quantum reports

Article

Undecidable, Unrecognizable, and Quantum Computing

Michael Siomau

Dubai Men's Campus, Higher Colleges of Technology, Dubai 15825, UAE; msiomau@hct.ac.ae

Received: 20 May 2020; Accepted: 30 June 2020; Published: 1 July 2020

Abstract: Quantum computing allows us to solve some problems much faster than existing classical algorithms. Yet, the quantum computer has been believed to be no more powerful than the most general computing model—the Turing machine. Undecidable problems, such as the halting problem, and unrecognizable inputs, such as the real numbers, are beyond the theoretical limit of the Turing machine. I suggest a model for a quantum computer, which is less general than the Turing machine, but may solve the halting problem for any task programmable on it. Moreover, inputs unrecognizable by the Turing machine can be recognized by the model, thus breaking the theoretical limit for a computational task. A quantum computer is not just a successful design of the Turing machine as it is widely perceived now, but is a different, less general but more powerful model for computing, the practical realization of which may need different strategies than those in use now.

Keywords: quantum computing; quantum information; theory of computation

1. Introduction

In a very broad sense, a quantum computer is a computational machine which operates according to quantum theory. Although such statements can be found in all quantum computing textbooks [1–3], they often come with little elaboration of the meaning. With no waste of time, the textbooks tell us that the parallel computation powered by the quantum superposition is recognized as the key feature of the quantum computer, leading us to the well-promoted notion of quantum parallelism. If this statement is evaluated from the viewpoint of Theoretical Computer Science, it is evident that the quantum computer is just a non-deterministic Turing machine (TM)—a variation of the TM, which allows parallel computation [4]. This simple analogy naturally leads us to the conclusion that the quantum computer, although it may be more efficient, cannot be more powerful than the TM, which is of course plausible, but essentially is a particular view on what a quantum computer could be.

If we choose to look at the quantum computer from the perspective of Theoretical Computer Science, then some points are not clear from the explanations above. Any non-deterministic TM has a deterministic equivalent [5]. It is natural to ask what is the deterministic TM-equivalent of a quantum computer? Also, the TM is an abstract computational model unbounded by any physical law. How the laws of the quantum theory bound the rules and change the logic of computation?

Systematic attempts to define a deterministic quantum TM have been done over the years [6–9], leading to interesting modifications of the TM, with subtle halting rules and artificially introduced finite sets of complex numbers to place proper unitary transition function [9].

In this paper, I'll go back to the first sentence of this paragraph and develop a different path toward the understanding of quantum computers through the TM model. I will go back to the origin of the TM to see what it can do and what it cannot, and what quantum theory changes. To avoid the confusion with the existing views on quantum computing, I will use the notion of a quantum computational machine, or simply the quantum machine (QM), instead of a quantum computer. Also, in this paper I'll consider only deterministic models for computation to underscore the difference with the quantum parallelism.

2. Results

A TM has an infinite tape with countable number of cells and a tape head that can read and write symbols and move around the tape according to the list of rules specified by the transition function

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}\,, \tag{1}$$

where Q is a set of the machine's states, Γ is the tape alphabet and $\{L, R\}$ defines whether to move the tape head to the left or to the right at each step. If the tape head is positioned over the k-th cell of the tape, the TM is in the state q_l and the current symbol on the tape is γ_m the TM is in configuration $TM(k, q_l, \gamma_m)$, where $k \in N$, $q_l \in Q$ and $\gamma_m \in \Gamma$. Current configuration determines the next configuration and the tape head move according to the transition function. If an input to the TM can be described by the tape alphabet, we say that the TM recognizes the input. On the other hand for some given alphabet, we may design a TM that recognizes it. Here, it is important to note that a TM can be constructed only for finite or countable alphabets. If the alphabet is uncountable, such as the set of real numbers, there is no TM that recognizes it [5].

