Effects of the Development of Quantum Computing on NP Difficulty Problems

Matt Fletcher

Abstract—With general purpose quantum computers on the verge of being invented in the next decade, some worry about what the effects these will have on society. Because quantum computers operate off of rules that break the laws of modern physics, they are far more powerful at computing certain algorithsm. Many secure operations nowadays are only secure due to the extreme difficulty in solving the math problems that they are built from. Two common examples of these operations are RSA Encryption and blockchain. RSA's security lies in prime factorization, and blockchain's security lies in the difficulty of its hashing algorithm. Since the 1980's, the RSA Encryption algorithm has been used as a near foolproof method of encrypting sensitive data sent over the Internet. Furthermore, blockchain's use in cryptocurrencies is only secure due to the irreversable hashing algorithm used, which ensures the security of the individual accounts. This paper analyzes the effects of quantum computing on the NP applications of RSA Encryption and blockchain, as well as a few other applications in the modern world.

Index Terms—Cryptography, Encryption, Quantum computing, P vs NP, Prime Factorization, RSA, Traveling Salesman

1 Introduction

In 1955, mathematician John Nash penned a letter to the National Security Agency that would change the face of computing forever [1]. In this letter, he presented a theory about the computational power required to find the solutions to various mathematical problems. In particular, he focused on the art of cryptography, and the length of time necessary to crack an encryption. Up to that point, computer scientists who created algorithms concentrated solely on the required time to solve a particular length of problem. Nash suggested that instead of fixating on the computational duration for a problem with a particular input length, scientists should consider the rate of growth of the problem resolution time given the length of the inputs grew at a linear rate. Furthermore, in lieu of computation time, scientists should focus on the number of computational steps[1] required to solve the problem. The shape of this growth curve would help classify the difficulty of the problem.

About 3 decades after Nash's letter, a theoretical physicist at MIT named Richard Feynman proposed the idea of a computer that would operate through the use of quantum mechanics. At the time, the idea was simply theoretical, as the technology at the time was not advanced enough to be able to create such a "quantum computer". However, scientists began to wonder what the effects of such a computer would have on math and science. This curiosity sparked what has now become the leading edge of innovation in computing power.

2 Classifying P and NP Problems

The rate of growth of a problem's difficulty as its input lengths increase can vary drastically. For example, the difficulty of addition of numbers grows at a linear rate. Doubling the number of digits to be added results in approximately double the computation time. Therefore, these problems grow at a linear rate[2]. However, problems such as multiplication grow at a slightly faster pace. Doubling the length of the inputs results in a growth of 2^2 times the number of computational steps. Tripling the length of the input results in a growth of 3^2 times the number of computational steps. This growth may be somewhat rapid, but can be expressed as a polynomial[3]. The exponent of the growth rate is always constant. Polynomial growth can therefore be displayed as some combination of terms in the form n^k, where n is a variable and k is a constant.

Another type of problem is anything similar to cracking a password by brute force. For simplicity sake, assume the password is a PIN composed only of digits 0-9. In the worst case scenario, for a 1 digit PIN, the computer would take $10^1 = 10$ guesses. For a 2 digit PIN, the computer would take $10^2 = 100$ guesses. For a 4 digit pin, the computer would take $10^4 = 10\,000$ guesses. This is growing at a rate of 10^k, where k is some constant. This growth pattern is known as exponential growth, where the variable is in the exponent. The resultant curve is extremely sharp. For context, if a password was able to consist of any numbers, letters, or special characters on a standard US keyboard, with only an eight character password, the number of computational steps required skyrockets up to 82^8, or 2 with fifteen zeros following.

Mathematicians had been noticing problems with similar growth patterns in all different fields of math and science. So in 1971, mathematicians Stephen Cook and Leonid Levin formulated one of the most famous Millenium Prize Problems[4] in all of mathematics: the P vs NP problem.

1. A computational step is the number of state changes that the machine processes in carrying out the set of steps to solve the problem.

2. Linear growth is still technically considered a polynomial growth of the form n^k with $k = 0$.

3. A polynomial is an expression that can be written in the form $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$

4. The Millenium Prize problems are 7 unsolved math problems from throughout mathematical history. The Clay Mathematics Institute will award one million dollars to the first confirmed proof of each problem.

