Time Hierarchy theorem for Quantum Turing machines

Syed Mujtaba Hassan

December 22, 2023

Abstract

Hierarchy theorems in complexity theory show that there is some separation between complexity classes. Time and space hierarchy theorems are known for syntactical models of computation such as deterministic and nondeterministic Turing machines. For semantical models general hierarchy theorems are rare. We show that time hierarchy does indeed exist for Quantum Turing machines. A loose time hierarchy is shown for Quantum Turing machines. It is shown that a Quantum Turing machine that runs with zero error can compute more functions in time $o(nt(n)2^{t(n)}log(t(n)) + nlog(t(n)))$ than in time $t(n)$, which gives us the first time hierarchy for class **EQTIME**. A short survey of Quantum complexity theory is also provided.

1 Introduction

In computational complexity theory, time hierarchy theorems are important results that show that there is some separation between complexity classes. In other words, given more time a Turing machine can compute more functions. It shows that there doesn't exist anyone complexity class that bounds all other complexity classes. Time and space hierarchy theorems are known for syntactical models of computation such as deterministic and nondeterministic Turing machines. If $t : \mathbb{N} \to \mathbb{N}$ is a time constructible function, then for deterministic Turing machines the tightest Time hierarchy theorem that we have is $\mathsf{DTIME}(t(n)) \subset \mathsf{DTIME}(o(t(n)log(t(n))))$, while for its nondeterministic counterpart we have that $\mathsf{NTIME}(t(n)) \subset \mathsf{NTIME}(o(t(n+1)))$ (The proper definitions of these terms is given in preliminaries and section 4). For semantical models general hierarchy theorems are rare. I show that time hierarchy does indeed exist for Quantum Turing machines. I propose the first general time hierarchy theorem Quantum Turing machines model proposed by Bernstein and Vazirani. I show that a Quantum Turing machine that runs with zero error can solve more problems given more time, specifically $\mathsf{EQTIME}(t(n)) \subset \mathsf{EQTIME}(O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n))))$. This however is a much looser hierarchy than one that exists for deterministic and nondeterministic Turing machines.

Preliminaries

Throughout this paper I will use notation analogous to the one used by Micheal Sipser [17]. A Turing machine (TM) is a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$ where Q is a finite set of states, Σ is the set of input alphabets, Γ is the set of tape alphabets, where $\Gamma \in \Sigma$, $\delta : Q \times \Sigma \to Q \times \Sigma \times \{L, R\}$, $q_0 \in Q$ is the start state, $q_a \in Q$ is the accept state and $q_r \in Q$ is the reject state [17]. While for a nondeterministic turing machine $\delta : Q \times \Sigma \to P(Q \times \Sigma \times \{L, R\})$ [17].
For a complex number c, c^* denotes its complex conjugate. For a countable set D, let $l_2(D)$ be the space of all complex values functions on D bounded by the l_2 norm. $l_2(D) = \{x : D \to \mathbb{C} | \sqrt{\sum_{i \in D} x(i)x^*(i)} < \infty\}$ [9][8]. The inner product $\langle . | . \rangle$ is defined as $\langle a|b \rangle = \sum_{i \in D} a^*(i)b(i)$, for $a, b \in l_2(D)$ [8]. For an $n \times n$ matrix $U \in \mathbb{C}^{n \times n}$, $U^\dagger$ denote the conjugate transpose of U [22]. U is called unitary if $UU^\dagger = U^\dagger U = I$ [22].

A function $t : \mathbb{N} \to \mathbb{N}$ is called time constructible if $t(n) \geq n$ and there exits a TM that on input 1^n computes the binary representation of $t(n)$ in $O(t(n))$ time [1][17]. The set $\widetilde{\mathbb{C}}$ denote the set of tractable complex numbers, which is the set of complex numbers where the n^{th} digit of their real and imaginary part can be computed in polynomial time with a deterministic Turing machine [22].

