Systematic Review of Quantum Computing’s Exponential Speedup

*California State University Long Beach*

***Abstract—Since the conception of quantum computing in the early 1980’s it has been known that quantum computers can provide a speedup over classical computers. This text seeks to uncover whether claims of an exponential quantum speedup have been proven. The chronological development of computation will be discussed in an attempt to understand computational limits and the algorithmic classification of computing. The reviewed literature shows that the exponential speedup provided by quantum computing has been exemplified but never proven. The cases which have exemplified an exponential advantage are commonly thought to be a result of incomplete understanding, not inherent superiority.***

***Keywords***—**Quantum computing, computational complexity, classical computing, Turing complete, quantum superiority, quantum advantage.**

1. INTRODUCTION

There are two mainstream types of computing. The first is classical computing. Classical computing also known as binary computing encompasses personal computers, data centers, phones and any other device with a discrete processor. The second form of computing is quantum computing. Quantum computing is the exploitation of quantum-mechanical properties such as entanglement and superposition to solve well defined problems. While both classical and quantum computers are currently making tremendous progress in fields like artificial intelligence, there is a current theme of thought that classical computing will soon be inferior to quantum2. Well-established problems that would take a classical computer thousands of years to compute are thought to solvable by a quantum computer in seconds.

In order to understand the limitations and differentiation of binary and quantum computers a high-level understanding of their mechanisms is necessary.

*A. Classical Computing Background*

Binary computers are built upon two values. These values are often abbreviated with the numbers 1 and 0. By combining sequences of 1s and 0s, unique states of a binary computer can be created. For example: 00, 01, 10, 11 are the four distinct states for the sequence of length two. Each of these states can be used to represent a number. In the example of length two there are four distinct states, therefore four numbers can be represented. This representation of numbers allows calculation to be conducted. It is essential to note that as soon as a single 1 or 0 is altered, a different state is entered. For this reason, classical computers can only represent one state at any given time. This single state limitation means that for certain classes of problems, the number of required state switches is so large that in practice, classical computers become unusable.

The following optimization problem can provide such an example:

Given a transit system with 274 operational subway lines, identify which line to remove from operation so that the removal has the lowest impact on the overall system.

The subway lines are not independent; therefore, a product of 2274 states must be tested to guarantee the optimal solution. 2274 is so large that in order to solve this problem a classical computer would require more state changes than there are atoms in the universe.

*B. Quantum Computing Background*

Unlike classical computers, quantum computers can represent more than one state at a time through superposition. Values in a quantum computer do not have to be either a 1 or a 0, they can be both. For this reason, unlike classical computers, the overhead associated with switching states is non-existent. An example of a quantum algorithm utilizing this increased efficiency is given in part c of the results section.

With the basics of quantum and classical computing established, this paper will outline the framework of computation set forth by Alan Turing and Alonzo Church and then the algorithmic categorization used to classify computational speedups. With these frameworks established, the implications of the three most relevant quantum papers of the last 25 years will be scrutinized to discern the validity of an exponential quantum speedup.

1. METHODS

To establish the theory of computation and algorithmic classification the first publications which defined the respective idea were used. More famous or well cited publications were discarded. To identify the three papers which best reflect the current understanding of an exponential quantum advantage the following methodology was used. All papers to be given consideration must have been contained in the Association for Computing Machinery’s digital library. The top two papers, by number of citations, containing “quantum” in their title and indexed under both “Theory of computation” and “Design and analysis of algorithms” were selected. Any paper indexed under both “Theory of computation” and “Complexity classes” that addressed quantum complexity classes was also included.

This review will not delve into current research concerning classical computers as their limitations are well established. The methods of reaching such limits are outside the scope of this paper. Popular classical computing applications such as artificial intelligence will also be ignored as they are nothing more than recursive analysis. The complexity of recursive analysis was covered by Gödel in the early 1900’s 3. Analog computing, one of the major frameworks that exists alongside quantum and classical computing will not be explored. Analog computing is the least common of the three frameworks and currently is applied across a small number of fields.