2.1 P-Type problems

The first type is known as a problem with difficulty P. To find the largest number in a list of numbers, a computer must iterate through each element in the list one time[5]. Assume this uses n computational steps. If the length of the list is doubled, the computer must go through $2n$ steps, or double the number of steps, in order to determine the largest element in the list. Tripling the length results in $3n$ computational steps. Therefore, increasing the list from n elements to $k \cdot n$ elements will result in having to use k times the number of computational steps[6]. This increase results in a linear growth. By computer standards, this is considered a slow growth. Mathematicians decided to classify these problems as type P problems, as they could be solved in **P**olynomial time. Not only are these problems easy to solve, but they are also easy to confirm if a potential solution is correct.

2.2 NP-type Problems

"Nondeterministic Polynomial time problems" (also known as NP problems) are not only difficult to solve, but also easy to check a solution to in polynomial time[7]. One of the most common examples of an NP problem is finding a subset of a list that satisfies a given requirement. This is often referred to as the hackey-sack problem. If someone is presented with a collection of small sacks with specified random weights, and then asked to find a subset of those sacks that result in a given weight, the only way to approach this problem with current computational methods is a brute force algorithm, guessing a random subset, and checking the result. Unlike the P-type problem of finding the largest number in a list, the number of computational steps required for this problem results in a much faster growth. Assume that initially, just 2 sacks are given. Call these c_1 and c_2. To figure out which combination yields the correct answer, one must try the following combinations:

- c_1
- c_2
- $c_1 + c_2$

This is only 3 iterations, which seems rather tame. Mathematically, the number of iterations required can be represented with the following expression, where r is the number of sacks in the initial collection [8].

$$\sum_{k=1}^{r} {}^{r}C_k$$

However, the number of computational steps in this problem grows at an extremely fast rate. For 4 sacks in the initial collection, the number of iterations required jumps to $\sum_{k=1}^{4} {}^{4}C_k = 15$. Doubling the length of the input quintupled the number of computational steps. Through the use of Pascal's Triangle, one can find that the required number of computational steps for an initial collection of k sacks is $2^k - 1$. For an idea of the rate of growth, a collection of 30 sacks would require over 1 billion computational steps.

Yet another example of an NP problem is the factoring of a number into its prime factors. In his 1801 book *Disquisitiones Arithmeticae*, mathematician Frederick Gauss proved that any number has exactly 1 prime factorization [2]. Checking if a given factorization for a number is correct is a P-difficulty problem. However, in order to find the unique factorization for a number, the only method currently known is guessing and checking every single factor for that number. As the length of the target number grows linearly, the computation time grows exponentially. As the growth of computation time is not of a polynomial form, the factorization problem is considered an NP-difficulty problem. In other words, a NP problem can only be solved with brute force, so given inputs of sufficient length, the problem can be considered for all practical purposes unsolvable.

3 Evolution of Computational Methods

COMPUTERS HAVE evolved drastically over the decades. In first generation computers, or computer technology designed from 1939 to 1954, computers operated using vaccuum tubes. Even a basic computer with hardware specifications orders of magnitude weaker than a modern day pocket calculator would occupy an entire room, and cost hundreds of thousands of dollars [3].

These vacuum tubes, due to the excess heat they created, were extremely unreliable and were prone to failure every few hours. See Fig. 1 for an example of a vacuum tube computer. Computers soon transitioned to using microswitches known as transistors. Compared to the vacuum tubes, transistors could be packed far more densely into the same area. Soon enough, however, these transistors hit their limits. Due to the high power usage per transistor, they could only be so small before the electricity running through them resulted in intereference with the neighboring transistors. Eventually, transistors were changed to a special type of transistor known as a MOSFET[9]. These operate on the same underlying principle as a regular transistor, with a binary existance of a 1 or a 0 for on and off. Due to their low power usage, MOSFET's could be packed far more densely onto a circuit board. Eventually, these transistors were integrated into the layers of silicon in the processing chip, which allowed the size of an individual transistor to be in the tens of nanometers [4][10]. According to Moore's law[11], the area density (and therefore processing power) of computers grows at an exponential rate, doubling each year [5].

However, regular computers will soon reach the limit of how small these specialized transistors can get. Current

5. For each element in the list, check if it's larger than the previous element. If it is, store that value as the largest value.

6. A sample Python script that does this calculation is in Appendix A.

7. "Easy" and "difficult" in this case mean that they are respectively solvable and not solvable by a computer in a reasonable amount of time.

