Group 1: Review of: Classical Verification of Quantum Computations in Linear Time

Justin Garrigus
Department of Computer Science and Engineering
University of North Texas
justingarrigus@my.unt.edu

***Abstract*—Quantum computing is a new area of computer science rapidly developing in real time. The fundamentals of quantum mechanics have been known since the early 1900s and the properties of atomic-scale particles have been essential to the development of classical computers, but only recently have non-theoretical quantum computers been developed. As such, the field of quantum computing is forming into a tangible subset of computer science which is capable of solving practical problems useful in the real-world. Although, the programming paradigm that quantum computers present is inherently different from that of classical computers as they introduce fundamentally random but correlated values, most notably with the ability to explore all possibilities of a combinatorial problem at once before being collapsed to a single possibility when measured.**

Quantum computation has a rich history that combines a wide and diverse set of domains including physics, mathematics, material science, engineering, computer science, cryptography, and more. This report will summarize and introduce a beginner's perspective to quantum computing and quantum mechanics. Additionally, it will describe the problem of quantum verification in a world with large-scale quantum servers, and a population with the desire to tap into that computational power without wanting the server to know or lie about the specific computation being performed. One paper addresses this problem by proposing a set of methods that runs in linear time compared to alternative approaches that all run in cubic time.

***Index Terms*—quantum computing, verification, theoretical computer science**

I. Introduction

Quantum computing exists at the intersection of many different research areas. While it is of practical interest to computer scientists to utilize the computational power to solve real-world problems, it is important for other researchers to learn how to even build a quantum computer that can maintain stability for more than a few microseconds or that can hold more than a few hundred bits of data. Due to the relative novelty of real-world quantum computation and the explosive popularity it is gaining in the last few years, it is important for all computer science and engineering researchers to understand the basics of the domain to see if their own field can benefit from its usage.

The field of quantum mechanics began with the discovery that both the momentum and the position of a particle cannot be known with absolute certainty, which came to be known as the Heisenberg uncertainty principle. This was coupled with an equation formulated by Erwin Schrödinger that describes the properties of a wave function, introducing the idea that particles could be inherently probabilistic and impossible to predict every quality of with perfect accuracy. Furthermore, it almost seemed that the state of a particle was determined by whether or not an observer was viewing it.

Fig. 1. The Einstein-Podolsky-Rosen paradox, in which a particle splits exactly down the center to form two identical sub-particles. EPR posits the Heisenberg uncertainty principle is false.

These ideas were in direct contrast with the "typical" notion of science that physicists were motivated to find: that the world consists of predictable quanta, and that all qualities of the world are fixed and expressible with equations regardless of the presence of any scientist watching them. The biggest proponent of this belief was Albert Einstein, and he never fully accepted the quantum reality despite inadvertently being one of its pioneers.

Albert Einstein was a believer in multiple popular beliefs including *hidden variable theory* (that the perceived "randomness" particles exhibit is actually due to a non-random–but hidden–variable) and *local realism* (that an object cannot instantly affect a different far-away object). He proposed several thought experiments that attempt to quantify his belief, most notably that of the Einstein-Podolsky-Rosen paradox shown in figure 1. The EPR paradox suggests that both the position and the momentum of a particle can be known with absolute certainty; if a particle splits down the middle, then measurement of the position of one sub-particle and momentum of the other sub-particle can reveal the position and momentum of both due to the symmetry of movement [5].

On its own, the EPR paradox presents a strong argument that particles *do* possess both a position and a momentum, but Einstein's rival at the time presented a new idea. Niels Bohr was the leader of the Copenhagen interpretation of quantum mechanics–which is the leading counterargument to local realism–and suggests that particles *cooperate* despite being separated. In this case, the two particles in the EPR paradox would work together to obfuscate their properties; when one particle is measured, the properties of the other particle are fundamentally altered in such a way that maintains

the Heisenberg uncertainty principle. Einstein referred to this as "spooky action at a distance" due to the claim that this influence appears to move instantaneously, which he previously proved was impossible due to general relativity forbidding faster-than-light travel.[1]

These opposing claims led to a standstill until 1964, in which a testable experiment was proposed which could confirm either local realism or the Copenhagen interpretation as true. This experiment, named a "Bell test", is a simple logical inequality that measures if two variables are correlated. The first loophole-free[2] Bell tests were executed by three independent labs in 2015, and they collectively confirmed that neither the hidden-variable theory nor local realism were accurate at describing how the world worked.

The result of these observations is a new set of terminology that builds on quantum mechanics, but is robust enough to be used for constructing quantum computing primitives discussed in this paper. First and foremost is the concept of *entanglement*, which is the correlation of two particles.[3] The EPR paradox shown in figure 1 depicts the creation of two entangled particles, but other methods of entanglement include forcing an atom to emit photons of correlated polarity. The benefit of entanglement is that the state of two separate objects are related, but no physical value is shared between them. In other words, the particles do not keep track of a *state* in the way classical particles do; large objects may store information such as "color", which will exist regardless of the number of observers monitoring the object, but quantum particles do not store this information naturally, and the state of one of a set of correlated objects is generated[4] as soon as an observation of one of the objects is made.

