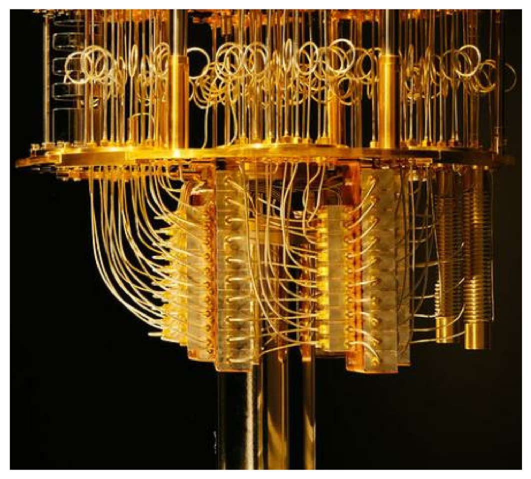
Compiling Quantum Programs

Li-Heng Henry Chang

Advisors: Paul Cadden-Zimansky and Sven Anderson

Computer Science & Physics Programs, Bard College, May 2023



The precise description of the world requires quantum mechanics. However, present computers run classically, and cannot simulate quantum systems efficiently. But quantum computers can! Quantum compiling bridges the development of quantum computer software and hardware. The problem of quantum compiling is converting high-level quantum programs into low-level hardware executable code. Because quantum systems are subjected to noise and decoherence, the challenge of quantum compiling is tied to the optimization of program runtimes and the lengths of compiled sequences. We review universal quantum computation and the Solovay-Kitaev theorem, develop compiling schemes inspired by each idea, and analyze their results.

What is a Qubit?

The smallest unit of information to keep track of a quantum state is a "quantum bit" or "qubit". Two physically distinguishable states are mapped to the computational basis states $|0\rangle$ and $|1\rangle$

$$|0\rangle \to \begin{bmatrix}1\\0\end{bmatrix}, |1\rangle \to \begin{bmatrix}0\\1\end{bmatrix}.$$

Mathematically, a single-qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ can be described by two complex numbers such that they are normalized $|\alpha|^2 + |\beta|^2 = 1$.

$$|\psi\rangle \to \alpha\begin{bmatrix}1\\0\end{bmatrix} + \beta\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}\alpha\\\beta\end{bmatrix}$$

The amplitudes squared $|\alpha|^2$ and $|\beta|^2$ are the probabilities of obtaining the $|0\rangle$ and $|1\rangle$ states, so $|\alpha|^2 + |\beta|^2 = 1$. This is called the normalization condition.

Quantum Circuit & Algorithm

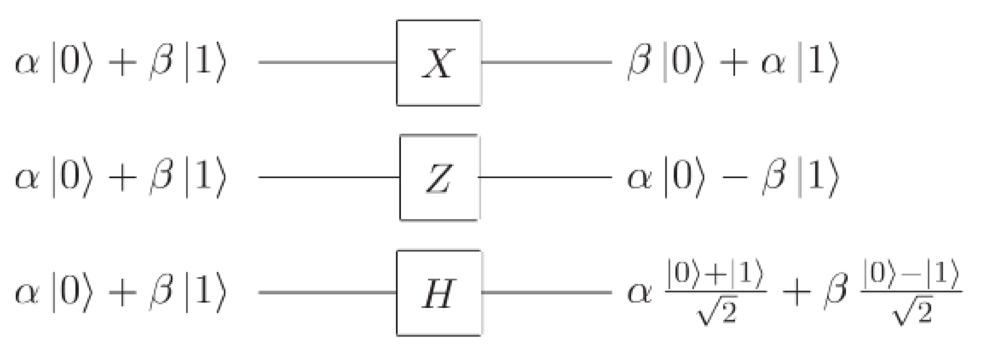
How do we describe the evolution U of a state over time $|\psi'\rangle = U|\psi\rangle$? The normalization condition says that the operation U has to be *unitary* $U^\dagger U = I$. A general single-qubit operation is a gate described by a 2x2 unitary matrix

$$U(\theta,\phi,\lambda) = \begin{bmatrix}\cos\theta e^{i\lambda} & \sin\theta e^{i\mu}\\ -\sin\theta e^{-i\mu} & \cos\theta e^{-i\lambda}\end{bmatrix}.$$