2 Quantum Turing machine

The idea of a Quantum Turing machine (QTM) was first introduced by Benioff in 1980 [2][3]. The idea was later formalized by Deutsch in 1985 [6]. Later Bernstein and Vazirani introduced their own formalism for Quantum Turing machine which is generally known as BV-QTM [4][9][22]. The Quantum Turing machine model most widely used today is the one by Bernstein and Vazirani [8]. We define a Quantum Turing machine as analogous to a deterministic Turing machine. For defining our Quantum Turing machine we use a formalism similar to the one used by Micheal Sipser for a deterministic Turing machine in [17].

Definition 1. *A Quantum Turing machine (QTM) M is a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$ where Q is a finite set of states, Σ is the set of input alphabets, Γ is the set of tape alphabets, where $\Gamma \in \Sigma$, $\delta : Q \times \Sigma \times Q \times \Sigma \times \{L, R\} \to \widetilde{\mathbb{C}}$, $q_0 \in Q$ is the start state, $q_a \in Q$ is the accept state and $q_r \in Q$ is the reject state.*

We now seek to define computation by a Quantum Turing machine. Let C_M denote the set of all configurations of a QTM M. Computation of M is performed in the inner-product space $H_M = l_2(C_M)$ with the basis $C_B = \{|c\rangle \,|c \in C_M\}$ [4][8]. The set C_B is called the computational basis [9]. The transition function gives us mapping $a : C_M \times C_M \to \mathbb{C}$. The time evolution $U_M : H_M \to H_M$ operator is then defined as:

1. If $|c\rangle$ is a basis state then
$$U_M |c\rangle = \sum_{c' \in C_M} a(c, c') |c'\rangle$$
2. If $|\psi\rangle = \sum\limits_{c \in C_M} \alpha_c |c\rangle$ is a superposition then
$$U_M |\psi\rangle = \sum_{c \in C_M} \alpha_c U_M |c\rangle$$

At every time step the machine is in a superposition of configurations which is denoted by $|\psi\rangle$. The computation by a Quantum Turing machine is done by application of the time evolution operator U_M on $|\psi\rangle$ [9][4][22][8]. It is to note that if computation is not time or space bounded then U_M is of infinite dimension. What is to note here is that the halting protocol for the Quantum Turing machine is not well defined by just this formalism. There are many proposed halting protocols for Quantum Turing machine, such as by Deutsch [6], Bernstein and Vazirani [4], and Ozawa [15], among others. The issue with defining the halting protocol for a Quantum Turing machine is that upon measurement of the state of the machine at any time step the superposition collapses thus spoiling the computation. In this paper we use the halting protocol used by Abuzer Yakaryilmaz1 and A.C. Cem Say in [21]. A finite register is built in a Quantum Turing machine. At each time step, each configuration reads the finite register if the configuration is of a non-halting state and if C is not already written in the register then the machine writes C in the register. If the configuration is of a halting state, and if H is not already written in the register then the machine writes H in the register. After each time step, only the contents of the register are observed (thus not spoiling the computation) if C is written in the register then all the register contents are erased and the computation continues. Else the machine is halted and measured. This gives a halting condition for a Quantum Turing machine. Now to define a Quantum Turing machine we have to make sure the operator U_M defined by the transition function is unitary. Ozawa and Nishimura showed that the following conditions on the transition function are necessary and sufficient for the corresponding operator U_M to be unitary [14][8][15]:

1. $\forall (q,a) \in Q \times \Sigma$,
$$\sum_{(q',a',d)\in Q\times\Sigma\times\{L,R\}} |\delta(q,a,q',a',d)|^2 = 1$$

2. $\forall (q_1,a_1),(q_2,a_2) \in Q \times \Sigma$ where $(q_1,a_1) \neq (q_2,a_2)$,
$$\sum_{(q',a',d)\in Q\times\Sigma\times\{L,R\}} \delta(q_2,a_2,q',a',d)^* \delta(q_1,a_1,q',a',d) = 0$$

3. $\forall (q_1,a_1,a_1'),(q_2,a_2,a_2') \in Q \times \Sigma \times \Sigma$,
$$\sum_{q'\in Q} \delta(q_2,a_2,q',a_2',R)^* \delta(q_1,a_1,q',a_1',L) = 0$$

4. $\forall (q_1,a_1,a_1'),(q_2,a_2,a_2') \in Q \times \Sigma \times \Sigma$,
$$\sum_{(q',d)\in Q\times\{L,R\}} \delta(q_2,a_2,q',a_2',d\in\{L,R\}\backslash\{d\})^* \delta(q_1,a_1,q',a_1',d) = 0$$

So we have that in order to define a Quantum Turing machine we have to make sure that our defined transition function follows these constraints. However, it has been shown that all our usual programming primitives such as iterating, if conditions, etc all satisfy these conditions so we can program and define a Quantum Turing machine the way we define and program deterministic and nondeterministic Turing machine [4][8].

