Holonomic Quantum Computing and Noncommutative Geometry

Zakaria Giunashvili
A. Razmadze Mathematics Institute Georgian Academy of Sciences, Department of Theoretical Physics,
Aleksidze st. 1, Tbilisi 380093, Georgia;
Georgian Technical University, Department of Applied Mathematics.

Abstract

We investigate the notion of computability from the point of view of Quantum Computation. A quantum algorithm for the holonomic Quantum Computation is considered from the point of view of connection in the differential fiber bundle over the parametrized space of controls and the graph (in the case of discrete algorithm). The noncommutative differnetial geometric approach to the quantum computation process is considered as a special case of the holonomic Quantum Computing, which allows to involve in the model of holonomic QC the classical computing (finite state machine) too.

1. Introduction

The application of quantum physical principles to the field of computing leads to the concept of the quantum computer, in which data is stored not as bits in conventional memory, but as the combined quantum state of many 2-state systems of qubits.

In this chapter we introduce the theoretical foundations, and basic concepts of a quantum computer and several models of quantum computation.

The basic idea of modern computing is the view of computation as a mechanical, rather than a purely mental process. A method, or procedure *P* for achieving some desired result is called effective in case when [1]:

1. *P* is set out in terms of a finite number of exact instructions each of which is expressed by means of a finite number of symbols;
2. *P* will, if proceeded without error, always produce the desired result in a finite number of steps;
3. *P* can be carried out by a human being without aid of any machinery save paper and pencil;
4. *P* requires no insight or ingenuity on the part of the human being carrying it out.

Alan Turing and Alonzo Church formalized the above definition by introducing the concept of *computability by Turing machine* and the mathematically equivalent concept of *recursive functions* with the following conclusions:

Turing's thesis: *Logical computing machines* (Turing machines) can do anything that could be described as purely mechanical [2].

Church's thesis: A function of positive integers is effectively calculable only if it is recursive [3].

The above statements are equivalent, and therefore, they are commonly referred to as the *Church-Turing Thesis* which defines the scope of classical computing science.

Despite its operationalistic approach, the above computability concept doesn't have much in common with the continuous nature of physics, so to construct a computing machine *M***,** we have to introduce a labeling function *m* which maps the analog physical states (e.g. the tension of a capacitor) to digital computational states (e.g. the value of a bit). The digital states have to be strings over some finite alphabet *A*.

Since the above definition of computability requires a finite number both, symbols and instructions, the labeling function only needs to apply on discrete intermediate machine states $S(t_0), S(t_1), \ldots$, so the temporal evolution of the machine state $S(t)$ is mapped onto a sequence of computational states $\{s_0, s_1, \ldots s_n\}$ where each transition $s_i \mapsto s_{i+1}$ corresponds to one function $I_i : A^* \to A^*$ from a enumerable set from a enumerable set *I* of simple instructions. The sequence of such functions $P = \{I_0, I_1, \ldots I_{n-1}\}$ is called *program*.

The states s_0 and s_n are called the *input* and the *output* states. The machine $M = (S,\ m,\ A,\ P)$ thus implements the function

$$f(s_0) = (I_0 \circ I_1 \circ \ldots \circ I_{n-1})(s_0) \text{ with } s_0 = m(S(0)) \in A^*.$$

The above definition of a computing machine poses strong restrictions on the interpretation of physical states. If we consider computation as a physical process, rather than a "mechanical" manipulation of symbols as defined above, we can drop all restrictions in the definition which don't have a physical equivalent.

As it is well-known, in quantum mechanics, the measurement of an observable *O* corresponding to a Hermitian operator *O* is only deterministic, if a system is in an eigenstate of the operator *O*. Following from the stochastic nature of quantum measurement, the labeling function *m* should be replaced by a probabilistic operator $M : H \to A^*$ which randomly chooses a string *s* according to some probability distribution $\delta_\varphi : s \to [0,1]$ with $\sum_s \delta_\varphi(s) = 1$.

Since it is not possible to non-destructively measure a quantum system and we are only interested in the result of a computation, it is not necessary that a labeling is defined for the intermediate steps of a computation, i.e. it is not required to "watch" the temporal evolution of the system, as long as a labeling for the input and output state is given.