Some notable single-qubit gates,

$$x = \begin{bmatrix}0&1\\1&0\end{bmatrix}, z = \begin{bmatrix}1&0\\0&-1\end{bmatrix}, H = \frac{1}{\sqrt{2}}\begin{bmatrix}1&1\\1&-1\end{bmatrix}, T = \begin{bmatrix}1&0\\0&\exp\left(\frac{i\pi}{4}\right)\end{bmatrix},$$

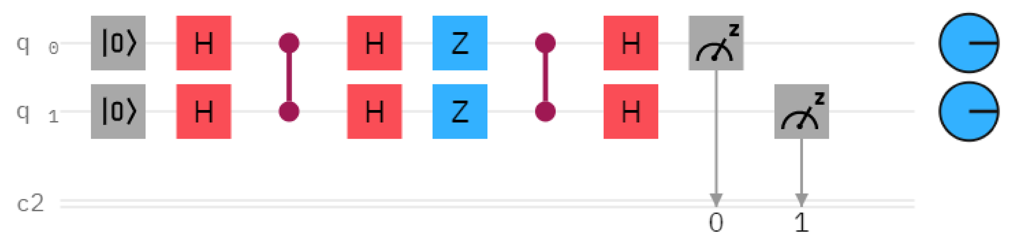
and their circuit representation,



In general, a quantum algorithm is a n-qubit operation that can be represented as a $2^n \times 2^n$ unitary matrix

$$U = \begin{bmatrix}\dots&\dots&\dots&\dots&\dots&\dots\\\dots&\dots&\dots&\dots&\dots&\dots\end{bmatrix},$$

or in terms of a circuit diagram (Grover's algorithm).



Compilation Problem

A quantum algorithm is a unitary transform U on a quantum state $|\psi\rangle$ that produces a computed state $|\psi'\rangle = U|\psi\rangle$. However, we are only given a small discrete set of gates $\mathcal{G}$ by physical hardware. How can we realize U using $\mathcal{G}$, with a small error ϵ? For instance, U is CCNOT operation (left in fig.), and $\mathcal{G} = \{CNOT, H, T\}$?

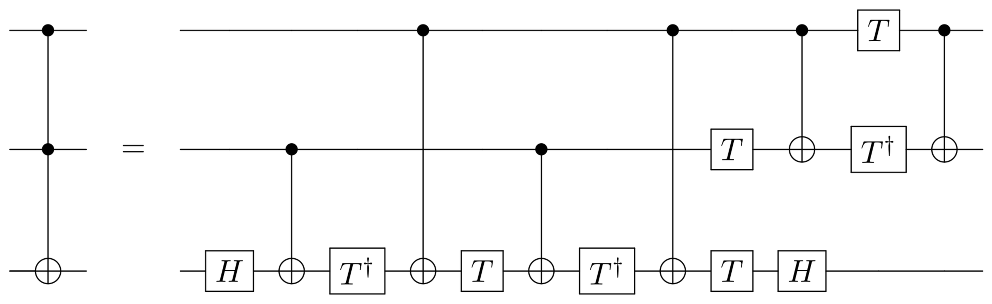


Figure: CCNOT gate compiled into H and T gates.