3 Universal Quantum Turing machine

The Turing machine proposed by Alan Turing in [19] is shown to be equivalent to all other physically realizable models of computation [17][5][20][16]. The Church-Turing Thesis says that all sufficiently complex physically realizable models of computation are equivalent [17][22]. With all that the Turing machines are used as the standard model of computation. The major advantage the Turing machine bought was the existence of a universal Turing machine. That is a Turing machine that can simulate any given Turing machine. The existence of a universal Turing machine is a very important result for computability and complexity theory. Many important theorems in the area such as the proof for undecidability of the Halting problem and the proof for the time hierarchy theorem for deterministic and nondeterministic Turing machine are based on the existence of a universal Turing machine. For deterministic Turing machine, we have that there exists a universal Turing machine U such that on input $\langle M, w\rangle$ where M is a QTM and $w \in \Sigma^*$, U simulates $M(w)$ and is T is the time M takes to decide w then U runs in $TlogT$ time. In order to show that time hierarchies exist for Quantum Turing machines we need a similar universal machine for Quantum Turing machines. Bernstein and Vazirani proved that a universal Quantum Turing machine does exist [4]. The universal Quantum Turing machine defined by Bernstein and Vazirani isn't defined the same way as a universal deterministic Turing machine is. Bernstein and Vazirani's Quantum Turing machine simulates a given Quantum Turing machine for a specified T time steps with a specified error ϵ. The specification of T and ϵ comes from the fact that in non-time and space-bounded computation the time evolution operator U_M is of infinite dimension so U_M is not precisely constructible. The following theorem is given by Bernstein and Vazirani in [4].

Theorem 1. *There exists a universal Quantum Turing machine U such that on input $\langle M, w, T, \epsilon\rangle$ where M is a QTM and $w \in \Sigma^*$, $T \in \mathbb{N}$, $\epsilon \in \mathbb{R}$ where $\epsilon > 0$. U simulates $M(w)$ for T time steps with precision ϵ and U runs in time polynomial to $|M|$, $|W|$, T and $1/\epsilon$.*

This theorem does show the existence of a universal Quantum Turing machine but the defined universal Quantum Turing machine isn't sufficient to construct a diagonalization argument. Other universal Quantum Turing machines have been proposed such as one by Deutsch in [6] and by Markus Muller in [13], however, both of these suffer from the same issue as Bernstein and Vazirani's universal Turing machine. The issue is imprecision and the fact that the complexity of the universal machine is dependent on the precision specified. A good survey of universal Quantum Turing machines is given in [7]. In order to show that a time hierarchy exists for Quantum complexity classes we need a machine that works analogous to its deterministic counterpart. One can say we need "determinism" to apply diagonalization.

4 complexity classes

We now discuss some complexity classes associated with Quantum Turing machines. First, we shall first define some complexity classes defined for deterministic and nondeterministic Turing machines. Let $t : \mathbb{N} \to \mathbb{N}$ be a time constructible function, then $\textsf{\textbf{DTIME}}(t(n)) = \{L \subseteq \Sigma^*|$ there exists a deterministic TM M such that M decides L and M runs in $t(n)$ time $\}$. And $\textsf{\textbf{NTIME}}(t(n)) = \{L \subseteq \Sigma^*|$ there exists a nondeterministic TM M such that M decides L and M runs in $t(n)$ time $\}$. With these definitions, we define our class **P**, class **NP**, and class **EXP**.