While the transitions between the states S_i and S_{i+1} still have to correspond to some operators U_i from a enumerable instruction set of quantum transformations, the operators U_i don't have to directly correspond to functions on A^* (because of the reversibility of unitary operators, a direct correspondence would only be possible for bijective functions $f : A^* \to A^*$).

The temporal evolution of a quantum system is mathematically described by unitary operators, and therefore, a quantum program $P = \{U_0, U_1, \ldots, U_{n-1}\}$ is a composition of elementary unitary transformations.

Components of a Quantum Computer

Now, let us briefly describe the components of a Quantum Computer.

A classical, as well as a quantum computer, essentially consists of the following 3 parts: a memory, which holds the current machine state, processor, which performs elementary operations on the machine state, and some sort of input/output system, which allows to set the initial state and, somehow, extract the final state of the computation.

Quantum memory. The quantum analogue of the classical unit of information (bit) is the quantum bit or as it is reffered in the modern quantum computing literature, *qubit*. Just as a classical bit is represented by a system which can adopt one of two distinct states "0" and "1", we can define a quantum bit as follows: a qubit or quantum bit is a quantum system whose state can be fully described by a superposition of two orthogonal eigenstates labeled by |0> and |1>.

After this, the general state $|\psi\rangle \in H$ of a qubit can be given by the linear combination of the above basic vectors $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ with the condition $|\alpha|^2 + |\beta|^2 = 1$. The value of a quantum bit can be considered as a Hermitian operator $N|i\rangle = i|i\rangle$ over the Hilbert space $H = C^2$, or in matrix representation

$$N = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

The expectation value of the observable *N* is given by the following expression

$$\langle N\rangle = \langle\psi|N|\psi\rangle = (\alpha^* \beta^*)\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = |\beta|^2$$

Therefore, the value $\langle N\rangle$ gives the probability to find the system in state $|1\rangle$ if a measurement is performed on the qubit.

If we combine two qubits, we obtain a 4-dimensional Hilbert state with basis consisting of the vectors $|00\rangle$, $|10\rangle$, $|01\rangle$ and $|11\rangle$. Consequently, the general state of the resulting system is

$$\psi = \alpha|00\rangle + \beta|10\rangle\gamma|01\rangle + \delta|11\rangle \quad \text{with } |\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$$

While the state of a classical computer can be given as the distinct states of all bits in memory and processor registers, the “state of a qubit” is a meaningful term, if the machine state is the combined state of more than one system. So, we can state that: *the machine state of an n-qubit quantum computer is the current state of a combined system of n identical qubit subsystems.*

Generally, the machine state of an *n*-qubit quantum computer is given by

$$|\psi\rangle = \sum_{(d_0 \ldots d_{n-1}) \in B^n} c_{d_0 \ldots d_{n-1}} |d_0 \ldots d_{n-1}\rangle \quad \text{with} \quad \sum_{(d_0 \ldots d_{n-1}) \in B^n} |c_{d_0 \ldots d_{n-1}}|^2 = 1$$

Thus, the combined Hilbert space *H* is the tensor product of *n* 1-qubit Hilbert spaces:

$$H' = H \otimes H \otimes \cdots \otimes H = C^{2^n}$$

The eigenvectors $|d_0 \cdots d_{n-1}\rangle$ can be interpreted as binary numbers, with d_0 as least significant digit. So, the general machine state can be written as

$$|\psi\rangle = \sum_{i=0}^{2^n-1} c_i |i\rangle \text{ with } |i\rangle = |d_0 + 2d_1 + \cdots + 2^{n-1} d_{n-1}\rangle \equiv |d_0 \cdots d_{n-1}\rangle$$

The operator N_i, which corresponds to the *i*-th binary digit is given by the equality

$$N_i |d_0 \cdots d_{n-1}\rangle = d_i |d_0 \cdots d_{n-1}\rangle$$

and has the expectation value

$$\langle N_i\rangle = \sum_{(d_0 \cdots d_{n-1}) \in B^n} d_i \ |c_{d_0 \cdots d_{n-1}}|^2$$

Processing units. In a classical *n*-bit computer, every computational step can be described by a transition function $I : B^n \to B^n$, $B = \{0, 1\}$, which takes the current state *S* of all bits as input and returns the appropriate post instruction state S'.