The second important quality of quantum mechanics is *superposition*, which is the combination of multiple states into one. With superposition, we can describe a particle as existing within different sets of states. The probability of observing one of these sets of states is expressed with a wave function like in Schrödinger's equation. When an object in a superposed state is observed, its wave function "collapses" and it is forced to "decide" which of the possible states to take on. To these ends, the particle loses its superposed and entangled properties at the time of measurement.

The job of the quantum computer is thus the following [6]:

1) Take a collection of particles and create a superposition of binary states. A set of n particles can therefore take on 2^n possible states at once.
2) Entangle the particles in such a way so only certain combinations of states are possible.
3) Use quantum gates that modify the states *without observing them*.[5]
4) Adjust the set of states such that some "favorable" states have a high probability of being "chosen" when the particles are observed [4].

The rest of this paper will describe the problem of quantum verification and a paper [1] which attempts a solution to the problem. Section II describes what motivates the use of verification methods and the necessary background for understanding the paper. Section III gives a high-level description of the method. Section IV shows the difference in the proposed method in comparison to previous work. Section V summarizes the the paper and discusses future work.

II. Motivation and Related Works

Currently, the benefit of quantum computation is still mostly theoretical due to the physical limitations of maintaining a coherent state between lots of particles for long periods of time. Popular quantum computing algorithms like Shor's algorithm [3] have practical applications to today's computing problems, but require thousands of logical bits of quantum states. Although, while physicists and engineers research ways to develop the hardware, theoretical computer scientists can work on creating abstractions that will eventually utilize the hardware with peak efficiency.

This paper revolves around a scenario where large quantum computing servers exist with thousands or millions of quantum bits, but clients are not capable of maintaining more than a small quantum "gadget". This may reflect a realistic scenario in the future since some types of quantum computers require temperatures close to absolute zero, which is not yet known to be practical for everyday users to keep on their own, but a single quantum device of limited size can be possible in its place.

The goal is for the client to utilize the benefit of the server without (1) having the server know what computation is being performed and (2) without the server being able to lie about the result of the quantum circuit. A malicious server is one which peeks or gains information on the user's data or even attempts to generate random classical data to attempt to pass it off on being from a genuine quantum computer.

The first issue is typically impossible for classical computers; for most problems, there is no way to employ a computer to execute some program without knowing about the program being run. Although, quantum data has the benefit of becoming disentangled when observed, so the client can learn when the server has made some observation on their data. This detail makes it possible for a quantum server to calculate sensitive

[1]Bohr opposed this rebuttal by asserting the particles *cooperate*, not *communicate*, so nothing is physically travelling between them even though they seem to "know" when the other particle was measured.

[2]Designing a "loophole-free" test is difficult in actuality; scientists have proposed dozens of different methods that two particles could "secretly" communicate with each other or otherwise maintain hidden variables, but these are disprovable with the help of randomness, precise timing, and large distances.

[3]The term *particle* is another abstract term, as genuine quantum computers can represent entanglement through photons, electrons, atoms, or entire molecules.

[4]Or rather the state "always existed but was undefined before the point of measurement." Remember this does not indicate a hidden variable, as the Bell test confirmed.

[5]This step is particularly difficult because an "observation" can be as simple as one particle hitting another and dissipating heat as friction from the collision. Observations are any actions that leave a trace, which is why large-scale quantum computers are incredibly difficult to create due to the scalability issue that these observations create.

information like private keys for the client without having ways of learning what those keys are. Additionally, the client has an additional ability of forcing the server to forget some information: since an observation "collapses" the state of a particle, the client can require the server to read a value in order to prevent them from using its entangled properties in the future.

Previous works on this method include multiparty quantum computation, obfuscation, and zero-knowledge approaches, but each of these run in cubic time relative to the size of the circuit due to the high error rate that these methods introduce, requiring repeated resampling of the circuit to ensure the error rate is acceptable. If the user has L gadgets, the error introduced in one of the gadgets is $1/L$, so quadratic rounds of sampling would be used to mitigate this. Additionally, each quadratic round would be performed on a linear execution of each quantum gate, which leads to a cubic time complexity in total, or $O(\text{poly}(\kappa)|C|^3)$ relative to the size of the quantum circuit $|C|$.

Before continuing, it is important to note the κ term in the time complexity. This indicates a "security factor" that is not indicative of the algorithm being used, and is instead a popular term applied to all cryptography-related algorithms. An example of a κ term may be the size of the key or number of primality iterations in RSA encryption, which guarantees a higher level of security at the expense of more computations to perform.

III. Methodology

The proposed method changes the time complexity of previous work from cubic time, $O(\text{poly}(\kappa)|C|^3)$, to linear time, $O(\text{poly}(\kappa)|C|)$. The crucial modifications it makes is to reduce the error rate of the circuit-sampling method through the help of better quantum value test methods.

