) \\ \text{ker}(0,0) \\ \text{ker}(0,1) \\ \text{ker}(1,1) \end{pmatrix} + \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{3} \end{pmatrix} \begin{pmatrix} \text{ker}(1,1) \\ \text{ker}(0,0) \\ \text{ker}(0,1) \\ \text{ker}(1,1) \end{pmatrix}$$

We can synthesize this density matrix (or indeed any other) using the **SynthesizeMixedState** algorithm. The resulting circuit begins

$$\rho_{\text{maximal}} = \begin{pmatrix} \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{3} \end{pmatrix}$$

Maximal states lie on a boundary in the tangle/entropy plane separating physically allowed states (white) from physically impossible ones (gray). For a given degree of mixedness, the maximal states have a greater tangle than the corresponding Werner states (black curve). A particular maximal mixed state has a density matrix of the form:

Fig. 3 Kwiat et al.'s plot of tangle versus linear entropy for feasible 2-qubit mixed states. The black line corresponds to unphysical states. The gray region corresponds to unphysical states. The black line corresponds to Werner states, and the boundary between the two contains maximal mixed states.



```

u = StateSynthesizer[ρTest]
(
0 0 0 0 0 0 0 0 0 -1/√2 0 0 1/√2 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1/√2 0 0 1/√2 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 1/√3 0 0 0 -√2/3 0 0 0 0 0 0 0 0 0
1/√3 0 0 0 -√2/3 0 0 0 0 0 0 0 0 0 0 0 0
0 1/√3 0 0 0 -√2/3 0 0 0 0 0 0 0 0 0 0 0
0 0 1/√3 0 0 0 -√2/3 0 0 0 0 0 0 0 0 0 0
1/√3 0 0 -1/√3 1/√6 0 0 -1/√6 0 0 0 0 0 0 0 0 0
0 √2/3 0 0 0 1/√3 0 0 0 0 0 0 0 0 0 0 0
0 0 √2/3 0 0 0 1/√3 0 0 0 0 0 0 0 0 0 0
1/√3 0 0 1/√3 1/√6 0 0 1/√6 0 0 0 0 0 0 0 0 0
)

```

A quantum circuit for this matrix yields a deterministic synthesis pathway for the desired maximal mixed state.

6. DECOMPOSING A UNITARY OPERATOR INTO AN EQUIVALENT QUANTUM CIRCUIT

A core component of the **SynthesizePureState** and **SynthesizeMixedState** algorithms is the ability to map an arbitrary unitary matrix into an equivalent quantum circuit. Previous approaches to decomposing arbitrary unitary operators have fallen into one of four categories. The majority of researchers use no formal scheme but instead rely upon trial and error, and human ingenuity, to arrive at a decomposition by hand. This approach is feasible for specially structured unitary matrices such as the Quantum Fourier Transform¹², and the quantum wavelet transform¹³, because the special structure of the unitary operator reflects a regular structure in the equivalent quantum circuit.

A second approach is to exhaustively enumerate the space of possible circuit designs of increasing complexity starting from the empty circuit^{14,15}. For each topologically distinct circuit, a computer finds optimal values for the parameters of all parameterized-gates in the circuit. In principle, this method is guaranteed to find the smallest circuit sufficient to implement the desired unitary operator. However, exhaustive search composed with numerical optimization is computationally expensive because the number of possible quantum circuit topologies grows exponentially with increasing numbers of gates in the circuit. Hence the method is only feasible for unitary operators that in fact have compact circuit descriptions.

A third approach uses genetic algorithms¹⁶. A random population of circuits is created, and each is assigned a "fitness" value that is a measure of how closely it comes to achieving the desired unitary operator. Pairs of circuits are selected for breeding in proportion to their fitness and then mutation and crossover operations are applied to make a new generation of circuits. By iterating this process one converges on a population of circuits that tend towards implementing the desired unitary operator. For genetic algorithms to work well, one needs a degree of decomposability in the problem, i.e., that part of the solution is basically correct while ignoring the rest. Because of the way the direct product of matrices tends to spread elements throughout the resulting matrix, it can be hard for a genetic algorithm to find satisfactory circuits for highly entangling unitary operators.