As it is well-known from the quantum mechanics, the temporal evolution of a quantum system can be described by unitary operators on the corresponding Hilbert space. For the case of *n*-qubit computing machine, the general form of such unitary operator is

$$U = \sum_{i=0}^{2^n-1}\sum_{j=0}^{2^n-1} |i\rangle u_{ij} \langle j| \text{ with } \sum_{i=0}^{2^n-1} u_{ki}^{*} u_{kj} = \delta_{ij}$$

If we compare boolean functions to unitary operators from a strictly functional point of view, we can identify the following three major differences between classical and quantum operations:

- **Reversibility:** Since unitary operators are defined as the operators satisfying the condition $UU^{*} = I$, for each unitary transformation U, the corresponding inverse transformation is U^{*}. As a consequence, quantum computation is restricted to reversible functions. A classical analogue would be the class of reversible boolean functions.
- **Superposition:** An eigenstate $|\psi\rangle = |k\rangle$ can be transformed into a superposition of eigenstates:

  $$|\psi'\rangle = U|k\rangle = \sum_{k'} U_{k'k} |k\rangle$$

  The mathematical explanation of this feature lies in the fact that the requirement $\langle i|U^{*}U|j\rangle = \delta_{ij}$ is weaker than the pseudo classical condition

  $$\langle i|U^{*}|\pi_i\rangle\langle\pi_i|\pi_j\rangle\langle\pi_j|U|j\rangle = \delta_{ij}$$

  which requires transformation eigenstates not only to be orthonormal, but also be of the form $|U|k\rangle = |\pi_k\rangle$ with some appropriate permutation function π over the space Z_{2^n}.
- **Parallelism:** If the machine state $|\psi\rangle$ already is a superposition of several eigenstates, then a transformation *U* is applied to all eigenstates simultaneously:

  $$U\sum_i c_i |i\rangle = \sum_i c_i U|i\rangle$$

  This feature of quantum computing is called *quantum parallelism* and is a result of the linearity of unitary transformations.

The basic instructions of a classical computer usually operate only on a very small number of bits and are typically independent from the total amount of available memory. Therefore it is more useful to describe those instructions not as boolean functions over the whole state space B^n, in the case of an *n* bit machine, but as parametrized functions f_x over the space B^n, where the vector $x \in Z^n$ only holds the bit-positions of the relevant arguments.

The meaning of the phrase “swapping the bits 3 and 5” is clear on a classical computer, but this meaning can’t be directly adobt to quantum computing, because unitary operator operates on state and single qubit doesn’t have a state.

A quantum register can be defined as a sequence of mutually different qubit positions $s = \langle s_0, s_1, \cdots, s_{m-1}\rangle$, which is the quantum analogue of the above argument vector *v*. For such registers, can be defined a class of *(n-m)!* reordering operators Π_s by the equalities:

$$\Pi_s |d_0, d_1, \cdots, d_{n-1}\rangle = |d_{s_0}, d_{s_1}, \cdots, d_{s_{n-1}}\rangle$$

After which we can formulate the following

Definition. *The register operator U(s) for an m-qubit unitary operator* $U : C^{2^m} \to C^{2^m}$ *and a m-qubit quantum register s on an n-qubit quantum computer is the n-qubit operator*

$$U(s) = \Pi_s^*(U \times I(n-m))\Pi_s$$

with an arbitrary reordering operator Π_s.

Since there are $\frac{n!}{(n-m)!}$ possible *m*-qubit registers on an *n*-qubit machine, any given *m*-qubit operator *U* can describe $\frac{n!}{(n-m)!}$ different transformations *U(s)*.

In analogy to boolean networks, unitary operators which can be applied to arbitrary set of qubits are also referred to as *quantum gates*.

A well known result from classical Boolean logic, is that any possible Boolean function $f : B^n \to B^m$ can be constructed as a composition from a small universal set of operators by connecting the inputs and outputs to arbitrary bits in a feed-forward network. The most well known examples of such universal sets of logical gates are $\{\vee, \neg\}$, $\{\to, \neg\}$, or $\{\overline{\wedge}\}$.