8. See Appendix B for an explanation of the ${}^{r}C_k$ notation

9. Metal Oxide Semiconductor Field Effect Transistors

10. For example, the Intel i7 Quad Core CPU packs 731 million transistors into a board only 0.63" by 0.63" [4].

11. Moore's law is based off of Intel co-founder Gordon Moore's observation that the density of transistors on microprocessors had approximately doubled every year since the early age of computers. Moore's law states that this trend will continue every year into the future.

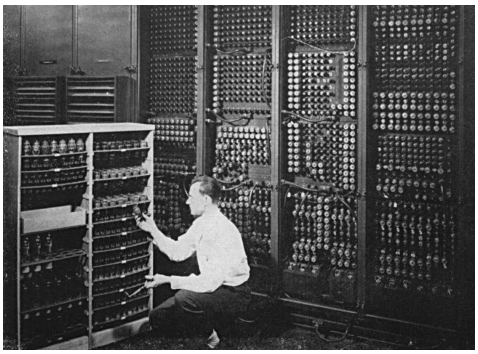

Fig. 1. An ENIACS computer, which operated off of vacuum tubes [7].

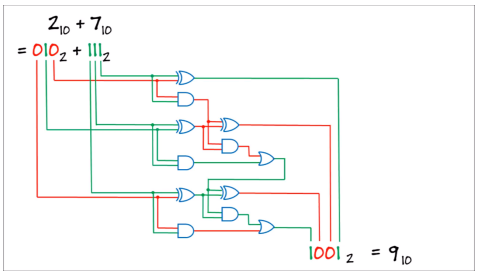

Fig. 2. A sample setup using logic gates to add 2 numbers [10].

CPU architecture is based off of 14 nanometer transistors. The next generation of transistors will reach 10 nanometers. [6] Computer scientists predict that transistors will reach their physical limit size sometime in the mid 2020's. This limit is due to the fact that the transistors are approaching the size of atoms [8]. Because of this asymptote, computers can no longer use an increase of transistor density for an speed boost. Another approach has to be derived.

3.1 How Transistors Run A Computer

How does a microscopic electronic component that can only be in 2 states operate a device that can perform billions of calculations per second? A transistor is a switch that can either be on or off to represent a 1 or a 0, respectively[12]. It will exist in exactly one of these states at any given time (in other words, a transistor cannot exist in a quasi-on state). One transistor by itself cannot do much. However, combining the transistors can result in extremely powerful calculations. Combining these transistors forms logical gates known as AND, OR,and NOT. For an AND gate, the input is a pair of bits, and the output is a single bit. When both input bits are on, the output is is on, but otherwise, the output is off. Another type of gate is the OR gate. This has 2 inputs and 1 output. If either of the inputs is true, or both are true, the output is true. Only if both are false is the output false.

For a NOT gate, the input is a single bit and the output is also a single bit. The output is simply the opposite of the input.

Combining these base transistors allows for other types of logic gates to be formed, including the XOR, NAND, NOR and XNOR gates.

By placing a NOT gate on the output of an AND gate, a NAND gate is formed, which outputs the opposite of the AND gate.

The XOR gate, also known as the eXclusive OR gate. Unlike the OR gate, the XOR gate will only show true if exactly 1 input is true. This is also known as the either/or gate.

The combination of these logical gates results a boolean table known as a Truth Table [9]. This allows for very low level instructions, such as STORE, LOAD, ADD, or COMPARE. See Figure 2 above for a sample addition machine.

12. Most computers use a 5V charge to signal an ON, and a 0V charge to signal an OFF [9]

These instructions can be used to address the data stored in RAM. The language used for these instructions is called 'assembly language' or 'machine code'[13].

A quantum computer operates uses a slightly different principle. Instead of operating on a physical piece of hardware (the transistor) that produces a bit, quantum computers operate by observation of a photon that represents a qubit[14]. A qubit is a special type of bit that can either represent a 1, a 0, or a superposition of a 1 and a 0. But how can something exist in both a 1 and a 0 at the same time? To understand this, one must first understand the principle behind Schrödingers cat.

