## Quantum Computing for Programmers

This introduction to quantum computing from a classical programmer's perspective is meant for students and practitioners alike. More than 25 fundamental algorithms are explained with full mathematical derivations and classical code for simulation, using an open-source code base developed from the ground up in Python and C++.

After presenting the basics of quantum computing, the author focuses on algorithms and the infrastructure to simulate them efficiently, beginning with quantum teleportation, superdense coding, Bernstein–Vazirani's algorithm, and Deutsc–Jozsa's algorithm. Coverage of advanced algorithms includes the quantum supremacy experiment, quantum Fourier transform, phase estimation, Shor's algorithm, Grover's algorithm with quantum counting and amplitude amplification, quantum random walks, and the Solovay–Kitaev algorithm for gate approximation. Quantum simulation is explored with the variational quantum eigensolver, quantum approximate optimization, and the NP-complete Max-Cut and Subset-Sum algorithms.

The book also discusses issues around programmer productivity, quantum noise, error correction, and challenges for quantum programming languages, compilers, and tools, with a final section on compiler techniques for transpilation.

**Robert Hundt** is a Distinguished Engineer at Google, where he led software development for Google's TPU supercomputers, the XLA compiler for TensorFlow, an open-source CUDA compiler, and currently the high-level synthesis toolchain XLS. He has more than 25 scientific publications, holds more than 35 patents, and is a senior member of IEEE.

# Quantum Computing for Programmers

ROBERT HUNDT

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre,
New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781009098175
DOI: 10.1017/9781009099974



First published 2022

Printed in the United Kingdom by TJ Books Limited, Padstow Cornwall

*A catalogue record for this publication is available from the British Library.*

*Library of Congress Cataloging-in-Publication Data*
Names: Hundt, Robert, author.
Title: Quantum computing for programmers / Robert Hundt.
Description: Cambridge, United Kingdom ; New York, NY : Cambridge University Press, 2022. | Includes bibliographical references and index.
Identifiers: LCCN 2021044761 (print) | LCCN 2021044762 (ebook) | ISBN 9781009098175 (hardback) | ISBN 9781009099974 (epub)
Subjects: LCSH: Quantum computing. | BISAC: COMPUTERS / General
Classification: LCC QA76.889 .H86 2022 (print) | LCC QA76.889 (ebook) | DDC 006.3/843–dc23/eng/20211105
LC record available at https://lccn.loc.gov/2021044761
LC ebook record available at https://lccn.loc.gov/2021044762

ISBN 978-1-009-09817-5 Hardback

To Mary, Thalia, and Johannes

# Contents

# Acknowledgments

A book like this would not be possible without the help of many people. Vincent Russo and Timofey Golubev found a large number of issues with the mathematical formulation, code, and writing. Gabriel Hannon provided valuable pointers to related physics concepts. Several of my questions were answered on the Quantum Computing Stack Exchange, a very helpful resource and community. Wes Cowley and Sarah Schedler provided line editing, Sue Klefstad produced the impressive index, and Eleanor Bolton provided outstanding copyediting services. I am grateful to Tiago Leao for pointing me to Beauregard (2003), which was key for my implementation of Shor's refactoring. Together with Rui Maia, Tiago also provides the community with a much-appreciated reference implementation (Leao, 2021).

I also want to thank many of my colleagues at Google. Dave Bennet and Michael Dorner went through the first draft of this text, which, I am certain, was not a pleasant experience. Their feedback helped to shape this work into an actual learning resource. Two of my colleagues stand out for their rigor and obsession with details: Fedor Kostritsa, who provided most detailed comments on text, math, derivations, and code; and Ton Kalker, who diligently reviewed the whole text and helped greatly in sharpening the mathematical formulation. Sergio Boixo and Benjamin Villalonga corrected many of my misunderstandings about the quantum supremacy experiment. Michael Broughton and Craig Gidney helped to improve the section on Grover's algorithm. Craig also maintains the elegant online simulator Quirk. Thanks also to Mark Heffernan, Chris Leary, Rob Springer, and Mirko Rossini. Finally, I owe gratitude to Aamer Mahmood for his extraordinary support.

Without exception, my contacts at Cambridge University Press were outstanding. First and foremost, I must thank my editor, Lauren Cowles, who did a tremendous job guiding me throughout the whole process.

Finally, and most importantly, I am incredibly thankful to my family, including my dog, Charlie, for their love, patience, and support during this all-consuming effort.