$$\textsf{\textbf{P}} = \bigcup_{c\geq 1} \textsf{\textbf{DTIME}}(n^c)$$

$$\textsf{\textbf{NP}} = \bigcup_{c\geq 1} \textsf{\textbf{NTIME}}(n^c)$$

$$\textsf{\textbf{EXP}} = \bigcup_{c\geq 1} \textsf{\textbf{DTIME}}(2^{n^c})$$

$$\textsf{\textbf{EXP}} = \bigcup_{c\geq 1} \textsf{\textbf{DTIME}}(2^{n^c})$$

$$\textsf{\textbf{NEXP}} = \bigcup_{c\geq 1} \textsf{\textbf{NTIME}}(2^{n^c})$$

With these deterministic and nondeterministic complexity classes, we now seek to define our complexity classes for Quantum Turing machines. We are mainly concerned with two types of complexity classes for Quantum Turing machines, one is the class of languages decided by a machine with no error and the other is the class of languages decided by the machine with a small probability of error.

Definition 2. *Let $t : \mathbb{N} \to \mathbb{N}$ be a time-constructible function.* $\textsf{\textbf{EQTIME}}$*$(t(n)) = \{L \subseteq \Sigma^*|$ There exists a QTM M such that M decides L with zero error and M runs in $t(n)$ time $\}$.*

Definition 3. *Let $t : \mathbb{N} \to \mathbb{N}$ be time time-constructible function.* $\textsf{\textbf{BQTIME}}$*$(t(n)) = \{L \subseteq \Sigma^*|$ There exists a QTM M such that $x \in L \implies$ Prob(M accepts x) $> \frac{2}{3}$ and $x \notin L \implies$ Prob(M rejects x) $> \frac{2}{3}$ and M runs in $t(n)$ time $\}$*

With these, we shall define the complexity classes that are of interest to us and that is:

$$\textsf{\textbf{EQP}} = \bigcup_{c\in\mathbb{Z}^+} \textsf{\textbf{EQTIME}}(n^c)$$

$$\textsf{\textbf{BQP}} = \bigcup_{c\in\mathbb{Z}^+} \textsf{\textbf{BQTIME}}(n^c)$$

These define the polynomial complexity classes for Quantum Turing machines. We have that the following relationship holds for these complexity classes [1][22].

$$\textsf{\textbf{P}} \subseteq \textsf{\textbf{EQP}} \subseteq \textsf{\textbf{BQP}} \subseteq \textsf{\textbf{EXP}}$$

An important question here is if $\textsf{\textbf{P}} = \textsf{\textbf{EQP}}$. An interesting result showed by Jonah Librande in [18] is that $\textsf{\textbf{BQP}} \not\subset \textsf{\textbf{NP}}$. This actually shows the power of a Quantum Turing machine over a deterministic Turing machine as this shows that $\textsf{\textbf{P}} \subset \textsf{\textbf{BQP}}$, as $\textsf{\textbf{P}} \subseteq \textsf{\textbf{NP}}$.

5 Time Hierarchy theorem

We now seek to show that a time hierarchy exists for Quantum Turing machines. We don't have general time hierarchy theorems for most semantic models. In [12] it has been shown that a tight time hierarchy exists for semantic models that take an advice bit. This gives us some sort of Time hierarchy theorem for Quantum Turing machines, however this is not good enough. This only shows that time hierarchy exists for the Quantum Turing machine model that takes an advice bit which is not the standard Quantum Turing machine model that is used. It also increases the input size of the machine. In [21] it has been shown that time hierarchies exist for nondeterministic Quantum Turing machines which are a variant of Quantum Turing machines, but they are not the standard variant of Quantum Turing machines that is generally used. These are similar to how nondeterministic Turing machines are a counterpart of deterministic Turing machines. So we didn't have a general time hierarchy theorem for Quantum Turing machines analogous to the one we have for deterministic and nondeterministic Turing machines, and the existence of such time hierarchies for Quantum Turing machines has been an open problem for many years. We now show that time hierarchy does indeed exist for Quantum Turing machines. We go on and prove the main theorem of our paper. As we noted earlier the current construction of a universal Quantum Turing machine that we have is not sufficient to apply diagonalization and find a time hierarchy theorem for a Quantum Turing machine. For the Quantum Turing machine model the approach of "flipping the output bit" cannot be applied with the universal Quantum Turing machine that we have. Instead, we use an approach similar to the one taken to devise the time hierarchy theorem for a nondeterministic Turing machine. That is we simulate our Quantum Turing machine "deterministically". With this idea we have the following theorem.