The infamous Schrödinger's cat experiment is performed as follows. Imagine a living cat inside a sealed bunker, with a sealed beaker of radioactive material, a Geiger counter attached to a hammer, and a small piece of radioactive material. As radioactive decay is a random process, the material has a 50% chance of having one of its atoms decay in the next hour (which would set off the Geiger counter, smashing the hammer into the vial of gas and releasing it into the bunker, killing the cat). As long as the bunker is sealed and unobserved, the cat has a 50% chance of being dead and a 50% chance of being alive. When the bunker is opened, however, the cat must either be dead or alive. The instant before the bunker is opened, the cat exists in a quantum superposition of both dead and alive [11].

Just like the cat, as long as the qubit is unobserved, it exists in a quantum superposition both a 1 and a 0. As soon as the qubit is observed, it "snaps" to either a 1 or a 0. Another way of thinking about this is like a coin that is flipped and is spinning in midair. At any point in the air, it is practically impossible to figure out if the orientation of the coin is heads or tails. However, smashing a fist down on top of the coin will force it to go to either a heads or a tails orientation. But how does this help a quantum computer work faster than a regular computer?

3.2 How a Quantum Computer can solve NP-type problems

The operations of a computer can be compared to an office worker sitting at their desk. The storage (hard drive) is comparable to a file cabinet for long term storage. The

13. This language is referred to as a low-level language, while more common languages like Python, FORTRAN, C/C++, and Java are referred to as high-level languages. This refers to the logical distance between the syntax of the language and the machine language, along with the amount of abstraction used in the language.

14. Qubit is a juxtaposition of QUantum BIT

Random Access Memory (RAM, also known as memory) is comparable to the top of the desk, where papers can be placed when they are being worked on. Finally, the CPU is comparable to the person sitting at the desk, actually performing the tasks. The person can only work on something on the top of their desk, as trying to read a piece of paper sitting in a file cabinet would be awkward and slow[15]. Therefore, the person pulls an item out of their file cabinet and places it on their desk. The desk can only hold so many papers, so when the desk runs out of room, some papers have to be put back into the file storage.

Similiarly to the desk analogy, a CPU instructs the hard drive to load certain pieces of information from the hard drive into the RAM. When the computer runs out of free memory, the CPU removes information from the RAM that isn't being used. Unlike the desk analogy, however, the CPU does not directly operate from the RAM. Instead, the CPU loads a small bit of information from the RAM into the register of the CPU. The register is a very small amount of extremely fast storage directly tied to the CPU. For example, an Intel i7 CPU has 64 bit, 80 bit, and 120 bit registers[16]. Once in the register, the bits go through the transistors discussed earlier, where the logical gates perform instructions based on the data going through them [12] [13].

The bits in a classical computer have a value of 1 or 0. This is relatively efficient, but storing an N-bit number requires N bits. In other words, storing a 64 bit value takes 64 bits, and storing a 256 bit value takes 256 bits. The transistors in the CPU only recognize the 2 states, and operate serially (in series) on each bit.

A quantum computer, on the other hand, uses bits that exist in a state that is either a $|1\}$, a $|0\}$, or a superposition that can be represented with the expression $a|1\} + b|0\}$. The state of the bit can be expressed using complex vector addition, varying the values of a and b. However, due to a quantum theory known as quantum entanglement, the state of 1 qubit affects the other qubits in the system. Two particles can be entangled even if they are separated by physical space. Due to the entanglement, the qubits allow for exponentially growing identification of bits. N qubits can represent 2^N classical bits. A very critical point about this: quantum computers do not measure the values of the individial qubits. Instead, they measure the "correlations", or interactions, between the qubits. Just like Schrödinger's cat, as soon as the qubits themselves are observed, the system changes from its original state. The cutting link to success came when two computers scientists, David Deutsch and Richard Jozsa, determined an algorithm that allowed for a computer to deduce what the original state of the qubits was most likely to have been before observation. Think of it like a house of cards that cannot be observed without them falling into a pile. The Jozsa-Deutsch algorithm allows for a computer to determine the most likely original state of the cards. Although the exact original state cannot be determined in a single computational run, by running multiple computations, the computer can deduce what the most likely original state of the qubits was, which will result in the solution to the problem [14].

This ability is what makes a quantum computer so powerful. Instead of operating serially on a stream of bits, the computer can operate in parallel, on multiple bits simultaneously. This capability, combined with the right algorithm, will be able to solve a NP problem in polynomial time.