The fourth and most systematic approach is to apply a recursive algebraic decomposition procedure such as the progressive matrix diagonalization of Reck¹⁷, the "quantum Givens" operations of Cybenko¹⁸, or the hierarchical CS

decomposition of Tucci¹⁹. Algebraic factorization is guaranteed to work, but is likely to result in quantum circuits that are exponentially large unless one embeds circuit compactification rules within the decomposition procedure.

Our approach uses a recursive algebraic scheme for constructing a quantum circuit decomposition of an arbitrary unitary operator, interleaved with circuit compactification rules that reduce the complexity of the final quantum circuit¹¹. Our scheme starts with a similar decomposition to Tucci, but uses different techniques for mapping the matrix factors into equivalent circuit fragments. The procedure works as follows: first we decompose the $2^n \times 2^n$ dimensional unitary operator into a product of $2^n \times 2^n$ dimensional block-diagonal matrices, and direct sums of bit-reversal matrices (which need never be implemented explicitly). Next we map these block-diagonal matrices into corresponding quantum circuit fragments, each involving only 1-qubit rotations about the y- and z-axes, 1-qubit phase shifts, and a standard two-qubit gate, such as *CNOT*, \sqrt{SWAP} , or *iSWAP*. One can pick whichever primitive 2-qubit gate one wants and obtain different quantum circuits accordingly. The last step is to join these quantum circuit fragments together, while again applying circuit compactification rules to minimize the size of the resulting circuit. The net result is a quantum circuit capable of implementing any (real or complex) unitary matrix, specialized to use one of several types of 2-qubit gates, appropriate for different physical implementations of quantum computing hardware.

Here is a simple example for a random real 4×4 unitary matrix to illustrate the procedure:

```

m = RandomRealUnitary[4]

```

$$\begin{pmatrix} 0.342812 & -0.890319 & -0.2879 & -0.0832238 \\ -0.516979 & 0.0078719 & -0.425175 & -0.742898 \\ -0.488762 & -0.423697 & 0.756394 & -0.0972615 \\ 0.613453 & 0.166588 & 0.405225 & -0.657051 \end{pmatrix}$$

```

{m1, m2, m3} = DoubleGSVD[m];
Map[MatrixForm, {m1, m2, m3}]

```

$$\left\{ \begin{pmatrix} 0.966331 & 0.257303 & 0 & 0 \\ -0.257303 & 0.966331 & 0 & 0 \\ 0 & 0 & 0.700235 & 0.713912 \\ 0 & 0 & 0.713912 & -0.700235 \end{pmatrix}, \begin{pmatrix} 0.97941 & 0 & -0.201884 & 0 \\ 0 & 0.467197 & 0 & -0.884153 \\ 0.201884 & 0 & 0.97941 & 0 \\ 0 & 0.884153 & 0 & 0.467197 \end{pmatrix}, \begin{pmatrix} 0.474051 & -0.880498 & 0 & 0 \\ -0.880498 & -0.474051 & 0 & 0 \\ 0 & 0 & 0.836166 & -0.548476 \\ 0 & 0 & 0.548476 & 0.836166 \end{pmatrix} \right\}$$

```

{MatrixToQuantumCircuitDiagram[m1], MatrixToQuantumCircuitDiagram[m2], MatrixToQuantumCircuitDiagram[m3]}

```

```

circ = MatrixToQuantumCircuit[m];
QuantumCircuitToDiagram[circ]

```

-Graphics-

7. TOWARDS SIGNAL, DATA AND IMAGE PROCESSING ON A QUANTUM COMPUTER

Arguably, one of the greatest problems the field of quantum computing faces today is the relative scarcity of quantum algorithms to tackle computational problems of practical significance. To some extent this is because current quantum algorithms do not interface easily with the real world. That is, the known quantum algorithms tend to solve mathematically contrived problems rather than process real data, signals, and images. The latter requirements dominate the com-

puter-intensive applications in the real world. However, the **SynthesizePureState** algorithm can be exploited to move quantum computing in the direction of solving more practical, data-driven, computational problems, as we now explain.