If we demand that a TM runs according to quantum theory, then we probably should restrict the δ transitions to unitary operations, so that for each configuration the interaction between the tape cell and the tape head is quantum. What difference does this restriction make? For simplicity, let me further restrict the class of the unitary operations to the controlled-unitary, so that the machine's states act as the control and the tape symbols are the subordinates. For a moment we may even assume that both sets Q and Γ are classical. Such a machine may be called a QM with classical input (QMCI). To understand how much this machine and the TM are different in their computational power, we need to compare the sizes of the sets of recognizable inputs. The QMCI is substantially less powerful than the TM, because it recognizes substantial less inputs. Indeed, an input $(abracadabra)$ is not recognizable by the QMCI, because there is no way to define the controlled-unitary operations for the corresponding English alphabet. Surprisingly, the restriction of the unitary interactions between the tape cell and the tape head suffices to solve the halting problem for an arbitrary input.

It is time to explain what the halting problem is. Suppose we are given some program, a TM and an input. By looking at the program and the input, can we decide if this program makes some meaningful computation in finite time or just do some trivial operation forever, like moves the tape head right at each step. More formally, given the list of the δ transitions for some TM and an input, the halting problem is to decide if the TM ever stop the computation or run forever. It is usually assumed that the stop condition is well defined, so that the TM does not run on a loop of infinite size, i.e., when the tape head simply runs far far away never coming back. The real problem is to detect a finite loop, i.e., when a TM makes seemingly meaningful computation over a finite tape segment never meeting the stop condition [5,10]. If at some point of the computation the TM has configuration $TM(k, q_l, \gamma_m)$ and after some steps of the computation the TM returns to this configuration and all the symbols over the tape are exactly the same as during the first instance, the TM will go the same path again and again never halting.

While all repeating TM configurations can be recorded on the infinite tape, there is no way to check if the tape symbols are left the same on the way upon returning to the configuration. The tape head may visit each cell many times, while the TM may be each time in a different state, meaning that different transitions will be applied in a different order depending on the current symbol on the tape. In other words, it is not possible to predict the cumulative effect of the δ transitions on a chosen cell.

But, for the QMCI the above task is possible due to the properties of the unitary operators. Suppose that $U = \{u_i\}$, $i = 1..n$ is a set of unitary operations of the transition function δ. Any member of the set may be applied to any cell of the tape arbitrary number of times. Since any unitary operation can be represented through the generators of the corresponding special unitary group, an arbitrary sequence of unitary operations applied to a chosen cell may be computed efficiently whenever the decomposition coefficients of the unitary operation are rational numbers. Iff the resulting unitary operation is equal to the identity matrix, the symbol on the tape remains the same irrespective of the symbol. Now, imagine that the TM is in configuration $TM(k, q_l, \gamma_m)$. After some finite number of steps the tape head moved over a segment of the tape and returned to configuration $TM(k, q_l, \gamma_m)$. If for each cell of the segment the cumulative unitary operation is identity, the TM will go over the segment again and again, thus never halting. Further insight allows us to set up an accurate criteria for detecting any finite loop (see Methods).

The fact that the QMCI can solve the halting problem is not really impressive, because the number of tasks programmable on the QMCI is very limited. But, QMCI is not an actual quantum machine—it is just a TM with a special type of δ function. Let me now consider a more general model for QM. Let the sets Q and Γ be pure states of some quantum discrete systems, and the δ transitions are general unitary operations over the joined Hilbert space $Q \otimes \Gamma$

$$\delta : Q \otimes \Gamma \xrightarrow{U} Q \otimes \Gamma \times \{L, R\} \,. \tag{2}$$

Such a QM with pure input (QMPI) can solve the self-contradicting halting problem, which is beyond the theoretical limit of the TM. The essence of the self-contradicting halting problem is to task a TM to perform a self-contradicting action, that is, to halt (H) when the input suggests to run (R), and to run (R) when the input suggests to halt (H) [5]. In contrast to TM, QMPI can do the desired task producing an entangled quantum state $|H\rangle_m |R\rangle_i + |R\rangle_m |H\rangle_i$ as the output, where $|H\rangle_m$ and $|R\rangle_m$ are machine's halt and run states, and $|H\rangle_i$ and $|R\rangle_i$ are the respective inputs (see Methods). The output literally reads as 'halt when run, and run when halt'. But, will the machine run or halt? On metaphysical level, the answer is probably the same as if you asked: 'Is the Schrödinger's cat alive or dead?' The result of the computation does not have a classical meaning, but the QMPI does deterministically the desired self-contradicting computation. From the point of view of the halting problem solution for QMCI, the QMPI halts providing the entangled state.