Universal Gates

A set of gates is **universal** for quantum computation if any unitary matrix can be approximated to arbitrary accuracy by a quantum circuit built with only those gates. *Single-qubit and CNOT gates are universal.*

$$\text{CNOT} = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$

Universal Decomposition

We want to express an algorithm U in terms of 2-level unitaries (each is some CNOT plus single-qubit gates)

$$U = \prod_i A_i, \text{ where } A_i \text{ are 2-level unitaries.}$$

Reduce one element at a time $UA_0 = U_1 \Rightarrow$ solve for a 2x2 unitary matrix $A'_0(\theta,\phi,\lambda)$:

$$\begin{bmatrix}\dots & a & b & 0 & \dots & 0\\ \dots&\dots&\dots&\dots&\dots&\dots\end{bmatrix} A_0 = \begin{bmatrix}\dots & c & 0 & 0 & \dots & 0\\ \dots&\dots&\dots&\dots&\dots&\dots\end{bmatrix} \Rightarrow \begin{bmatrix}a & b\end{bmatrix} A'_0 = \begin{bmatrix}c & 0\end{bmatrix}$$

Reduce rows until left with 2x2:

$$\begin{bmatrix}*&*&*&*\\*&*&*&*\\*&*&*&*\\*&*&*&*\end{bmatrix} \to \begin{bmatrix}1&0&0&0\\0&*&*&*\\0&*&*&*\\0&*&*&*\end{bmatrix} \to \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&*&*\\0&0&*&*\end{bmatrix}$$

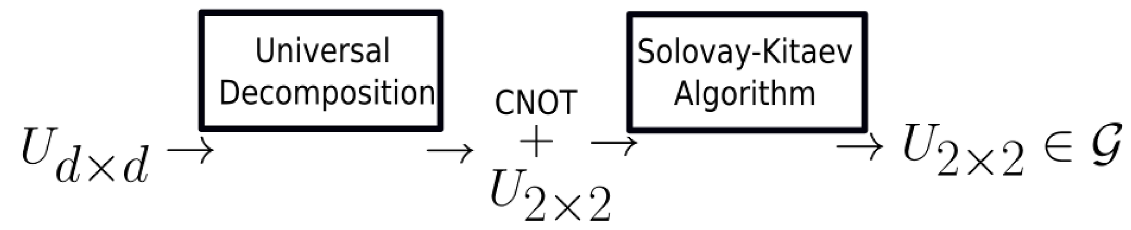
This procedure breaks a d-dimensional U into $\approx d^2 = (2^n)^2 = 4^n$ single-qubit operations A_i.

Solovay-Kitaev Algorithm

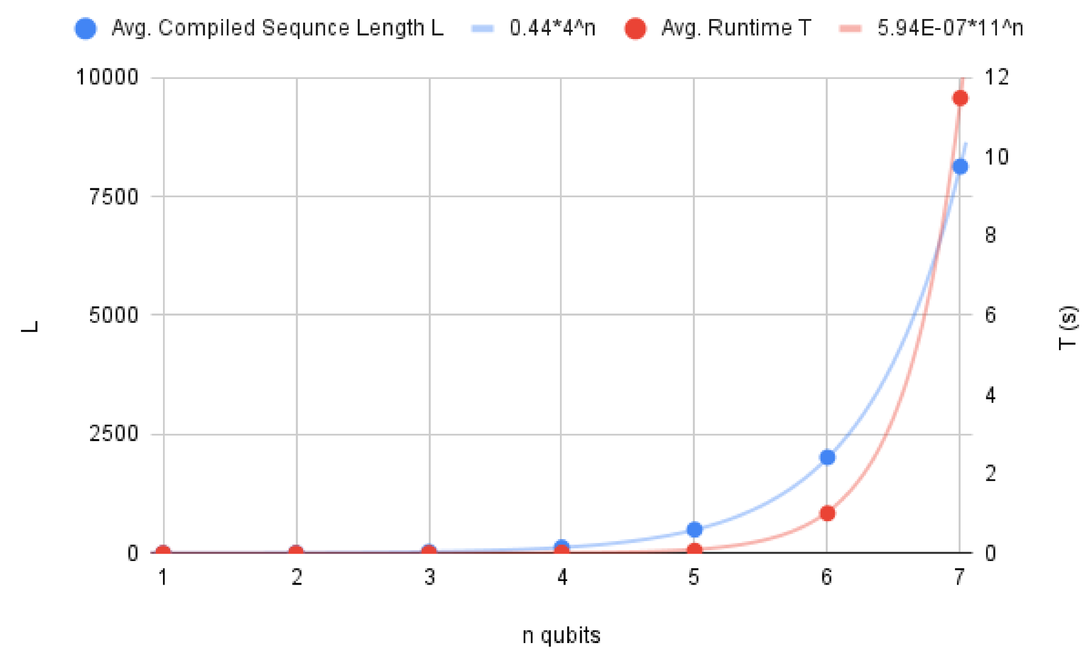
Solovay-Kitaev theorem guarantees a compiled sequence efficient in terms of error. A corresponding algorithm inspired by the theorem can approximate single-qubit unitaries U_{2x2} to any error using hardware gates $\mathcal{G}$. The algorithm starts with an initial approximation of length l_0 and error ϵ_0, then recursively increasing the length by five times $l_1 \to 5l_0$ and reducing the error exponentially $\epsilon_1 \to \approx \epsilon_0^{\frac{3}{2}}$ until the error threshold is met.

