A Survey Paper on Algorithmic Properties of Quantum Computation: What Makes an Algorithm "Good" for Quantum Computers?

JARED WEAKLY, Portland State University
NHUT LE, Portland State University

This is a survey paper that aims to answer the question of what properties make certain problems a better fit for quantum computers than for classical computers. The motivation comes from providing an accurate, highly approachable, and understandable answer to the question given in the topic; something the authors feel has not been accomplished yet. This paper was written for a class project for CS410.

CCS Concepts: • **Computer systems organization** → **Quantum computing**; • **Theory of computation** → *Quantum complexity theory*;

Additional Key Words and Phrases: Survey paper, algorithms, quantum computing, quantum algorithms, comparison

1 INTRODUCTION

When it comes to the subject of Quantum Computers, there are a few main questions that generally immediately come to the forefront of someone's mind. The first is "How do they work?" The second is "What makes them fast?" The third is "How can we take advantage of them?" In attempting to look up the answers to those questions, one will likely discover a somewhat depressing fact: No satisfactory answers that are approachable to the layman are readily available. Rather, it is somewhat of a fact of life that most all currently available explanations of how quantum computers work which are aimed at laymen are anywhere from moderately wrong to complete and unsubstantiated nonsense. Commonly, explanations such as "the computer tries all problems in parallel" or "each qubit doubles the processing power so a 50 qubit machine is as powerful as a 2^{50} bit computer" are used. It's regrettable that this sort of thing has happened as it has prevented a lot of understanding of quantum computing from reaching a more mainstream audience.

Many physicists and computer scientists will be quick to point out that quantum computing is not some miracle; quantum computers are very limited, just like classical computers. That being said, there are a few applications for which quantum computers seem extraordinarily well equipped to handle compared to their classical counterparts. This paper will attempt to introduce some motivation for how quantum machinery works in order to give an easily understandable yet accurate explanation of quantum computers. The paper will then expound on a few very famous and common examples of quantum algorithms and will provide some high level explanations for why they are effective. An eventual goal of the paper is to allow a reader to develop an intuition for how quantum computers work and for what problems a quantum computer is most suited for. Difficulty lies in the fact that people learn by using preexisting internalized analogies, concepts, metaphors, etc., to attach new concepts to old ones and thereby internalize new concepts; however, since quantum computing has its fundamental roots in the generalization of probability theory to the complex numbers, there aren't many concepts someone can use to help understand or develop an intuition for these things. As such, the authors have made the interesting choice to present some potential "hand-wavy" metaphors and explanations; these are not intended to be perfectly accurate, but rather they are intended to not be misleading and are intended to help foster a growing intuition.

2 INTUITION

2.1 Nature of Speedups

Quantum computers are occasionally far faster than classical computers for certain tasks. This speedup originates from potentially two sources: Superposition's ability to exploit highly structured promises, and a quantum system's ability to simulate quantum behavior efficiently. With superposition, one takes advantage of the structure of how a problem is laid out; intuitively, one sets up the problem in such a way that the wrong answers are most likely to "cancel out" and the right answers are most likely to remain. Efficient quantum simulation is actually not a magical property of quantum computers but rather an obvious one. Much like a person has an easier time speaking in their native language than they do in a foreign language, computers have an easier time working in their "native" world; since quantum computers are quantum, they can efficiently simulate quantum phenomena. As these are the two sources of quantum speedup, this paper will discuss one example from each category in order to offer some basic intuitive understanding of how each work. First, we will lay down a framework for an intuitive understanding of the underlying concepts and vocabulary often used when discussing quantum computers. After that, we will introduce the two motivating algorithm frameworks.

2.2 Concepts of Quantum Computing

Fundamental concepts such as Superposition and Amplitudes can be quite confusing and poorly explained, yet they are crucial to developing an understanding for why quantum computers work the way they do. Below, we introduce Amplitudes and Superposition in a way that attempts to establish an intuition and big-picture idea of the concepts.

2.2.1 Amplitudes. Amplitudes are a result of the generalizing of probability theory to include complex and negative probabilities. Below, we present an inductive buildup to complex probability by means of motivating examples.

Before we had the concept of negative numbers, mathematicians worked only with positive numbers. However, it is interesting to note that even hundreds of years before mathematicians fully formalized and accepted negative numbers as a legitimate mathematical concept, they were being used long before that in practice. Negative numbers were used to encode a binary concept into the number; embedding the concept of debt into the number were commonplace in the business world. Physics embedded direction into numbers by using negative numbers to mean the opposite direction.