Theorem 2. *Let $t : \mathbb{N} \to \mathbb{N}$ be a time constructible function then,*

$$\textsf{\textbf{EQTIME}}(t(n)) \subset \textsf{\textbf{EQTIME}}(O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n))))$$

Proof. We show that there exists a language $L \in \textsf{\textbf{EQTIME}}(O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n))))$ which is not in $\textsf{\textbf{EQTIME}}(t(n))$. We construct a QTM M that decides a language L and runs in time $O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n)))$ and no QTM that runs in time $t(n)$ can decide L.
Let $M = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$ be a QTM. M is a Quantum Turing machine that only performs deterministic steps. This can be constructed easily by defining the transition function of M such that $\forall q \in Q, \forall a \in \Sigma$ there exists a unique $q' \in Q$, $a' \in \Sigma$ and $d \in \{L, R\}$ such that $\delta(q, a, q', a', d) = 1$ and $\forall \overline{q} \in Q \backslash \{q, q'\}$ $\forall \overline{q} \in \Sigma \backslash \{a, a'\}$ $\forall \overline{d} \in \{L, R\} \backslash \{d\}$, $\delta(q, a, \overline{q}, \overline{a}, \overline{d}) = 0$. M simulates a given QTM M' on some input w for some T steps and α precision bits of amplitude as follows: M keeps a counter of T which is updated after performing each time evolution of M' M initialize the initial configuration of M' on w then for each time step M hold all the configurations of M' with nonzero amplitudes in the superposition $|\psi\rangle$ of M' on w at some i^{th} time step and their corresponding amplitude. To perform the time evolution M scans the transition function of M' each configuration in $|psi\rangle$ and computes the resulting configurations and amplitudes and writes them on the tape. Now M is defined as follows:
M = "On input M' where M' is a QTM:

1. Compute $t(|M'|)$.
2. Simulate $M'(M')$ for $t(|M'|)$ steps for precision α.
3. Scan the tape for distribution of amplitudes of configurations of M'. If each configuration with nonzero amplitude is a rejecting configuration then output 1, else output 0."

Now as we are only concerned with configuration of nonzero amplitude in the distribution of $M'(M')$ after $t(|M'|)$ time steps rather than the actual computation output of $M'(M')$ only using α bits of precision is sufficient as the lower bits gets canceled with each other on zero amplitude and nonzero amplitude we are not concerned with the actual amplitude rather than just the fact that its nonzero. Now for each time step T the size of $|psi\rangle$ superposition of configurations of M' is bounded by $\alpha|M'|T2^T$. The time to compute each $t(n)$ is bounded by $log(t(n))$ as $t(n)$ is time constructible. So for input of size n, M runs in time:

$$\sum_{i=1}^{t(n)} \alpha nt(n)2^{t(n)}log(t(n)) = \alpha n((t(n)-1)2^{t(n)+1}+2)log(t(n))$$

So M runs in $O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n)))$. Let L be the language of M, M decides L with zero error as M essentially works deterministically.
Now suppose there exits a QTM D that decides L with zero error and D runs in $t(n)$ time. Now as $L(M) = L(D)$, $\forall w \in \Sigma^*, M(w) = D(w)$. But for $D(D)$, if $D(D) = 0$ then $M(D) = 1$ as D runs in zero error so the final distribution of $D(D)$ would have all configurations of nonzero amplitude have the same state. And if $D(D) = 1$ then $M(D) = 0$. As D is a decider and D runs in $t(n)$ time D halts on each simulation M of D. And as D has zero probability of error D's output on each input is the same every time and final distribution of D on each input would have all configurations of nonzero amplitude have the same state. So now we have that $\exists w \in \Sigma^*$, such that $M(w) \neq D(w)$, which is a contradiction to the assumption that D decides the same language as M so such D cannot exist. So we have a language L that can be decided in $O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n)))$ time but cannot be decided in $t(n)$ time. So $L \in \textbf{EQTIME}(O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n))))$ while $L \notin \textbf{EQTIME}(t(n))$. So we have that $\textbf{EQTIME}(t(n)) \subset \textbf{EQTIME}(O(nt(n)2^{t(n)}log(t(n)) + nlog(t(n))))$. $\square$

So we proved the first general time hierarchy theorem for Quantum Turing machines.