1. DISCUSSION

In 1936 Alan Turing and Alonzo Church formally defined computation by establishing three criteria all computers must meet in order to be considered capable of computation4,14. First all computers must be able to store integers in memory for recall. Second a computer must be able to store in memory the result of subtracting two integers. Finally, based on a subtraction result, a computer must be able to alter the order in which it executes. As both classical and quantum machines meet these three criteria, they are considered computers. While the power of meeting these three criteria is readily observable to those who have used a computing device, the imposed constraints are not as easily perceived.

*A. Limits of Computing*

In 1962 [Tibor Rad](https://en.wikipedia.org/wiki/Tibor_Rad%C3%B3)o wrote a paper titled "On Non-Computable Functions". In it he gives an example of a problem a no computer can solve. Rado referred to his problem as the “busy beaver” problem10. Computers are rule based logic machines. That is, they take a value in, manipulate the value according to a list of rules, and return a new value. If a computer were to take in a value and loop endlessly over its list of rules, no value would ever be returned. The computer is thus useless. For this reason, Rado was only interested in machines whose manipulations terminate after a certain number of steps. Rado’s busy beaver problem requires an understanding that computers can be grouped by the number of rules they require in order to meet Turing’s criteria of computability. There are computers that can meet Turing’s standard with a single rule, others with two, three, four and so on and so forth. Rado showed that as the number of rules increase, so too do the number of steps the computer can take. Rado then showed that this number of steps grows at a rate faster than any computer could possibly compute. A rate which grows faster than even mathematical notation can express (shown in Table 1).

TABLE I

Rules to Possible Steps Relation

|  |  |
| --- | --- |
| **Number of Rules** | **Number of Possible Steps** |
| 2 | 6 |
| 3 | 21 |
| 4 | 107 |
| 5 | 47176870\* |
| 6 | 7.4×1036534 \* |
| 7 | 102\*10^10^101870535 \* |
| \* Indicates that the exact number of steps is unknown but must be at least as large as the value shown. | |

Due to the growth rate of the number of possibilities, asking a computer to list out the maximum number of steps for every value of rules is impossible. This task is a computationally unsolvable problem. Neither classical nor quantum computers will ever be able to complete such a task. Shortly after Rado’s paper was published, the questions computer scientists were asking started to change. Questions of possibility were replaced with those of speed.

*B. Algorithmic Complexity*

Three years later in 1965, Hartmanis and Stearns published, “*On the Computational Complexity of Algorithms*”. A paper that tied the functions of mathematical limits to the time domain a computer requires to solve a well-defined problem6. More simply, the pair set forth an algorithm-based classification system for labeling the speed at which computers can solve a problem. The time taken to arrive at a solution is referred to as runtime. Two of these classes - polynomial and exponential - are central to the quantum speedup question.

Both polynomial and exponential runtimes are formulated as a functional relation to the complexity of their inputs. An input is typically represented by the letter n. Polynomial algorithms are those whose runtime relational exponent is a constant whole number (n1, n2, n3, n4…) Exponential algorithms have polynomials as their exponent and a constant base (2n, 22n, 23n…) As n increases in size, the discrepancy between the runtimes of the two classes grows exponentially. With this algorithmic classification, the question of exponential quantum speedup can be rewritten as: Are there problems a quantum computer can solve in polynomial time that a classical computer would need exponential time to solve?

*C. Quantum Evolution*

Since quantum computing’s inception there have been three major developments regarding quantum speedups. The papers for these three developments are, *“*[*Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*](https://epubs.siam.org/doi/abs/10.1137/S0036144598347011)*”,* “*A fast quantum mechanical algorithm for database search”, and “Oracle separation of BQP and PH*”.

The first paper, published by Peter Shor in 1995, addresses the factorization of prime numbers by a quantum computer. For classical computers, an algorithm to factor primes efficiently has never been discovered. This is what makes today's encryption standards hard to crack. In his paper, Shor provided a quantum algorithm that can crack such encryptions in polynomial time11. This algorithm was the first discovered example of a quantum algorithm without a corresponding classical algorithm that can accomplish the same task in polynomial time. All known classical algorithms for factoring are at best exponential. Note that in this case, a classical algorithm might theoretically exist for factorization but have not yet been discovered.