As people started to model more complex behavior and relationships with numbers, the complex number became introduced and used. Complex numbers encode a two dimensional concept of dependency into a number. For instance, a two dimensional direction vector (equivalent to a compass) can be represented as a single number; angular spin can also be encoded in a complex number, which gave rise to the relationship between trig functions and Euler's constant–often used in Engineering to encode two-dimensional motion into a single number.

2.2.2 Superposition. The authors suggest that a suitable intuition for superposition is the simultaneous embodying of two concepts in one compound concept. For example, embedding two dimensions of direction into a single vector can be thought of as the "superposition" of the cardinal directions. A complex number representing direction represents simultaneously up/down and left/right; if up and down are represented equally in a number, it's a real number and if left/right are represented equally, it is purely imaginary.

Somewhat orthogonal to the concept of superposition is the fact that superposition in Quantum Mechanics can "cancel out." Continuing with the analogy of the two dimensional vector, suppose you have a vector that's $(0,0) \rightarrow (0,1)$ and one that's $(0,0) \rightarrow (0,-1)$, they would cancel each other out and result in nothing happening.

2.2.3 Amplitudes. We now return to Amplitudes in order to talk about them from a probability theory perspective; amplitudes are simply probabilities that are allowed to be negative or complex numbers. Probability theory in general allows us to talk about the likelihood of one event happening. Examples: 70% chance to rain, $\frac{1}{6}$th chance to roll a 1 on a die, … Classical probability theory always uses natural numbers; never negative numbers, never complex numbers.

Now, what would a negative probability look like? First, the authors would like to mention that they are "impossible" much in the same way that it is impossible to have negative four apples; while such a situation is absurd, the concept can be used to encode a binary condition. We extend this concept to probability by way of analogy; thus, complementing negative numbers, a negative probability would encode a binary condition. Examples: the probability that I will pay my debt given a future condition, the probability of satisfying a constraint given a certain chance of obtaining a necessary component eventually, traveling in a direction backwards through time.

We now turn to the question of what would complex probabilities look like? Complementing complex numbers, we suggest an intuition they complex probabilities would encode a two dimensional concept of conditional probability into a single probability. We motivate the understanding of how this would work with a hypothetical situation of a driver stuck in traffic.

Example The driver can move into one of 2 lanes or stay in their current lane. The driver is most likely to move into the left lane; if that lane becomes taken, the driver will most likely stay in their current lane. The driver is least likely to move into the right lane; if that lane becomes free, the driver will most likely move into the right lane. Otherwise, The driver will stay in their current lane.

A complex probability would allow someone to have a single probability for existing in a particular lane, taking into account all the possibilities; due to the property of linearity in quantum mechanics, it is guaranteed that one can always write all of the probabilities as a single superimposed probability. We now suggest an intuition to explain how the "canceling out" of complex probabilities can sometimes occur. In the previous scenario: if the left lane fills up and the right lane doesn't become free, the driver will not even consider moving as all probabilities of moving will cancel out. This canceling will take place without ever actually computing something.

2.3 Query model

We now discuss the Query Model of quantum computers. The query model is how quantum computation is analyzed and discussed; once it is understood, conceptually, how this model works, it should be much easier to understand how certain quantum algorithms work. With the query model, we are given a problem we want to solve and we are given a function; further, promises are made that this function has certain properties. We suggest the intuition that as a promise becomes more structured and strict, it omits less possible valid strings; the higher chance a string is invalid, the more likely it is that invalid strings will cancel each other out. There is more technical detail to be said about such things, but the goal here remains a high level intuition for a layman.

3 SPEEDUP AND PROBLEM STRUCTURE

It is important to remember that quantum computers achieve speedup through two possible reasons: superposition and quantum simulation. With regards to problem structure, if the problem is one of simulating quantum phenomena–eg, Boson sampling–then speedup is clear and the reason for it is fairly straightforward. The trickier and less obvious speedups possible come from those which rely on superposition. Intuitively, speedup that comes from superposition relies on being able to "cancel out" almost all of the wrong answers; this, intuitively, necessitates a promise that is very strict. As it turns out, this pattern follows through in all of quantum computing in general. The stronger the promise, the larger the speedup. However, only admitting one single answer does not suffice for having a large speedup; we see this with searching for a specific string. In the case of search, there is only one possible answer, but the structure of the search-space is completely undefined and thus we cannot maximize our ability to take advantage of superposition.

