Towards Distributed Quantum Algorithms

Pablo Andres-Martinez

Master of Science by Research

Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh

2018

Abstract

Acknowledgements

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Pablo Andres-Martinez*)

Table of Contents

Chapter 1

Quantum Computing: A brief overview

In this chapter... **TODO**

1.1 What is Quantum Computing?

Quantum computing aims to take advantage of quantum mechanics to speed-up computations. There is strong theoretical evidence that quantum computers are capable of solving some problems substantially faster than standard (classical) computers. Well known examples are:

- Shor's algorithm for *polynomial-time* large number factorisation **?**. There is no known algorithm on classical computers that can perform this task in polynomial time, and it is suspected not to be possible[1]. Quantum computers theoretically provide an exponential speed-up on this task.

- Grover's algorithm for efficient unstructured search **?**. The algorithm performs a brute-force search (i.e. requiring no knowledge about the search space) over a N data-points in time proportional to $\sqrt{N}$. A brute-force search in a classical computer should always take time proportional to N.

Besides, in May of the present year, Raz and Tal **?** gave formal proof of the existence of problems that a classical computer may never solve in polynomial time, but are solvable in polynomial time on a quantum one. Nevertheless, all of these results are theoretical in nature and there are some caveats on their practical implementation, discussed in § 1.2.1. Giving experimental evidence of this time-efficiency separation

[1] RSA, a widely used encryption system, relies its security on the assumption that factorisation of large numbers can not be computed efficiently.

between quantum and classical computers is a highly active area of research, known as *quantum supremacy* **?**. The general opinion of the scientific community is that we will eventually overcome these caveats, and take advantage of quantum computing in fields of science such as:

- Chemistry: **TODO**
- Machine learning: **TODO**
- Engineering: **TODO**

For these applications, we will require large scale quantum computers. Due to the obstacles in the way of building a large mainframe quantum computer (see § 1.2.1), some authors have advocated the alternative of building a grid of smaller quantum computer units that cooperate in performing an overall computation **?**. In this work, particularly in Chapter 2, we contribute to this perspective, providing a method for efficiently distributing any quantum program originally designed for a monolithic quantum computer.

1.1.1 The principles of Quantum Computing

The advantage of using quantum mechanics to perform computations is usually traced down to the following three principles:

- *Superposition*: In classical computing, the unit of information is the *bit*, which may take one of two values: 0 or 1. In quantum computing, the *bit*'s counterpart is the *qubit*, whose value may be *any linear combination* of 0 and 1, known as a *superposition*, and usually written as:

  $$|qubit\rangle = \alpha|0\rangle + \beta|1\rangle$$

  where α and β are complex numbers that must satisfy: $\alpha^2 + \beta^2 = 1$.

  A popular analogy of a qubit's superposition is a coin spinning[2]: the classical states (0 and 1) are *heads* and *tails*, but when the coin is spinning, its state is neither of them. If we knew exactly how the coin was set spinning, we would be able to describe the probability distribution of seeing heads or tails when it

[2]Note this is just an analogy, and while a coin spinning can be perfectly modelled using classical physics, a qubit can not. In fact, superposition is key in the (even weirder) two other principles that differentiate quantum and classical computing.

stops; these would be our α^2 and β^2 values. We may *measure* a qubit, and doing so corresponds in our analogy to stopping the coin: we will get either 0 or 1 as outcome.

The essential aspect of this analogy is that, before measurement, the *qubit*'s state is neither 0 nor 1. Through certain operations (that would correspond to altering the axis of spin of the coin), we may change the coefficients α and β of the superposition. Interestingly, in quantum computing, we encode input and read output (after measurement) as standard classical binary strings, and thus, *for input/output we use as many qubits as bits would be required*. What superposition provides is the ability to – during mid-computation – maintain a superposition of all potential solutions to the problem, and update all of them simultaneously with a single operation to the qubits. In some sense, superposition allows us to explore multiple choices/paths of the computation, using only the resources required to explore a single one of those paths.

- *Interference*: Superposition gives us the ability to simultaneously explore different paths to solve a problem. However, in the end we will need to measure the qubits – stop the coins, in order to read heads or tails – and the result will be intrinsically random. For quantum computing to be any better than a probabilistic classical computer, we require the ability to prune the paths that have led to a dead-end. This is precisely what *interference* provides: some operations on the qubits may make different classical states in the superposition cancel each other out. Interference is at the core of any speed-up achieved by a quantum algorithm, and taking advantage of it is the main challenge when designing quantum algorithms.

- *Entanglement*: Quantum mechanics allows us to have a pair of qubits a and b in a superposition such as:

$$|a,b\rangle = \frac{1}{\sqrt{2}}|0,0\rangle + \frac{1}{\sqrt{2}}|1,1\rangle$$

This implies that, when we measure the qubits, we may only read $a = 0, b = 0$ or $a = 1, b = 1$ as outcome (the coefficients for $|0,1\rangle$ and $|1,0\rangle$) are both 0). Then, what happens if we only measure a? In this case, we would also know b's outcome, without measuring it. More surprisingly, if we measured both a and b at the same instant, we would obtain $a = 0, b = 0$ half of the times and $a = 1, b = 1$ the other half. In short, it seems like acting on one qubit has an instantaneous

effect on the other. Whenever a group of qubits exhibits this property, we say they are *entangled*. Entanglement holds regardless how far apart a is from b; for instance, they could be on two different quantum processing units of a distributed grid. Indeed, entanglement will be key in our discussion of distributed quantum algorithms, and we explain how to use it to perform non-local operations in § 2.0.1.

1.2 Current hardware implementations

1.2.1 Caveats

Caveats:

number of qubits decoherence: the coins can not spin forever. however, superposition is an essential building block for quantum computing! error correction use of oracles

1.3 Programming on Quantum Computers

1.4 Summary

Chapter 2

Towards Distributed Quantum Algorithms

2.0.1 Non-local operations