The second paper, written by Lov Grover in 1996, shares many similarities with Shor’s publication. Like Shor, Grover produced an algorithm for quantum computation5. Unlike Shor, Grover’s algorithm was optimal. Optimality in this case means that the best possible algorithms run on a classical computer cannot match Grover’s algorithm on a quantum computer. Specifically, Grover proved that given a list of unsorted numbers a quantum computer can locate a specific number after at worst √n operations, with n being the number of elements in the list. A classical computer must check every element in the list, giving it a worst-case n number of operations. √n versus n is a quadratic not exponential speedup in computation. This proven quadratic speedup is what underlies that ideology that quantum computers are superior to their classical counterparts.

The most recent of the three papers was published by Ran Raz and Avishay Tal in 2019. The *Oracle separation of BQP and PH* directly addresses the algorithmic differences between quantum and classical computers13. Unlike Grover and Shor, Raz and Tal do not strictly rely upon algorithms to establish the separation of what quantum and classical computers can computationally achieve. The pair instead established the separation by posing a question to both a quantum and classical system. The question they posed is called Forrelation. Forrelation is a decision problem that relates the correlation of a Boolean function to a second function under a Fourier transform. By providing each system with hints towards the problem’s solution, the two were able to show the algorithmic divergence of the two systems. Tal and Raz found that quantum computers required a single hint. The pair then proved that even with an infinite number of hints classical computers were unable to reach a solution. This result shows that the possibility of an exponential speedup may be a conservative lower bound. Their solution is unfortunately not a comprehensive proof due to their use of an oracle, a theoretical tool, commonly used in quantum proofs. Nonetheless, their results are still arguably the closest to clearly separating classical and quantum capabilities in existence.



IV. CONCLUSIONS

Upon reviewing the feasibility of an exponential quantum speedup, the results are inconclusive. Algorithms such Shor’s Algorithm for Factorization offer exponential speedups over the best current classical algorithms. There is no indication that classical algorithms could not be improved to match the algorithms offering exponential speedup. The vast majority of papers published on quantum computing dealt directly with advances in physics or the hardware required to properly run quantum systems. Given that questions of quantum computation were surprisingly rare, any proclaimed exponential speedup results should be heavily scrutinized.

ACKNOWLEDGMENTS

This paper would not have been possible without the Association for Computing Machinery’s dedication to making computing knowledge readily accessible to all individuals.

REFERENCES

1. P. Arrighi and G. Dowek. *The physical Church-Turing thesis and the principles of quantum theory*. arXiv:1102.1612
2. F. Arute, K. Arya, R. Babbush et al. *Quantum supremacy using a programmable superconducting processor*. Nature (2019) 574, 505–510
3. L. Beklemishev. *Gödel incompleteness theorems and the limits of their applicability*. Russ. Math. Surv. (2010), 65 857
4. A. Church. *An unsolvable problem of elementary number theory.* American journal of mathematics, 1936.
5. LK Grover. *A fast quantum mechanical algorithm for database search.* Proceedings of the twenty-eighth annual ACM …, 1996
6. J. Hartmanis and R. E. Stearns. *On the computational complexity of algorithms*, Trans. Amer. Math. Soc. 117 (1965), 285-306
7. R.M. Karp. *Reducibility among combinatorial problems*. Complexity of computer computations, 1972.
8. L.A. Levin. *Universal Sequential Search Problems.* Problemy Predaci Informacii. 9 (1973), 265-266
9. B. Olivier, D. Graça, and A. Pouly. *Computing with Polynomial Ordinary Differential Equations.* Journal of Complexity 36 (2016): 106–40.
10. T. Rado, *On non-computable functions*, The Bell System Technical Journal,41 (1962), 877-884
11. PW Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer.* SIAM review, 1999.
12. J.W. Steinmetz. *An Intuitively Complete Analysis of Godel's Incompleteness.* arXiv:1512.03667
13. A.Tal and R. Raz, *Oracle separation of BQP and PH*, STOC 2019: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, 2019, 13–23
14. A. M. Turing. *Computing Machinery and Intelligence*. Mind, Volume LIX, Issue 236, 1950, 433–460
15. A.M. Turing. *On computable numbers, with an application to the Entscheidungsproblem.* Journal of Math 140 (1936), 177.205.52