Below, we present a "hierarchy" of structure:

No speedup A problem such as finding the most frequently occurring string in a set of strings is so unstructured that it actually has no quantum speedup over classical computers.

Quadratic Problems that accept only certain input, but impose no structure in the search space tend to be only quadratic speedup (eg, searching).

Polynomial A problem that accepts only certain input that relies on other input and thus imposes some sort of structure in the search space can achieve polynomial speedup.

Exponential A problem that accepts only certain input and imposes a strong structure onto the properties of the search space can achieve exponential speedup.

Beyond Intuitively speaking, it is possible to continue to add on promises and further add arbitrary structure to a problem in the hopes of achieving higher speedups. Less intuitively, it turns out that this isn't possible. It turns out that this gap, $O(k)$ quantum queries to $\sim N^{1-1/(2k)}$ randomized classical queries, is optimal. In other words, there is an upper limit to how much speedup a quantum computer can achieve over a randomized classical computer.

4 HIDDEN SUBGROUP PROBLEM

We have seen so far how problem structure is incredibly important in quantum computing and how it plays a central role in determining how fast an algorithm is capable of running. It turns out that many problems that have quantum solutions which are far faster than classical solutions also share similar properties in problem structure. These properties were generalized into something called the Hidden Subgroup Problem (HSP).

A basic explanation of the HSP is that you are trying to find a hidden subgroup of another, larger, group. The reason this allows for such a large speedup is that these groups have an inherent structure; for the problems that we know how to solve most efficiently, that structure is called "abelian." HSP exploits the structured promises given naturally by the relationships groups with their subgroups.

In the case of abelian groups, there are tons of global properties that are inherent to abelian structures that complement quantum computers very naturally. This makes them, and structures like them, a perfect fit for solving problems quickly with quantum computers.

We digress a bit from the informal at this point in order to provide a briefly technical definition of the HSP. A concise definition of the problem, from Wikipedia, is as follows.

First, a definition of what we mean by "hiding": Given a group G, a subgroup $H \leq G$, and a set X, we say a function $f : G \longrightarrow X$ hides the subgroup H if $\forall g_1, g_2 \in G, f(g_1) = f(g_2) \iff g_1H = g_2H$ for the cosets of H.

Now we give the formal definition of the Hidden subgroup problem: Let G be a group, X a finite set, and $f : G \to X$ a function that hides a subgroup $H \leq G$. The function f is given via an oracle, which uses $O(\log |G| + \log |X|)$ bits. Using information gained from evaluations of f via its oracle, determine a generating set for H.

4.1 Properties Abelian HSP exploits.

As for the HSP itself, it takes advantage of a quantum computer's ability to quickly check global properties of things by querying in terms of global properties as much as possible. As we see from the formal definition given above, the function that we query is required to be constant on the cosets of the hidden subgroup and different for each coset.

In addition to that, abelian groups share several guaranteed properties because of their structure:

- Any two elements commute.
- Its center is the whole group.
- Deriving subgroups is trivial.
- All finite abelian groups are generated from a direct product of cyclic groups.
- The above structure rule can be used to generate the complete list of finite abelian groups.
- Periodicity and cyclic nature of abelian groups can also be easily exploited.
- Several symmetric properties exist that can be exploited as well.

4.2 What we can do with HSP

Almost every single problem in the complexity class BQP can be reduced to a special case of the HSP framework. This allows for the solving of: factoring, discrete logarithms, PellâĂŹs equation, SimonâĂŹs problem, and more. The HSP is an incredibly general framework that allows for solving a very wide variety of problems. Nevertheless, the question remains: Can we generalize this to nonabelian groups?

4.3 Generalization attempt

We know that we can build an HSP for any group out of HSPs for simple groups. Further, we know how any simple group can be built and there are only a limited amount of them. This suggests that we should be able to build a general HSP construction that will work for any problem that can be phrased as a HSP, regardless of the type of group used. Unfortunately, currently we have been unable to construct these due to an inability to fully take advantage of the structures that exist only in simple groups; as of now, they simply don't offer enough constraints for us to take advantage of.

4.4 Intuition: What kind of problems would make sense to tackle with HSP?

So far we have seen that problems that can be reduced to the HSP can hopefully be solved efficiently. Currently, we only have efficient solutions for abelian subgroups; due to the nature of their structure and how they can be exploited, they are an excellent natural fit for quantum computing. This suggests that any problem which doesn't care what order your input and output is, which has several symmetric and global properties (eg determining if a function is constant), and so on, would be a good fit for quantum computers. Keeping this in mind, let us ask ourselves about the Traveling Salesman Problem: Is this a good fit for quantum computers?