Compiler Design

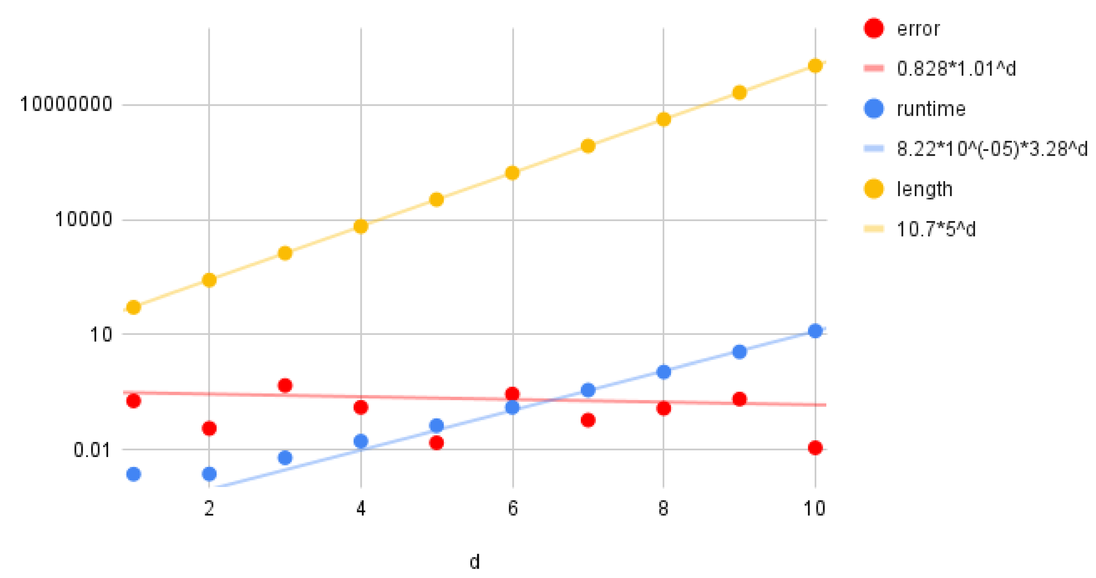
Universal Decomposition turns an algorithm U into a product of single-qubit plus CNOT gates. The Solovay-Kitaev algorithm converts any single-qubit gates into a sequence of gates from hardware gate set $\mathcal{G}$. Combining Universal Decomposition and Solovay-Kitaev algorithm we have a compiler!



Implementation & Analysis



The Average Compiled Sequence Length and Average Runtime of Universal Decomposition versus the number of qubits n that defines the size of the unitary operator state space. Dots indicate data and curves the best fit lines.



The log plot of average error ϵ_d, runtime t_d, and output compiled sequence length l_d over the recursive depth d of the Solovay-Kitaev algorithm.

References

The thesis, code base, and references can be accessed at https://github.com/Hazarre/quantum-compilation.