The QMPI would be a perfect model for QM if not a serious drawback. During its operation, the machine must be able to read the symbols on the tape. But, in quantum theory reading the state of a quantum system is not a straightforward task. Even so the input of the QMPI are pure states, which may be initially agreed to be prepared in some basis states, during the computation the states of the quantum systems on the tape may transform into complex superpositions. Measuring these states in the basis after each step of the computation will destroy the superposition breaking the rule of the unitary evolution. A way to overcome the drawback may be not to measure the states on the tape till the end of the computation. This idea is not too bad, because the halting problem solution is based only on the analysis of the transition function and is valid for an arbitrary input. Other words, we always know if the machine halts irrespective of the input or intermediate values on the tape. But, how the QM would know which of the δ transitions to use if not by reading the tape? My suggestion is to support the QM with a classical counterpart (QMCC).

Suppose we have a double tape, the first string of which consists of the classical alphabet Γ_c, and the second string is populated with some quantum systems having states Γ_q. Let me assume a strict correspondence between the strings so they form an ordered double set. Suppose also that the machine's states is also an ordered double set $Q = Q_c \times Q_q$, where Q_c are classical states and Q_q are the states of some quantum system. Then a QMCC may be defined as a unitary transformation over the double sets

$$\delta : Q_c \times Q_q \otimes \Gamma_q \times \Gamma_c \xrightarrow{U \times U} Q_c \times Q_q \otimes \Gamma_q \times \Gamma_c \times \{L, R\}. \tag{3}$$

At each step the machine reads the classical states $Q_c \times \Gamma_c$ and chooses the δ transform from the list of the δ function. Then a unitary transform is applied both on the classical part of the input $Q_c \times \Gamma_c$ and on the quantum $Q_q \otimes \Gamma_q$. The halting problem can be always resolved for this machine through the

δ function analysis irrespective of the quantum input $Q_q \otimes \Gamma_q$. When the computation is finished, the quantum states of the machine and the states of the systems on the tape are measured. Depending on a particular algorithm, the classical states on the tape may assist to construct the optimal measurements for the quantum states.

At the first glance the QMCC is equivalent to the QMCI, because any two-tape machine has a single-tape equivalent. But, in fact it is much more powerful than the QMCI and even more powerful than the TM. The difference is in the symbols written on the tape. The set of the quantum states is uncountably infinite, while only members of countably infinite sets are recognizable by the TM. A state of a quantum system may be brought into correspondence with a real number. Such numbers cannot be written on a finite tape because they have infinite many decimals [5] (see Methods). The price we have to pay for this enormous power is that the quantum output eventually has to be measured or compared with a copy of the input. The exact measurement of a quantum state is not possible. Thus the answer of the QM computation is always approximate. In practice, it means we have to run the QMCC multiple times for each input to get a set of identical outputs and then measure them. The quantity of the output copies defines the quality of the approximation. It is important to note that this deterministic procedure has nothing to do with parallel computation. The necessity for multiple runs emerges from fundamentals of quantum theory.

3. Discussion

Proposed model for a quantum computational machine is a straightforward modification of the Turing machine according to my understanding of the quantum theory. It is unlikely to outperform the parallel computation by the quantum computer in terms of efficiency of the computation. On the other hand, the quantum machine may look beyond the theoretical limit of classical computation. As the combination of the classical and quantum features sometimes leads to completely new phenomena [11–13], it could be that a hybrid quantum computer partially exploiting the advantages of the deterministic and the parallel computation may be the best choice to challenge fundamentally unsolvable problems.