4 Effects of Quantum Computing

WHAT effects will the rise of a general purpose quantum computer have on real life applications? Several benefits are readily apparent, including military interception of foreign messages, optimization of route generation, and advancement of artificial intelligence. However, several drawbacks exist, the two most critical being encryption techniques and blockchain.

4.1 Cryptography

The United States Military expresses a large amount of interest in quantum computing, primarily for its use in cryptanalysis, or code-breaking. Cracking a code is defined as an NP problem, as the difficulty of the problem increases exponentially as the length of the message grows. Because quantum computers can analyze large numbers of bits simultaneously, they are excellent for problems that require brute force calculations. This allows for the breaking of coded messages from enemy intelligence, ultimately allowing for an advantage over the enemy to successfully complete a mission while saving lives.

4.2 Route Generation

Another use of quantum computing arises in finding the shortest route between a set of points. Imagine a door-to-door salesman, with a set of houses he has to visit. To find the most efficient route to visit all of these houses requires a brute force approach. Guessing a random route, as well as measuring the length of that route are both extremely easy problems, classified as P difficulty. However, determining the shortest route is considered an NP problem[17]. In math terms, this problem is attempting to find the most efficient Hamiltonian Cycle to take through a set of N nodes. In more simplified terms, this is finding the shortest route between a set of random points. One practical application of this is route generation for logistics management companies.

Logistics companies such as the United Postal Service (UPS) deliver millions of packages every day, with the route changing daily. But finding the shortest path is not the only issue. Certain packages must be delivered before others, due to priority shipping methods (the customer who paid for priority overnight express will get his package delivered before the customer who paid for the base shipping rate). The current algorithms use a heuristic approach, approximating the best route and using "rules of thumb", resulting in sub-optimal routes. Again, the only way to find the best route currently is with brute force, which is impossible with classical computing. In fact, the number of possible routes for a single UPS truck to take daily exceeds the the

15. Interestingly enough, trying to work on a paper while the paper is still in the file cabinet can be compared to a pagefile, where the CPU pulls data directly from storage.

16. Compare these register sizes to the RAM in an average desktop, which is measured in gigabytes.

17. See Appendix C

number of nanoseconds that the Earth has existed [15]. With quantum computing, this calculation would be able to be solved in polynomial time, allowing UPS to find optimal routes for all of its trucks. This would allow for both time and money to be saved. Furthermore, because the trucks are traveling less on the road, they are emitting less pollution.

4.3 Artificial Intelligence

One highly important section of the field of quantum computing is its application in the field of artificial intelligence. According to IBM, one such application is "Making facets of artificial intelligence such as machine learning much more powerful when data sets are very large, such as in searching images or video." [16] Machine learning (such as IBM's Watson) relies on searching a vast database of information to calculate results. As the size of the database increases, the length of time to search for the required information also increases. Through the use of quantum computing, these database searches can be completed on orders of magnitude faster than would be possible with standard computing methods [16].

The initial drawbacks for most usage would be the cost of the hardware, as well as developing the complex algorithms that can properly run on quantum-based hardware. However, as with most all first-generation hardware, the initial cost is astronomically high. If quantum computers follow anything like the path of classical computers' development, the price will soon drop, allowing companies to use quantum algorithms in the use of artificial intelligence, turning a high-cost item into a cost-effective alternative.

The many benefits of quantum computing seem to point towards a positive outlook. However, as with any technology, there will always be drawbacks.

4.4 RSA Encryption

As a shopper enters their credit card details for an online store website, the shopper is assured that their credit card details and other personal data are protected if the green lock icon is present in their browser. The only reason this lock icon has any significant value is thanks to two major points:

1) Prime factorization is classified as an NP problem

2) the difficulty of cracking this encryption (known as RSA encryption) is extremely difficult and time consuming using current computation methods.

In 1978, three computer scientists, Ron Rivest, Adi Shamir and Leonard Adleman, publicly released an encryption algorithm titled after their names, RSA [17]. The basis of this asymmetric encryption technique[18] is two secret prime numbers (known as private keys) multiplied together to form a public key. To decrypt the message, the two original numbers must be known. Finding these original two numbers is only possible through repetitive brute force calculations[19]. For an idea of the size of these numbers, the public key length for most banking systems nowadays is over six hundred digits long[20] [18]. To complete these calculations with current methods and computing technologies would require somewhere on the order of 6.4 quadrillion years to try all possible combinations.