In the TSP, there's only one shortest path; thus, we only have one correct answer. It's not quite a global property, but a promise of a singular solution is still sufficient for some speedup. Also, in the TSP, the path a to b to c is equivalent to the path c to b to a; there's a limited form of commutation here. It's not fully commutative as a to c to b is not equivalent, but there looks to be enough symmetry and structure to offer some speedup. Thus, it seems quite reasonable to expect some sort of speedup–likely quadratic–over classical computers; an exponential speedup would require rephrasing the problem in a unique way, much like Shor's factoring algorithm rephrased factoring as global properties of co-primes.

5 BOSON SAMPLING

Quantum computers promise an exponential speed-up over classical. However, the construction of a universal quantum computer which could implement any quantum algorithms or quantum simulation is getting much challenging. Nevertheless, an intermediate computer between quantum computer and classical computer or single purpose machines capable of solving particular problems, may become possible on a shorter timescale. For example, the problem of building a quantum emulator - a well controlled quantum system whose dynamics approximate those of a classically intractable physical quantum system of interest - may be solvable in the medium term. Another example is Boson Sampling. Linear optics quantum computer is a leading candidate for the implementation of quantum computer. Universal quantum operators can be implemented by using linear optics, photon production and counting, quantum memory and fast feed forward. Boson Sampling, which was proposed several years ago, is a scheme for using linear-optical networks to solve sampling problems that appear to be intractable for a classical computer. It is strongly believed to solve problems that are classically hard.

5.1 Boson Sampling Problem

A linear optical quantum process, consisting of input of photons in a Fock state, unitary evolution implemented only via beam splitters and phase shifters and simultaneous photon-counting measurement of all modes, cannot be efficiently simulated classically up to some reasonable complexity assumptions. The large probability distribution that is sampled when photons are detected after a random transformation exponentially becomes more difficult using a classical computer as the number of input photons and the number of input and output ports increases. That difficulty is due to the unusual behaviour of photons. When two photons reach a beam splitter at exactly the same time, they will always follow the same path afterwards - both going either left or right - and it is that behaviour that is so hard to model classically. Those problem called boson sampling problems are related to the #**P** complexity class that is hard to solve in classical computers. Therefore, Boson Sampling device is introduced to sample that phenomenon.

5.2 Boson Sampling

Boson Sampling is a device having a set of photons arriving at a number of parallel input ports. The set of photons will go through a unitary evolution implemented only via beam splitters. Then, the output is a second set leaving via a number of parallel outputs. The illustration of Boson sampling device is shown in Fig. 1.

5.3 Boson Sampling Formalism

Suppose we start with m modes. Some configurations of $n < m$ modes are initialized with single photons $|1\rangle$ and the $(m - n)$ remaining modes are in vacuum state $|0\rangle$. Therefore, the initial state is:

$$|\psi_{in}\rangle = |1_1, \cdots, 1_n, 0_{n+1}, \cdots, 0_m\rangle = \hat{a}_1^\dagger \cdots \hat{a}_n^\dagger |0_1, \cdots, 0_m\rangle$$

where $\hat{a}_i^\dagger$ is the photon creation operator in the ith mode and $N = O(n^2)$.

The input state then passes through a unitary evolution U implemented only via beam splitters

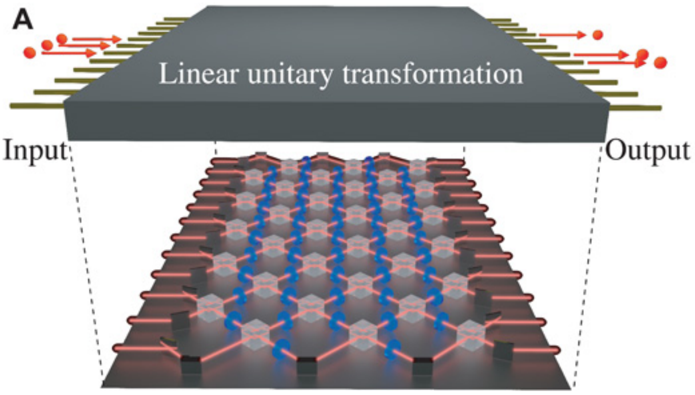

Fig. 1. The boson-sampling model

and phase shifters. The map would be:

$$\hat{U}\hat{a}_i^\dagger\hat{U}^\dagger = \sum_{j=1}^{m} U_{i,j}\hat{a}_j^\dagger$$

where $\hat{U}$ is a unitary matrix characterizing the linear optics network.