Unitary operators on a Hilbert space can be described as abstract rotations of this Hilbert space. The general form of a rotation of a single qubit is

$$\omega, \alpha, \beta, \varphi) = \exp(-i\varphi)\begin{pmatrix} \exp(i\alpha)\cos(\omega) & -\exp(-i\varphi)\sin(\omega) \\ \exp(i\varphi)\sin(\omega) & \exp(-i\alpha)\cos(\omega) \end{pmatrix}$$

Applying these set of operators to arbitrary 2-dimensional subspaces of a Hilbert space *H*, any unitary transformation of this Hilbert space can be constructed by composition in at most $\begin{pmatrix} \dim(H) \\ 2 \end{pmatrix}$ steps. In our definition of universal quantum gates we are restricted to subspaces corresponding to quantum registers, therefore, in the case of an *n*-qubit quantum computer, we can work with only *n*-possible 1-qubit subspaces and the corresponding sets of register operators $U_{2^{(i)}}(\omega, \alpha, \beta, \varphi)$. It is clear that for any *i* and *j* the corresponding register operators commute with each other: $[U_{2^{(i)}}, U_{2^{(j)}}] = 0$, which implies that any composition *U* of the above register operators gives the transformation of the form

$$U \mid d_0, d_1, \cdots, d_{n-1}\rangle = (U^1 \mid d_0\rangle)(U^2 \mid d_2\rangle) \cdots (U^{n-1} \mid d_{n-1}\rangle)$$

So, just as the NOT gate itself is not universal in the classical Boolean logic, to construct a universal set of quantum gates, we require an additional 2-qubit operation, to create entangled multi-qubit states.

One possibility for a nontrivial 2-quit operator is XOR which is defined as $XOR : |x, y\rangle \mapsto |x, x \otimes y\rangle$. The matrix representation of this operator is

$$XOR = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

As it is shown in [4], the set of operators $\{U_2(\omega,\alpha,\beta,\varphi),\ XOR\}$, is universal for the family of unitary tramsformations. Furthermore, as the subset $\{U_2(\omega',\alpha',\beta',\varphi')^n\}$ is dense in $\{U_2(\omega,\alpha,\beta,\varphi)\}$, for such set of parameters that the quotients between them are irrational, the subset $\{U_2,\ XOR\}$ is universal for most operators U_2, in the sence that any unitary transformation U can be approximated to arbitrary precision.

Input and output. As it was mentioned, the interpretation of computing as a physical process, rather than the abstract manipulation of symbols, leads to an extended notion of computability, and the concept of unitary transformations is an adequate paradigm for the computability in the physical sence.

Unitary transformations describe the transition between machine states and therefore the temporal evolution of a quantum system. The notion of a quantum computer as a computing machine requires. however, that the evolution of the physical system corresponds to a processing of information. Unlike classical symbolic computation, where every single step of a computation can be mapped onto a bit-string, physical computation requires such a labeling only for the initial and the final machine state. This requirement is in full accordance with that interpretation of quantum physics, which states that the setup and the outcome of any experiment has to be described in classical terms.

As the machine state ψ is not directly accessible, any physically realizable labeling has to correspond to some observable (Hermitian operator).

One natural choice for such observable for an *n*-qubit quantum computer would be the classical values of the single qubits with the Hermitian operators

$$N = (N_0, N_1, \ \cdots\ , N_{n-1}) = N_0 + 2N_1 + \ \cdots\ + 2^{n-1}N_{n-1}$$

$$N_i \mid d_0,\ d_1,\ \cdots, d_{n-1}\rangle = d_i \mid d_0,\ d_1,\ \cdots, d_{n-1}\rangle$$

To set a quantum computer to a desired initial state corresponding to some boolean input string, it suffices to provide means to initially set all qubits to |0> and then apply any unitary transformation that carries this state to the desired one. Consequently, one of the important operators is the reset operator, which is a constant operator and is defined as $R \mid \psi\rangle = \mid 0\rangle$.

Various Models of Quantum Computation

In classical information theory, the concept of the universal computer can be represented by several equivalent models, corresponding to different scientific approaches. From a mathematical point of view, a universal computer is a machine capable of calculating partially recursive functions, computer scientists often use the Turing machine as one of the models, in electro-engineering the most popular model is logic circuits and in programming, the preferred model is a universal programming language.