However, with quantum computing's parallel computation techniques, multiple factors can be tried simultaneously, cutting the time to crack an encryption drastically. This is not just a science fiction threat, though. Previously, a variant of RSA cryptography known as elliptic curve cryptography[21] was approved by the National Security Administration (NSA) as the minimum for encryption standards of private data. In August of 2017, the NSA released a statement about the threat from quantum computing:

"For those vendors and partners that have already transitioned to Suite B [elliptic curve cryptography], we recognize that this took a great deal of effort on your part, and we thank you for your efforts. Unfortunately, the growth of elliptic curve use has bumped up against the fact of continued progress in the research on quantum computing, which has made it clear that elliptic curve cryptography is not the long term solution many once hoped it would be. Thus, we have been obligated to update our strategy." [19]

In other words, even the NSA is concerned with the rise of quantum computing. Furthermore, any data that is currently stored with non-post-quantum cryptography methods will be subject to cracking within the next few decades at most.

Many major organizations, such as Internet Service Providers (ISPs) and the NSA store user data in an encrypted (and currently secure) manner. The sole reason this data is secure is because of the reliance in standard encryption methods. If a quantum computer is developed by a malicious source (foreign or domestic), everyone's so-called "private" information is at imminent risk of being decoded and stolen.

4.5 Blockchain and Cryptocurrencies

Yet another aspect of technology that would be affected with quantum computing is the use of blockchain for cryptocurrencies. In technical terms, blockchain is a public peer-to-peer ledger to account for and validate transactions of a decentralized cryptocurrency, such as Bitcoin, Ethereum, and Monero.

The most commonly used cryptocurrency is Bitcoin; however, almost all cryptocurrencies operate off of the same general principle [20].

When Alice wants to transfer some amount of Bitcoin to Bob, the transaction information is stored in the ledger, which is a public record of all of the transactions that have occurred. The records include Alice and Bob's "wallet" ID's, as well as the the amount transferred. Groups of transactions are collected into a block. These blocks are chained

18. Please note that this is a highly simplified explanation of the encryption algorithm. To describe this in full detail would take far more in-depth explanations and mathematics than would be appropriate for this paper.

19. Although some algorithms exist to reduce the work slightly using methods to eliminate groups of guesses, it is still a polynomial reduction on an exponential growth, which means the reduction is almost unnoticeable.

20. A 617 digit long public key is the key length for 2048 bit encryption.

21. Elliptic curve cryptography is an encryption method similar to RSA that uses discrete logarithms of elliptic curves to choose the private keys. However, it still operates off of the same principle, with a private key and a public key related to each other through irreversable complex mathematical algorithms.

together (hence the name), and then a "pointer" refers each block to the previous block.

Where do these blocks come from? They originate from the maintainers of the ledger. Unlike a bank, where a central authority confirms transactions, cryptocurrencies are decentralized. The maintainers of the ledger are known as 'miners', which are computers with special hardware optimized for solving large amounts of math problems [21]. When Alice sends Bob a certain value of Bitcoin, this value is recorded in the public ledger. Other miners on the network run this information through a hashing algorithm known as SHA256. For example, if Alice sends Bob 2 bitcoin, a value would be added to the ledger saying something like ALICE-2 BOB+2. At this point, anyone could theoretically change the value of 2 Bitcoin to any value desired[22]. Therefore, the value added to the ledger is hashed. Hashing is an algorithm that gives a standard size output for any size input, where even a slight change in the input results in a drastically different ouput. For reasons beyond the scope of this paper, miners must plug in random values into the hashing algorithm to verify the output. Once a majority of miners agree on the output, that value is set in stone as the correct value.

The security of cryptocurrencies relies on the assumption that no individual will ever control more than 50% of the network [22]. Once this happens, the majority control can be used maliciously in various ways, the most egregious being that an alternate history of transactions in the ledger could be passed off as truthful. This means that malicious parties could theoretically manipulate an entire currencies value, as well as steal currency from other users.

5 Conclusion

Once a reliable general purpose quantum computer is available, the face of mathematics will be changed forever. Math and science problems that would not have been possible before will now be able to be accomplished in mere seconds. Some benefits include the use of quantum computing for cryptanalysis to aid military intelligence, optimal route generation for logistics companies, and optimization of data retrieval for artificial intelligence.

But as with all new technology, there will always be drawbacks. These include the misuse of quantum computing to both crack commonly used encryption methods and destroy the cryptocurrency economy.