Or briefly, it could be: $\hat{a}_i^\dagger \rightarrow \sum_j U_{i,j}\hat{a}_j^\dagger$. Additionally, an arbitrary U can always be decomposed into a polynomial number of optical elements. Thus, any U of this form can always be efficiently experimentally constructed.

At the end, the output state would be:

$$|\psi_{out}\rangle = \sum_S \gamma_S \, |n_1^{(S)}, \cdots, n_m^{(S)}\rangle$$

where S is a configuration,
$n^{(S)}$ is the number of photons in the ith mode associated with configuration S,
γ_S is the amplitude associated with configuration S,
and m outputs.

Furthermore, The probability of measuring configuration S is given by $P_S = |\gamma_S|^2$. By detecting the number of output photons, the Boson distribution for some particular input configuration and unitary scatterer U can be sampled. In addition, if U is picked randomly, it's hard for classical computer to simulate the simulating of the distribution.

It is shown that the amplitudes γ_S are related to matrix permanents:

$$\gamma_S = \frac{Per(U_S)}{\sqrt{n_1^{(S)}! \cdots n_m(S)!}}$$

U_S: an $n \times n$ sub-matrix of U.
$Per(U_S)$: the permanent of U_S.
We will go through two examples below to understand more closely the relationship between γ_S and matrix permanents.

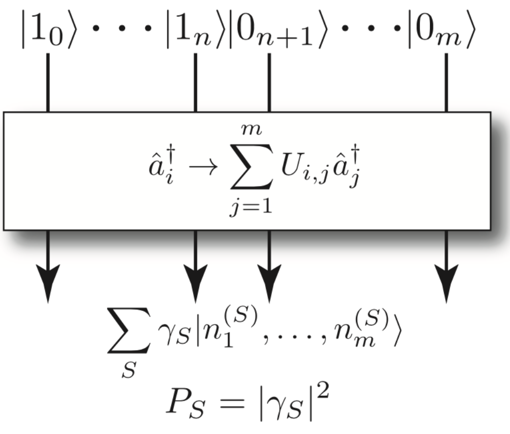
$|1_0\rangle \cdots |1_n\rangle|0_{n+1}\rangle \cdots |0_m\rangle$
$\hat{a}_i^\dagger \rightarrow \sum_{j=1}^{m} U_{i,j}\hat{a}_j^\dagger$
$\sum_S \gamma_S |n_1^{(S)}, \ldots, n_m^{(S)}\rangle$
$P_S = |\gamma_S|^2$

Fig. 2. The boson-sampling model. The input has m modes filled by n single photons and $(m-n)$ remaining modes are in vacuum state. After the linear optics network, U , we measure some configuration of photons S. The probability of measuring configuration S is $P_S = |\gamma_S|^2$.

5.3.1 Two single photons at input. Let consider there are 2 single photons at the inputs and we will calculate the amplitude of measuring one photon at output 2 and another one at output 3. There are 2! = 2 ways for photons to get the outputs. This is shown in Fig. 3. Thus, the amplitude could be written as:

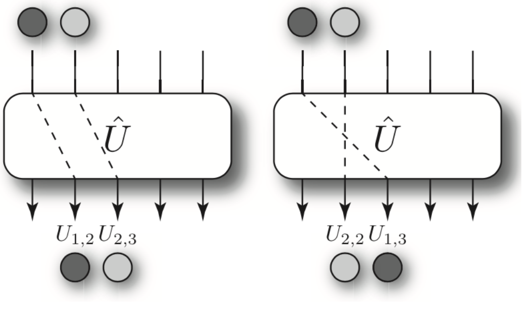

Fig. 3. Two-photon boson-sampling. In measuring the amplitude of measuring a photon at each of the output modes 2 and 3, there are two ways: either the photons pass straight through, or swap, yielding a sum of two paths.

$$\gamma_{2,3} = \underbrace{U_{1,2}U_{1,3}}_{\text{walker don't swap}} + \underbrace{U_{1,3}U_{2,2}}_{\text{walkers swap}} = Per\begin{pmatrix} U_{1,2} & U_{2,2} \\ U_{1,3} & U_{1,3} \end{pmatrix}$$

which is a 2 × 2 matrix permanent.