Conclusion

We also provided a short survey of Quantum complexity theory In this paper, we proved the first general time hierarchy theorem for Quantum Turing machines. However, the current Hierarchy is quite loose compared to the one for deterministic and nondeterministic Turing machines. This was the case when the first time hierarchy was introduced for deterministic Turing machines by Juris Hartmanis and Richard Edwin Stearns in [10], and was later improved by Hennie and Stearns in [11]. So improving this hierarchy is one possible area of future work. Ideally finding a tighter time hierarchy would be the goal for some future work. Another area of Future work can be to devise a time hierarchy theorem for the **BQTIME** class. In this paper, we also provided a short survey of Quantum Complexity theory. Quantum Complexity theory is still a new field in which many aspects are still not defined. Which gives us a lot of area of research to be done in the field. The issue is understanding Quantum Turing machines and computation by this model is itself quite difficult as our brains are used to thinking classically. A better intuition of Quantum processes and well definitions of our Quantum computing structures would help us better understanding the area.

References

[1] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, USA, 1st edition, 2009.

[2] Paul Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of Statistical Physics*, 22:563–591, 05 1980.

[3] Paul Benioff. Quantum mechanical hamiltonian models of turing machines. *Journal of Statistical Physics*, 29:515–546, 11 1982.

[4] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997.

[5] Alonzo Church. An unsolvable problem of elementary number theory. *Journal of Symbolic Logic*, 1(2):73–74, 1936.

[6] David Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400:117 – 97, 1985.

[7] Willem Fouché, Johannes Heidema, Glyn Jones, and Petrus H. Potgieter. Deutsch's universal quantum turing machine (revisited), 2007.

[8] J. Gruska. *Quantum Computing.* Advanced topics in computer science series. McGraw-Hill, 1999.

[9] Stefano Guerrini, Simone Martini, and Andrea Masini. Quantum turing machines computations and measurements. *Applied Sciences*, 10, 03 2017.

[10] Juris Hartmanis and Richard Edwin Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[11] F. C. Hennie and R. E. Stearns. Two-tape simulation of multitape turing machines. *J. ACM*, 13(4):533–546, oct 1966.

[12] Dieter Melkebeek and Konstantin Pervyshev. A generic time hierarchy with one bit of advice. *Computational Complexity*, 16:139–179, 01 2007.

[13] Markus Müller. Strongly universal quantum turing machines and invariance of kolmogorov complexity. *Information Theory, IEEE Transactions on*, 54:763 – 780, 03 2008.

[14] Harumichi Nishimura and Masanao Ozawa. Computational complexity of uniform quantum circuit families and quantum turing machines, 2000.

[15] Masanao Ozawa. On the halting problem for quantum turing machines. 01 1998.

[16] Rózsa Péter and Stephen Cole Kleene. λ-definability and recursiveness. *Journal of Symbolic Logic*, 2:38, 1937.

[17] Michael Sipser. *Introduction to the Theory of Computation.* International Thomson Publishing, 1st edition, 1996.

[18] A. M. Turing. Computability and λ-definability. *The Journal of Symbolic Logic*, 2(4):153–163, 1937.

[19] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 01 1937.

[20] Alan Turing. Computability and λ-definability. *Journal of Symbolic Logic*, 2(4):153–163, 1937.

[21] Abuzer Yakaryilmaz and A. C. Cem Say. Proving the power of postselection. *Fundamenta Informaticae*, 123(1):107–134, 2013.

[22] Noson S. Yanofsky and Mirco A. Mannucci. *Quantum Computing for Computer Scientists*. Cambridge University Press, 2008.