There are three key steps to using quantum computers to process signals, data, and images. First, the classical data must be encoded in some quantum state in a format suitable for subsequent quantum processing. Second the computational operations must be performed on that data to prepare some quantum state containing information of interest. Finally this quantum state must be observed to extract a property of interest. The **SynthesizePureState** algorithm is immediately useful for encoding an arbitrary data set of N items in $O(\log_2 N)$ qubits.

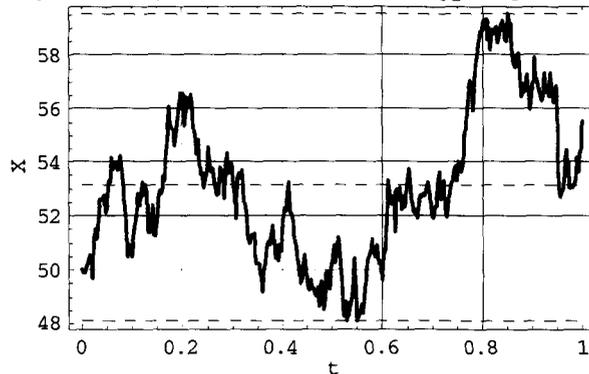


Fig. 4 An arbitrary signal of N data points can be encoded in a pure state of $O(\log_2 N)$ qubits.

All that is required is that we renormalize the data-values to ensure that the sum of their square moduli is unity. Having done so, we then simply interpret the sequence of data values as the sequence of amplitudes in the column vector representation of a pure state. It is then clear that the **SynthesizePureState** algorithm can immediately be used to synthesize a pure state encoding these (re-normalized) data values.

Currently, for N data values, the complexity of this encoding operation appears to be $O(N^2)$, but it is possible that there might be a more efficient scheme that is no worse than linear in the size of the data, i.e., $O(N)$. The better news is that the typical unitary transformations which arise in signal, image and data processing, e.g., the Fourier, Wavelet, Discrete Cosine, and Fractional Fourier transforms, are all *exponentially* more efficient on a quantum computer than a classical computer. So the additional computational cost expended in encoding a data set in a quantum state might be (depending on the application) offset by the remarkable gains in more efficient quantum processing. The deciding factor will be whether the kind of information sought from the data is accessible via *sampling* or *quantum parallelism*. This is because, although the data-encoding step is no worse than polynomial in the cardinality of the data set, and the processing step is (typically) logarithmic in the size of the data set, one is not able to “see” the complete result of the computation as one can classically. Nevertheless, one can *sample* from the final state (as in Shor’s algorithm²⁰) or one can obtain some *collective property* of the final state (as in the Deutsch-Jozsa algorithm²¹). At this time it is an open question whether there are any signal, data or image processing tasks for which such sampling or collective property extraction is sufficient to resolve a question of interest about the data.

A good candidate would be a task, such as deciding the presence or absence of a pattern within a data set, signal or image. As the information sought is a *decision* about the processed image, rather than the processed image *per se*, it is conceivable that there might be some quantum advantage, even taking account of the overhead incurred in converting the image into a quantum state prior to quantum processing. The key observation is that the quantum world strongly distinguishes “truth” from “proof”. In other words, by using a quantum computer, one might be able to obtain the “truth” about the presence or absence of a pattern within an image, but be unable to “prove” how you arrived at this conclusion.

8. CONCLUSIONS

In this paper we have introduced a general method for synthesizing any pure or mixed state defined on n -qubits. Our method is constructive, deterministic, and systematic, and can be specialized to work in one of several different hardware contexts. The method is inspired by treating state synthesis as a computational problem, first by finding a desired

unitary matrix capable of creating the desired state, and then reducing this unitary matrix to an equivalent quantum circuit. The details of our technique for mapping an arbitrary unitary matrix into an equivalent quantum circuit are described in reference 11.

ACKNOWLEDGEMENTS

This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautic and Space Administration, with funding from the National Security Agency, and the Advanced Research and Development Activity.