5.3.2 Three single photons at inputs. If there are 3 single photons at the inputs, there are 3! = 6 ways for photons to reach the outputs 1,2 and 3 which is shown in Fig. 4. Then, the amplitude is

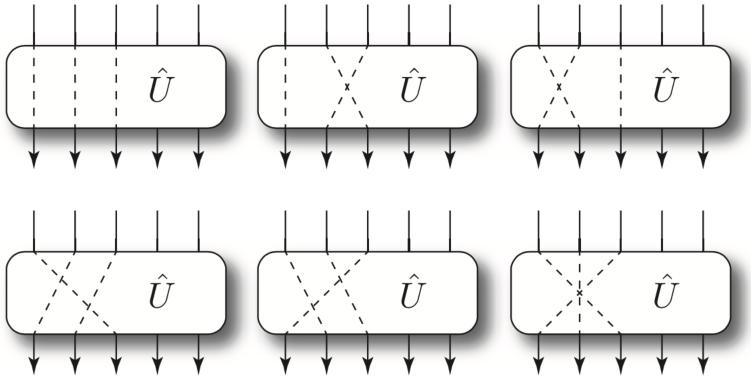

Fig. 4. Two-photon boson-sampling. In measuring the amplitude of measuring a photon at each of the output modes 2 and 3, there are two ways: either the photons pass straight through, or swap, yielding a sum of two paths.

calculated as:

$$\begin{aligned}\gamma_{1,2,3} &= U_{1,1}U_{2,2}U_{3,3} + U_{1,1}U_{3,2}U_{2,3}\\ &+U_{2,1}U_{1,2}U_{3,3} + U_{2,1}U_{3,2}U_{1,3}\\ &+U_{3,1}U_{1,2}U_{2,3} + U_{3,1}U_{2,2}U_{1,3}\\ &= Per\begin{pmatrix} U_{1,1} & U_{2,1} & U_{3,1}\\ U_{1,2} & U_{2,2} & U_{3,2}\\ U_{1,3} & U_{2,3} & U_{3,3}\end{pmatrix}\end{aligned}$$

which is a 3 × 3 matrix permanent.

In general, if there are n single photons at the inputs, there are $n!$ ways for photons to reach the outputs. Thus, the amplitude is related to $n \times n$ matrix permanent. Calculating $n \times n$ matrix permanent is $\#P$-complete, which is even harder than NP-complete. There is an algorithm calculating $n \times n$ matrix permanent with $O(2^n n^2)$ runtime. Thus, if boson sampling is able to be simulated classically by calculating matrix permanent, it will requires exponential resource.

Since there are n single photons which are divided into m inputs, the number of configurations S is:

$$|S| = \begin{pmatrix} n+m-1\\ n\end{pmatrix}$$

which is super exponential is n. Thus, with 'efficient' n of trials, we cannot to sample a given configuration more than once. It makes us not to calculate any P_S with more than binary accuracy. Thus, boson-sampling does not let us calculate matrix permanents, or it would require an exponential number of measurements to determine matrix permanent with high accuracy.

6 FURTHER DISCUSSIONS FOR THE TOPIC.

While this paper has attempted to provide an approachable, yet accurate, intuition and understanding behind how quantum computers work and what problems quantum computers excel at, it recognizes that there is still much work to be done in this area. Further areas of research could include: interactive educational media; expanding on more types of algorithms, we only discusses the HSP and Boson Sampling and there are a few other very interesting areas of algorithms that can provide further illumination into how quantum computers work; more comprehensive motivating examples; and more accurate–yet no less understandable–intuitive concepts.

A small meta note about definitions: This paper is being presented inside a 400/500 level quantum computing course; as such, in the interest of saving space, an introductory section going over basic definitions has been omitted. Were this to be submitted as a potential learning resource for a layperson, introductory materials for basic concepts (eg superposition, promises, structure) would be included; these introductory materials would include the definition, some motivating examples, intuition building, and more as needed. That being said, wherever convenient, an attempt at introductory and motivating context has been provided as a potential example for future work.

7 CONCLUSION

The HSP is an incredibly general framework that is one of the prime examples of being able to exploit superposition in quantum mechanics in order to achieve dramatic speedups in quantum algorithms. The other way that one might achieve dramatic speedups is through the fact that quantum computes, being natively quantum in behavior, can efficiently simulate quantum phenomena; this is the main source from which Boson Sampling draws its computational speedup from.