As an unknown author once said, "Computers are incredibly fast, powerful, and stupid. Humans are incredibly slow, innacurate, and brilliant. Together, they are powerful beyond imagination."

Appendix A
Sample Code for Largest Element

Assume the list of numbers is stored in a list named num_list. The following is written in Python. Note how there is only 1 FOR loop.

22. Note that the actual value added to the ledger is far more complicated. This is just a psuedocode explanation of the general process.

```
greatest_element = num_list[0]
for element in num_list:
        if element>greatest_element:
                element = greatest_element
return element
```

Appendix B
Proof of Iteration Count for Sacks

Start with a test case, then prove for a general solution. Let the number of sacks in the initial collection be 4.

To figure out the total possible number of sacks, use a case by case scenario. Every final count of sacks will either contain 1, 2, 3, or 4 sacks.

- **Case 1**, picking 4 sacks.
  There are 4 possible choices for the first sack, 3 possible for the second, 2 for the third, and 1 for the fourth. Mathematically represented, there are ${}^4P_4 = 4! = 4\cdot3\cdot2\cdot1$ possible combinations. However, as the order the sacks are picked in does not matter (picking sacks 1,2,3, and 4 is the same as picking sacks 3,2,4, and 1), divide by $4!$. The mathematical represenatation of this compensation is ${}^4C_4$.
  $$\frac{4!}{4!} = 1$$
  This makes sense, because there's only 1 way to pick a collection of all 4 sacks.
- **Case 2**, picking 3 sacks.
  With the same logic as the previous case, there are ${}^4P_3 = 4\cdot3\cdot2$ possible unordered combinations. However, each choice of 3 sacks can be arranged in 3! ways, so therefore divide the total number of collections $(4!)$ by the number of arrangements of each $(3!)$.
  $$\frac{4!}{3!} = \frac{24}{6} = 4$$
- **Case 3**, picking 2 sacks.
  $${}^4C_2 = \frac{{}^4P_2}{2!} = 6$$
- **Case 4**, picking 1 sack.
  $${}^4C_1 = \frac{{}^4P_1}{1!} = 4$$

  Adding each of these together results in $1{+}4{+}6{+}4 = 15$.
  The number of possible combinations for each case is not a random number. These numbers actually come from the Binomial Coefficients of Pascal's Triangle.

$n=0$:				1				
$n=1$:			1		1			
$n=2$:		1		2		1		
$n=3$:	1		3		3		1	
$n=4$:	1	4		6		4		1

  The row $n = 4$ has the same values as each of the cases discussed above. (The only number that does

not show is the first 1, as this represents the number of ways to pick zero elements from the list of initial sacks. As this is not a valid solution to the problem, it is ignored.)

APPENDIX C
TRAVELING SALESMAN NP PROOF

A simple explanation behind this statement: To find the shortest route between a set of nodes $N_1, N_2, \cdots N_Q$ of size Q, start at any node N_1. At this point, Q guesses have been checked. Now, there are $Q-1$ points left to try. After another point N_2 has been selected, $Q(Q-1)$ guesses have been attempted. After another point, the number of guesses goes to $Q(Q-1)(Q-2)$. This is increasing at a rate of $Q!$, which is a non-polynomial growth.

APPENDIX D
SELF-ASSESSMENT

This paper definitely challenged my knowledge of computing techniques. The strongest part of my paper is the explanations of complex ideas, putting them into terms that an average person without a strong knowledge of computing would understand. The weakest part of my paper is the explanation of how a quantum computer exactly works. The reason for this is because my knowledge of quantum computing is still very limited.

ACKNOWLEDGMENTS

The author would like to thank Dr Gaines Hubbel, for his amazing EH105 class, as well as the author's family and friends for proofreading. The author would also like to give a shoutout to Carrie, for her late night sanity checks. Furthermore, the author would like to thank Dunkin Donuts, Reddit ELI5, StackOverflow, and Wikipedia, without which he would have no clue what he was talking about.