REFERENCES

1. E. Knill, R. Laflamme, and G. J. Milburn, "A Scheme for Efficient Quantum Computation with Linear Optics," *Nature* **409**, 46 (2001), and Franson, J.D., Donegan, M.M., Fitch, M.J., Jacobs, B.C., and Pittman, T.B., "High-fidelity Quantum Logic Operations using Linear Optical Elements," *Phys. Rev. Lett.* **89**, 137901 (2002).
2. D. Gottesman, and I. L. Chuang, "Demonstrating the Viability of Universal Quantum Computation using Teleportation and Single-Qubit Operations," *Nature* **402**, 390 (1999).
3. G. Fishman, "*Monte Carlo: Concepts, Algorithms, and Applications*," Springer-Verlag, (1996).
4. D. Aharonov, A. Kitaev, N. Nisan, "Quantum Circuits with Mixed States," in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computation (STOC)*, pages 20-30, (1997).
5. W. J. Munro, D. F. V. James, A. G. White, P. G. Kwiat, "Maximizing the Entanglement of Two Mixed Qubits," *Phys. Rev. A* **64**, 030302 (2001).
6. P. Kwiat, J. Altepeter, D. Branning, E. Jeffrey, N. Peters, and T. Wei, "Taming Entanglement," <http://xxx.lanl.gov/abs/quant-ph/0303040> (2003).
7. L. K. Grover, "Synthesis of Quantum Superpositions by Quantum Computation," *Phys. Rev. Lett.*, Volume 85, Number 6, 7th August (2000).
8. L. Grover, and T. Rudolph, "Creating Superpositions that Correspond to Efficiently Integrable Probability Distributions," <http://xxx.lanl.gov/abs/quant-ph/0208112> (2002).
9. A. Ben-Kish, B. DeMarco, V. Meyer, M. Rowe, J. Britton, W.M. Itano, B.M. Jelenkovic, C. Langer, D. Leibfried, T. Rosenband, D.J. Wineland, "Experimental Demonstration of a Technique to Generate Arbitrary Quantum Superposition States," <http://xxx.lanl.gov/abs/quant-ph/0208181> (2002).
10. P. Echternach, C. P. Williams, S. C. Dultz, S. L. Braunstein, and J. P. Dowling, "Universal Quantum Gates for Single Cooper Pair Box Based Quantum Computing," *Quantum Information and Computation* **1**, 143 (2001).
11. L. Song, and C. P. Williams, "Quantum Circuit Decomposition of an Arbitrary Unitary Operator," submitted to *Physical Review* (2003).
12. A. Barenco, A. Ekert, K.-A. Suominen, and P. Torma, *Phys. Rev. A* **54**, 139 (1996).
13. A. Fijany and C. P. Williams, "Quantum Wavelet Transforms: Fast Algorithms and Complete Circuits," *Springer-Verlag Lecture Notes in Computer Science*, Vol. **1509**, p. 10 (1999).
14. D. P. Di Vincenzo, and J. Smolin, in *Proc. Workshop on Physics and Computation*, Dallas, TX, November (1994).
15. D. P. DiVincenzo et al., *Nature* **408**, 339 (2001).
16. C. P. Williams, A. Gray, "Automated Design of Quantum Circuits," *Springer-Verlag Lecture Notes in Computer Science*, Vol. **1509**, 113 (1999).
17. M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani, *Phys. Rev. Lett.*, **73**, 58 (1994).
18. G. Cybenko, *Computing in Science and Engineering*, IEEE Computer Society, March/April, p. 27 (2001).
19. R. Tucci, [quant-ph/9902062](http://xxx.lanl.gov/abs/quant-ph/9902062) (1999).
20. P. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring," *Proc. 35th Annual Symposium on Foundations of Computer Science*, pp.124-134 (1994).
21. D. Deutsch and R. Jozsa, "Rapid Solution of Problems by Quantum Computation," *Proc. Royal Soc. London, Series A*, Vol. **439**, pp.553-558 (1992).