Boson sampling is a so-called sampling problem whereby the goal is sample a statistical distribution using finite number of measurements. Thus, finding a computational application is further complicated. There is non-existing application for Boson sampling. Boson sampling is just an interesting proof-of-principle demonstration that can be an example of passive linear optics' outperform over classical computers. Finally, it does not solve a problem of practical interest.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Song for providing an excellent and engaging opportunity to learn about Quantum Computing.

REFERENCES

[1] An Introduction to Boson-Sampling. (????). https://www.researchgate.net/publication/263471405_An_Introduction_to_Boson-Sampling

[2] Scott Aaronson. 2007. Shor, I'll do it. (Feb. 2007). http://www.scottaaronson.com/blog/?p=208

[3] Scott Aaronson. 2008. The Limits of Quantum Computers. *Scientific American* 298, 3 (March 2008), 62–69. https://doi.org/10.1038/scientificamerican0308-62

[4] Scott Aaronson. 2011. Scott Aaronson: Quantum Computing Promises New Insights. *The New York Times* (Dec. 2011). http://www.nytimes.com/2011/12/06/science/scott-aaronson-quantum-computing-promises-new-insights.html

[5] Scott Aaronson. 2014. When Exactly Do Quantum Computers Provide a Speedup? (Dec. 2014). http://www.scottaaronson.com/talks/speedup-austin.ppt

[6] Scott Aaronson and Andris Ambainis. 2009. The Need for Structure in Quantum Speedups. *arXiv:0911.0996 [quant-ph]* (Nov. 2009). http://arxiv.org/abs/0911.0996 arXiv: 0911.0996.

[7] Scott Aaronson and Alex Arkhipov. 2010. The Computational Complexity of Linear Optics. *arXiv:1011.3245 [quant-ph]* (Nov. 2010). http://arxiv.org/abs/1011.3245 arXiv: 1011.3245.

[8] Alastair A. Abbott. 2012. The Deutsch-Jozsa Problem: De-quantisation and Entanglement. *Natural Computing* 11, 1 (March 2012), 3–11. https://doi.org/10.1007/s11047-011-9276-7 arXiv: 0910.1990.

[9] Dave Bacon and Wim van Dam. 2010. Recent progress in quantum algorithms. *Commun. ACM* 53, 2 (Feb. 2010), 84–93. https://doi.org/10.1145/1646353.1646375

[10] Matthew A. Broome, Alessandro Fedrizzi, Saleh Rahimi-Keshari, Justin Dove, Scott Aaronson, Timothy Ralph, and Andrew G. White. 2013. Photonic Boson Sampling in a Tunable Circuit. *Science* 339, 6121 (Feb. 2013), 794–798. https://doi.org/10.1126/science.1231440 arXiv: 1212.2234.

[11] Jacques Carolan, Jasmin D. A. Meinecke, Pete Shadbolt, Nicholas J. Russell, Nur Ismail, Kerstin WÃűrhoff, Terry Rudolph, Mark G. Thompson, Jeremy L. O'Brien, Jonathan C. F. Matthews, and Anthony Laing. 2014. On the

experimental verification of quantum complexity in linear optics. *Nature Photonics* 8, 8 (July 2014), 621–626. https://doi.org/10.1038/nphoton.2014.152 arXiv: 1311.2913.

[12] Andrew M. Childs and Wim van Dam. 2010. Quantum algorithms for algebraic problems. *Reviews of Modern Physics* 82, 1 (Jan. 2010), 1–52. https://doi.org/10.1103/RevModPhys.82.1 arXiv: 0812.0380.

[13] Mark Ettinger, Peter Hoyer, and Emanuel Knill. 2004. The quantum query complexity of the hidden subgroup problem is polynomial. *Inform. Process. Lett.* 91, 1 (July 2004), 43–48. https://doi.org/10.1016/j.ipl.2004.01.024 arXiv: quant-ph/0401083.

[14] C. Gogolin, M. Kliesch, L. Aolita, and J. Eisert. 2013. Boson-Sampling in the light of sample complexity. *arXiv:1306.3995 [quant-ph]* (June 2013). http://arxiv.org/abs/1306.3995 arXiv: 1306.3995.

[15] Richard Jozsa. 2001. Quantum factoring, discrete logarithms and the hidden subgroup problem. *Computing in Science & Engineering* 3, 2 (April 2001), 34–43. https://doi.org/10.1109/5992.909000 arXiv: quant-ph/0012084.