From the point of view of classical computability theory, a problem is solvable if there is an efficient algorithm to solve the problem. We are likely to design many algorithms for the QM that halt for an arbitrary input and are efficient. But, the classical definition says nothing about our ability to access or understand the result of the computation. In quantum computing, we may design a program that halts, having no information about the input states of the quantum systems. Such an input is computable, which doesn't mean the output is accessible with an arbitrary precision. To make things even more bizarre, classically self-contradicting problem may be computable, but with no classical meaning. If the definition of the computability is not adjusted with respect to the quantum features, the QM contradicts the Turing-Church thesis [10], which is hard to believe due to its very limited applicability.

4. Materials and Methods

Assume we are in possession of many copies of some discrete n-dimensional quantum system. A unitary operation defined for the states of the system is a matrix with $n^2 - 1$ independent parameters. Equivalently, any unitary matrix may be decomposed in $n^2 - 1$ basis formed by $SU(n)$ generators. The coefficients of the decomposition may be complex numbers. But, such numbers cannot be written on a tape. Indeed, any rational number (as a part of a complex number) contains infinite number of decimals requiring infinite memory to write it down [5]. Thus, to write a decomposition of a unitary operator on a finite tape, we have to assume that the coefficients are rational numbers. Unitary operations with rational coefficients form a countably infinite set. Each subset of this set generates a QM.

Since, for any unitary operation u_i from the set of the unitary operations U of the transition function δ, there is a decomposition into some basis matrices with rational coefficients, any sequence of unitary operations can be computed efficiently, and the resulting unitary operation is also decomposable with rational coefficients. It is always possible to compare the coefficients of any such matrix and establish if they are equal to the coefficients of the identity matrix.

A finite loop is defined as a repeating sequence of configurations over a finite tape segment. Due to determinism, any algorithm may have at most one loop, which may be entered only once and cannot be exit. A loop may appear from the first step of the computation or after some nontrivial steps. In terms of Venn diagram, the first case of computation correspond to a bud, the second—to a stem resulting in a bud.

Suppose at some step of the computation t the QMCI is in configuration $TM(k, q_l, \gamma_m)$. During the next p steps the tape head moved over some segment of the tape and returned to the configuration $TM(k, q_l, \gamma_m)$. The deterministic QMCI machine will loop over this segment if and only if the cumulative effect of unitary operations for each cell is identity.

In practice, an algorithm to detect a finite loop may look as follows. If the tape head is observed to circulate over a segment of the tape, we may start a subroutine at arbitrary cell of the segment. We would like to keep track of the cumulative effect of the unitary operations on each cell, i.e., what is the resulting unitary operation at each step of the computation. For that we need to memorize $n^2 - 1$ rational numbers for each cell. For each cell visited by the tape head in the segment we compute the cumulative unitary. If the QMCI returned to the configuration we started with, we compere the cumulative unitaries with the identity matrix. If we have full much with the segment of identities, the QMCI loops over this segment. But, if we don't have the match, it doesn't mean there is no loop, because the loop may be nontrivial. The QMCI may have left a nontrivial trace of unitary operations over the segment, i.e., $U_1, U_2, U_3...$, where $U_i \neq I$. But, only if the QMCI loops over the segment, the second run will leave this trace twice $U_1U_1, U_2U_2, U_3U_3...$ Because $U_iU_i = I$, the cumulative effect of the unitaries will be identity for all cells of the segment. This is necessary and sufficient condition to detect a finite loop. If this condition is not fulfilled, the subroutine is aborted. This procedure remains valid both for QMPI and QMCC, because it utilizes only basic property of the unitary operations.