REFERENCES

[1] Nsa.gov. (n.d.). NSA Declassified Documents. [online] Available at: www.nsa.gov/news-features/declassified-documents/nash-letters/assets/files/nash_letters1.pdf [Accessed 25 Nov. 2017].
[2] "Theorem of Prime Factorization", Encyclopedia Britannica, 2017. [Online]. Available: https://www.britannica.com/topic/fundamental-theorem-of-arithmetic. [Accessed: 27- Nov- 2017].
[3] "History of Computers", Homepage.cs.uri.edu, 2017. [Online]. Available: homepage.cs.uri.edu/faculty/wolfe/book/Readings/Reading03.htm. [Accessed: 25- Nov- 2017].
[4] "Computer History 101: The Development Of The PC", www.tomshardware.com, 2011. [Online]. Available: www.tomshardware.com/reviews/upgrade-repair-pc,3000-2.html, [Accessed: 25-Nov-2017]
[5] S. Shankland, "Moore's Law: The rule that really matters in tech", CNET, 2017. [Online]. Available: www.cnet.com/news/moores-law-the-rule-that-really-matters-in-tech/. [Accessed: 25- Nov- 2017].
[6] "14nm Process Technology: Opening New Horizons", Intel Developer Forum, San Francisco, CA, 2017.
[7] Computer History Museum, ENIACS Tube Change. 2017.
[8] S. Anthony, "Transistors will stop shrinking in 2021, but Moores law will live on", ARSTechnica, 2017. [Online]. Available: arstechnica.com/gadgets/2016/07/itrs-roadmap-2021-moores-law/. [Accessed: 25- Nov- 2017].
[9] W. Sangosanya, "Basic Logic Gates", Ee.surrey.ac.uk, 2017. [Online]. Available: www.ee.surrey.ac.uk/Projects/CAL/digital-logic/gatesfunc/. [Accessed: 25- Nov- 2017].
[10] Frame of Essence, Building the Bits and Qubits. 2017.
[11] M. Kramer, "The Physics Behind Schrdinger's Cat Paradox", News.nationalgeographic.com, 2017. [Online]. Available: news.nationalgeographic.com/news/2013/08/130812-physics-schrodinger-erwin-google-doodle-cat-paradox-science/. [Accessed: 25- Nov- 2017].
[12] "Quantum vs Classical Computation", Thphys.nuim.ie, 2017. [Online]. Available: http://www.thphys.nuim.ie/staff/joost/TQM/QvC.html. [Accessed: 26- Nov- 2017].
[13] Woodford, Chris. (2012/2017) Quantum computing. Retrieved from http://www.explainthatstuff.com/quantum-computing.html. [Accessed (26- Nov- 2017)]
[14] Paul E. Black, "Deutsch-Jozsa algorithm", in Dictionary of Algorithms and Data Structures [online], Vreda Pieterse and Paul E. Black, eds. 17 December 2004. (accessed 25-Nov-2017) Available from: www.nist.gov/dads/HTML/deutschJozsaAlgo.html
[15] D. Zax, "Brown Down: UPS Drivers Vs. The UPS Algorithm", Fast Company, 2017. [Online]. Available: www.fastcompany.com/3004319/brown-down-ups-drivers-vs-ups-algorithm. [Accessed: 26- Nov- 2017].
[16] "Quantum computing applications - IBM Q - US", Research.ibm.com, 2017. [Online]. Available: www.research.ibm.com/ibm-q/learn/quantum-computing-applications/. [Accessed: 25- Nov- 2017].
[17] G. Simmons, "Rivest-Shamir-Adleman encryption", Encyclopedia Britannica, 2017. [Online]. Available: www.britannica.com/topic/RSA-encryption. [Accessed: 25- Nov- 2017].
[18] Encryption and HUGE numbers. Youtube: Numberphile, 2012
[19] "Information Assurance–Suite B Cryptography", Nsa.gov,2017. [Online]. Available: www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml. [Accessed: 26- Nov- 2017].
[20] "How does Bitcoin work? - Bitcoin", Bitcoin.org, 2017. [Online]. Available: https://bitcoin.org/en/how-it-works. [Accessed: 26- Nov- 2017].
[21] D. Aggarwal, G. Brennen, T. Lee, M. Santha and M. Tomamichel, "Quantum attacks on Bitcoin, and how to protect against them", Arxiv.org, 2017. [Online]. Available: https://arxiv.org/abs/1710.10377. [Accessed: 26- Nov- 2017].
[22] D. Galeon, "Experts: "Quantum computers will break Bitcoin security within 10 years"", Futurism, 2017. [Online]. Available: https://futurism.com/bitcoins-security-quantum-computers/. [Accessed: 26- Nov- 2017].