[16] Greg Kuperberg. 2003. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *arXiv:quant-ph/0302112* (Feb. 2003). http://arxiv.org/abs/quant-ph/0302112 arXiv: quant-ph/0302112.

[17] Greg Kuperberg. 2011. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *arXiv:1112.3333 [quant-ph]* (Dec. 2011). http://arxiv.org/abs/1112.3333 arXiv: 1112.3333.

[18] Xijia Miao. 2011. The universal quantum driving force to speed up a quantum computation – The unitary quantum dynamics. *arXiv:1105.3573 [quant-ph]* (May 2011). http://arxiv.org/abs/1105.3573 arXiv: 1105.3573.

[19] Ashley Montanaro. 2016. Quantum algorithms: an overview. *npj Quantum Information* 2, 1 (Nov. 2016). https://doi.org/10.1038/npjqi.2015.23 arXiv: 1511.04206.

[20] Cristopher Moore, Alexander Russell, and Leonard J. Schulman. 2005. The Symmetric Group Defies Strong Fourier Sampling: Part I. *arXiv:quant-ph/0501056* (Jan. 2005). http://arxiv.org/abs/quant-ph/0501056 arXiv: quant-ph/0501056.

[21] Michele Mosca. 2008. Quantum Algorithms. *arXiv:0808.0369 [quant-ph]* (Aug. 2008). http://arxiv.org/abs/0808.0369 arXiv: 0808.0369.

[22] Anargyros Papageorgiou and Joseph F. Traub. 2013. Measures of quantum computing speedup. *Physical Review A* 88, 2 (Aug. 2013). https://doi.org/10.1103/PhysRevA.88.022316 arXiv: 1307.7488.

[23] Peter P. Rohde and Timothy C. Ralph. 2012. Error tolerance of the BosonSampling model for linear optics quantum computing. *Physical Review A* 85, 2 (Feb. 2012). https://doi.org/10.1103/PhysRevA.85.022332 arXiv: 1111.2426.

[24] N. Spagnolo, C. Vitelli, M. Bentivegna, D. J. Brod, A. Crespi, F. Flamini, S. Giacomini, G. Milani, R. Ramponi, P. Mataloni, R. Osellame, E. F. Galvao, and F. Sciarrino. 2014. Efficient experimental validation of photonic boson sampling against the uniform distribution. *Nature Photonics* 8, 8 (June 2014), 615–620. https://doi.org/10.1038/nphoton.2014.135 arXiv: 1311.1622.

[25] Max Tillmann, Borivoje DakiÄĞ, RenÃľ Heilmann, Stefan Nolte, Alexander Szameit, and Philip Walther. 2013. Experimental Boson Sampling. *Nature Photonics* 7, 7 (May 2013), 540–544. https://doi.org/10.1038/nphoton.2013.102 arXiv: 1212.2240.

[26] Wim van Dam and Yoshitaka Sasaki. 2012. Quantum algorithms for problems in number theory, algebraic geometry, and group theory. *arXiv:1206.6126 [quant-ph]* (June 2012). http://arxiv.org/abs/1206.6126 arXiv: 1206.6126.

[27] FrÃľdÃľric Wang. 2010. The Hidden Subgroup Problem. *arXiv:1008.0010 [quant-ph]* (July 2010). http://arxiv.org/abs/1008.0010 arXiv: 1008.0010.

[28] Hui Wang, Yu He, Yu-Huai Li, Zu-En Su, Bo Li, He-Liang Huang, Xing Ding, Ming-Cheng Chen, Chang Liu, Jian Qin, Jin-Peng Li, Yu-Ming He, Christian Schneider, Martin Kamp, Cheng-Zhi Peng, Sven HÃűfling, Chao-Yang Lu, and Jian-Wei Pan. 2017. High-efficiency multiphoton boson sampling. *Nature Photonics* advance online publication (May 2017). https://doi.org/10.1038/nphoton.2017.63

[29] John Watrous. 2008. Quantum Computational Complexity. *arXiv:0804.3401 [quant-ph]* (April 2008). http://arxiv.org/abs/0804.3401 arXiv: 0804.3401.

[30] Pawel Wocjan and Shengyu Zhang. 2006. Several natural BQP-Complete problems. *arXiv:quant-ph/0606179* (June 2006). http://arxiv.org/abs/quant-ph/0606179 arXiv: quant-ph/0606179.