 

# Introduction

I think I can safely say that nobody understands quantum mechanics

Feynman (1965)

I have been impressed by numerous instances of mathematical theories that are really about particular algorithms; these theories are typically formulated in mathematical terms that are much more cumbersome and less natural than the equivalent formulation today's computer scientists would use

Knuth (1974)

This book is an introduction to quantum computing from the perspective of a classical programmer. Most concepts are explained with code, based on the insight that much of the complicated-looking math typically found in quantum computing may look quite simple in code. For many programmers, reading code is faster than reading complex math notation. Coding also allows experimentation, which helps in building intuition and understanding of the fundamental mechanisms of quantum computing. I believe that this approach will make it efficient and fun to get started.

Contrary to other learning resources, we will not use available software frameworks in this book, such as the well-developed Qiskit toolkit from IBM or Google's Cirq. Instead we build our own infrastructure from the ground up, based initially on Python's `numpy` library. It turns out that for learning the fundamentals, only a few hundred lines of code are required. This initial code is slow, but explicit. It is easy to debug and experiment with, which makes it an excellent learning vehicle.

We then improve the infrastructure, accelerate it with C++, and detail an elegant, sparse representation. We introduce basic compiler concepts that allow transpilation of our circuits to those other platforms – Qiskit, Cirq, and others. This enables the use of these systems' advanced features, such as scale-out performance and advanced error models.

Typically, an introduction to quantum computing is prefaced by a sizeable reintroduction of complex linear algebra. We will not follow this pattern here. Many programmers do have a solid foundation in linear algebra, but others lack the background or interest in this topic. It is my goal to be an attractive learning resource for both groups, without getting into the details of linear algebra. Hence I only assume basic familiarity with complex numbers, vectors, and matrices. We review a handful of core concepts in Chapter 1. As we go along in the text we add a small number of additional mathematical concepts that are necessary to understand the algorithms. We hope that this format will be helpful to the linear algebra-challenged while not being too shallow for the cognoscenti. After the introduction of the basic mathematical concepts, the book is organized into the following major sections:

- In Chapter 2 we introduce core concepts of quantum computing and implement them with full matrices and vectors in Python. We discuss states, operators, entanglement, and measurement. We show a variety of ways to construct, describe, and analyze qubits and quantum circuits. Quantum mechanics, superposition, entanglement, and measurement are all complex and deeply philosophical topics. In this text, however, we focus exclusively on the computational aspects of the theory.
- This is followed in Chapter 3 with introductory algorithms, utilizing the infrastructure developed so far. The material is presented in an elaborate way, with detailed mathematical derivations.
- The basic infrastructure developed up to this point does not scale to a higher number of qubits, which are needed for complex algorithms. In Chapter 4, we detail techniques for faster gate application and accelerate with C++. We demonstrate how a sparse representation can produce the best performing results for a certain class of algorithms. These sections lead to building a high-performance quantum simulator. Readers not interested in infrastructure may skim or even skip this section.
- In Chapter 5 we convince ourselves that quantum computers indeed have capabilities that go *beyond classical* machines.
- With this insight, Chapter 6 on complex algorithms provides important details on several of quantum computing's core algorithms, including Grover's search, the quantum Fourier transform, phase estimation, quantum random walks, Shor's algorithm for integer factorization, and a variational quantum eigensolver with some applications. This section also details the seminal Solovay–Kitaev algorithm to approximate any unitary gate from a small universal gate set. The foundation built up in the earlier chapters is sufficient to implement and fully appreciate these marvelous algorithms.
- Chapters 7 and 8 then address practical issues around programmer productivity. We touch on quantum error correction, which is essential to the feasibility of quantum computing. We also discuss quantum programming language design, compilers, and tools to further improve programmer productivity.
- The Appendix contains additional interesting material that did not fit the main text's flow. Specifically, it contains a detailed description of the sparse simulation infrastructure.

## Source Code

Much of the content in this book is explained with both math and code. To avoid turning this book into a giant code listing, however, we abbreviate less interesting or repetitive pieces of code with constructs such as `[...]`. Scaffolding code, such as Python `import` statements or C++ `#include` directives, are typically omitted. The full sources are hosted under a permissive Apache license on GitHub, along with instructions on how to download, build, and run:

```
https://github.com/qcc4cp/qcc
```

Contributions, comments, and suggestions are always welcome. Typesetting the code may have introduced errors, but the source of truth is the online repository. The code is also likely to have evolved beyond what is published here.