Quantum Machine with pure input (QMPI) can solve the self-contradicting halting problem by running the following pseudo code. Suppose the machine's states $|H\rangle_m$ and $|R\rangle_m$ are the states of qubit $|0\rangle, |1\rangle$. Let say $|H\rangle_m = |0\rangle$ and $|R\rangle_m = |1\rangle$. Let the input state is also the states of qubit $|H\rangle_i = |0\rangle$ and $|R\rangle_i = |1\rangle$. Let the transition function consists of a two unitary transformations followed by stop. Suppose the machine is set up to halt, that is, the machine's state is $|0\rangle$. If the tape symbol is $|0\rangle$, the unitary operation $U_1 = I \otimes I$ is to be applied resulting in the output $|00\rangle$, that is, halt. If, in contrast, the tape symbol is $|1\rangle$, that is, contradicting to the machine's setup, the following unitary is to be applied $U_2 = (H \otimes I)(C - NOT)$, where H is Hadamard gate, $C - NOT$ is controlled NOT gate [1]. As the result of the self contradicting task, the machine becomes entangled with the tape. If for pure input states, the machine generates entanglement between the states of the machine and the tape, it maybe an indication of self-contradiction in the algorithm provided.

In general a quantum state of a qubit may be written in the computational basis $|0\rangle, |1\rangle$ as $\alpha |0\rangle + \beta |1\rangle$, where α and β are complex numbers. The absolute value $|\alpha|$ can be measured giving a rational number between 0 and 1, while $\frac{|\alpha|}{|\beta|}$ for $|\beta| \neq 0$ is an arbitrary rational number. This number cannot be measured exactly with a finite number of measurements. But the advantage comparing to the TM is in its exact representation during the computation. QM performs the computation on the rational number exactly without generating any processing error. TM in contrast accumulates an error at each step of the computation. Suppose, we use a finite number n of memory sells to represent a rational number, which is to be processed on the TM. At each step of the computation the number is read, modified according to the δ function and re-written on the tape. This generates error of order of the last sell of the representation. This error is linearly accumulated after each step of the computation.

Funding: This work was supported by Sheikh Saud bin Saqr Al Qasimi Foundation for Policy Research through 2020 Seed Grant.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nielsen, M.A.; Chuang, M.A. *Quantum Computation and Quantum Information*; Cambridge University Press: Cambridge, UK, 2000.
2. Yanofsky, N.S.; Mannucci, M.A. *Quantum Computing for Computer Scientists*; Cambridge University Press: Cambridge, UK, 2008.
3. Bernhardt, C. *Quantum Computing for Everyone*; The MIT Press: Cambridge, MA, USA, 2019.
4. Simon, D. On the Power of Quantum Computation. *SIAM J. Comput.* **1997**, *26*, 1474–1483. [CrossRef]
5. Sipser, M. *Introduction to the Theory of Computation*, 3rd ed.; Cengage Learning: Boston, MA, USA, 2012.
6. Deutsch, D. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A* **1985**, *400*, 97–117.
7. Yao, A. Quantum circuit complexity. In Proceedings of the 34th Annual IEEE Symposium on Foundations of Computer Science, Palo Alto, CA, USA, 3–5 November 1993; pp. 352–361.
8. Bernstein, E.; Vazirani, U. Quantum complexity theory. *SIAM J. Comput.* **1997**, *26*, 1411–1473. [CrossRef]
9. Molina, A.; Watrous, J. Revisiting the simulation of quantum Turing machines by quantum circuits. *Proc. R. Soc. A* **2019**, *475*, 20180767. [CrossRef] [PubMed]
10. Reiter, E.M.C. *Johnson, Limits of Computation: An Introduction to the Undecidable and the Intractable*; CRC Press: Boca Raton, FL, USA, 2013.
11. Siomau, M. Structural complexity of quantum networks. *AIP Conf. Proc.* **2016**, *1742*, 030017.
12. Siomau, M. Any Quantum Network is Structurally Controllable by a Single Driving Signal. *Quantum Inf. Process.* **2019**, *18*, 1. [CrossRef]
13. Siomau, M. Percolation Transition Control in Quantum Networks. *Quantum Inf. Process.* **2020**, *19*, 139. [CrossRef]