To begin, the method utilies "noisy trapdoor claw-free functions" (NTCFs) to act as a generic key-distribution method. These functions are "2-to-1" in the sense that computation is easy in one direction but difficult through the other, but a "trapdoor" within the function allows the client to give the ability to the server to easily reverse the direction of the computation. A common NTCF is asymmetric encryption which transforms a message into an encrypted form with a public key that is difficult to reverse without an associated private key. In the context of this paper, the NTCF allows both the client and server to evaluate this function with the server getting a superposition of two states and the client getting the individual states (therefore, the server could measure their state to get exactly one outcome, or only 50% of the client's state).

The main theory around the method is based on Remote State Preparation and Verifiability (RSPV): the client wants to send some state to the server and confirm the server really set the state. The server could technically fake the assignment of quantum variables, but not without incurring a polynomial-time cost in performance [2]. The client can verify the state was set by (1) requiring the server to perform a partial measurement of their state, (2) calculating a value θ from that measurement and from secret information held by the client, and having the server measure a quantum bit in a basis (measurement scheme) related to θ.[6] A server which does not actually initialize the state would give a measurement that is unlikely to match the client's expected value.

The client can employ a "switch gadget technique" similar to the method described in section II to reveal private keys to the server without sacrificing secrecy by forcing the server to forget those keys after some critical computation was performed. The keys are used to parse "phase tables"[7] (mappings of encoded to decoded states), so a memory-wipe of these keys would lead to the phase tables being incomprehensible.

The final component of this process is the verification tests on the server's state by the client. There are three tests that are randomly cycled between for each gate in the quantum circuit:

- Standard basis test: the client requests a vector from the server and the server returns the vector measured from the NTCF in the standard basis.
- Individual basis test: the client reveals the value of a state to the server, to which the server performs a Hadamard gate–a special operation to reset the value of a quantum particle–and the client measures their gadget for a value.
- Collective phase test: a set of gadgets are prepared in parallel so they can compute a pair of values for the phase table before comparing them and resetting their values with the Hadamard gate.

IV. Results

With each of the component parts given previously, the algorithmic design of the paper can be summarized with the following code:

```
for-each gate in circuit:
    client->encode state
    client->send state to server
    server->choose and run random test function
    server->return result to client
    server->execute gate
```

Listing 1. Simple buffer overflow attacks

If the server makes any observation of the data, it would collapse the wave function and destroy the entanglement that was initially generated when the state was sent. This would lead a subsequent test to give a result that is unexpected, which the client can easily detect.

This method improves the runtime of previous approaches from cubic time to linear time due to the encoded state representation in both NTCFs and in the phase table along with the tests with improved accuracy. Typical quantum applications currently consist of more than tens of thousands of circuits [1] which makes a cubic solution inconvenient for a server to

[6] A difficulty of this paper is that it assumes a thorough familiarity with quantum computation, so some aspects of this report are simplified for pedagogical purposes.

[7] These phase tables utilize the "random oracle model", a special hashing scheme that translates one-to-one an input to a new output with no overlapping in outputs for any given unique input.

reasonably evaluate. The compromise of validity-checking is only valid if the overhead presented in the design is negligible compared to the complexity of the server, which is true for this paper. In any case, the runtime of the design is currently dominated by the security parameter κ rather than the tests or evaluation of individual quantum gates.

Future work should expand on this area by possibly reducing the number of times the security parameter is involved in computations. For example, the code given in listing 1 shows that the client must encrypt every state that the server operates on, and the server must return the calculated value for every gate. A possible improved solution might combine multiple gates together before communicating their results en-mass, but this presents a more difficult verification problem for the client who has no access to a quantum computer of their own. Additional improvements may also reconsider the "gadget" design by giving the user a reduced number of quantum devices or intentionally limiting the power of each quantum device further to more accurately reflect the future.

V. Conclusion

This paper gave a brief introduction to the field of quantum computing and described a paper that solves the quantum verification problem in linear time compared to other solutions which take cubic time. Quantum computation relies on the entanglement (correlation-at-a-distance) and superposition (combination of multiple distinct states) of particles to allow the mass-computation of many different states at once. Quantum verification is a method of both preventing a server from learning of the data of a program and for preventing a server from lying about computing the result of a quantum circuit, which is both made possible by the unique properties of quantum particles including the collapse of wave-functions on observation of any particle. Future work can expand the paper's method by finding ways to verify multiple states at once while maintaining the same error or detection rate by the client.

References

[1] Jiayu Zhang, "Classical verification of quantum computations in linear time," In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science* (FOCS), pages 46–57, 2022.

[2] Y. Zhou, E. M. Stoudenmire, and X. Waintal, "What limits the simulation of quantum computers?" in *Physical Review X*, 2020.

[3] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," in *SIAM Journal of Computing*, 1997, pp. 1484–1509.

[4] A. Berthiaume, "Quantum computation", DePaul University, 1997.

[5] M. Raymer, "Quantum physics: What everyone needs to know," in *Oxford University Press*, 2017.

[6] M. A. Nielsen and I. L. Chuang, "Quantum Computation and Quantum Information," in *Cambridge University Press*, 2010.