Each of these models has its quantum analogue:

Partially Recursive Functions	$\leftrightarrow$	**Unitary Operators**
Turing Machine	$\leftrightarrow$	**Quantum Turing Machine**
Logical Circuit	$\leftrightarrow$	**Quantum Gates**
Universal Programming Language	$\leftrightarrow$	**Quantum Programming Language**

The notion of computation as a physical process requires that quantum computation can be described by the same means as any other physical reality, which, for quantum physics is the mathematical formalism of Hilbert space operator algebra.

The equivalent of the recursive functions (which is the mathematical concept of computability) in quantum computing are unitary operators on some complex Hilbert space. As every classical computation problem can be formulated in terms of the partially recursive functions, any quantum computation problem must have corresponding unitary operator. And the computational algorithm consists of the algorithmic decomposition of this unitary operator into elementary operations, which itself are some unitary operators on smaller complex spaces.

In analogy to the classical Turing Machine several propositions of Quantum Turing Machines, as a model of a universal quantum computer have been made [5, 6]. In these models, the complete machine-state is given by a superposition of base states $|l, j, s\rangle$, where l is the inner state of the head, j the head position and s the binary representation of the tape content. To keep the hilbert space H separable, the infinite bit-string s should have the zero tail state condition, that is, only a finite number of bits different from 0 are allowed.

The quantum analogue of the transition function of a classical probabilistic Turing Machine is the step operator T , which has to be unitary to allow for the existence of a corresponding Hamiltonian and meet locality conditions for the effected tape-qubit, as well as for head movement.

Quantum circuits are the quantum analogues to the classical boolean networks, with some major differences:

- Since all quantum computations have to be unitary, all quantum computations have to be unitary, all quantum circuits can be evaluated in both directions.
- Only n to n networks are allowed, that is, the total number of inputs has to be equal to the total number of outputs.
- No forking of inputs is allowed. This follows from the fact that qubits cannot be copied, i.e. there exists no such unitary operator: $U|\psi\rangle|0\rangle = |\psi\rangle|\psi\rangle$ for $|\psi\rangle \in C^2$.

To allow for implementation of all possible unitary transformations, a universal set of elementary gates must be available, out of which composed gates can be constructed. Therefore, each m-qubit gate describes up to $\frac{n!}{(n-m)!}$ different unitary transformations, depending on the wiring of the inputs.

When we come to programming and the design of no-classical algorithms, we can look at the mathematical description as the specification and quantum circuits as the assembly language of the Quantum Computing.

As classical programming languages, quantum programming languages provide a constructive means to specify the sequence of elementary operators, while allowing nested levels of abstraction. In its simplest form, a quantum algorithm consists of a unitary transformation and a subsequent measurement of the resulting state. For more traditional computational tasks, as e.g. searching or mathematical calculations, efficient quantum implementations often have the form of probabilistic algorithms. More complex quantum algorithms, as e.g. Shor's algorithm for quantum factoring can include classical random numbers, partial measurements, nested evaluation loops etc.

A formal way to describe the classical control structure, is to consider quantum operations as special statements within a classical procedural language. Therefore any quantum programming language has to be universal programming language.

The Quantum Adiabatic Algorithm for the Hilbert's Tenth Problem

The halting problem for Turing machines is a manifestation of undecidability: a Turing computation is equivalent to the evaluation of a partial recursive function, which is only defined for a subset of the integers. As this subset is classically undecidable we cannot determine in advance whether the Turing machine will halt or not.

The proof f the unsolvabilty of the halting problem is by contradiction with the assumption of the existence of a computable halting function *h(p, i)* which has two integer arguments: *p* is the encoded integer number for the algorithm and *i* is its encoded integer input. Formally, this can be written as:

$$h(p, i) = \begin{cases} 0 \ \ if \ \ p \ halts \ on \ input \ i \\ 1 \ if \ \ p \ does \ not \end{cases}$$

One can then construct a program $r(n)$ having one integer argument n in such a way that it calls the function *h(n, n)* as a subroutine and

$$\begin{cases} r(n) \ halts \ if \ \ h(n, n) = 1 \\ r(n) \ loops \ infinitely \ otherwise \end{cases}$$

The application of the halting function *h* on the program *r* and input *n* results in

$$h(r, n) = \begin{cases} 0 \ if \ \ h(n, n) = 1 \\ 1 \ if \ \ h(n, n) = 0 \end{cases}$$

A contradiction is clearly manifest once we put $n = r$ in the last equation above. However, this contradiction argument might be avoided if we distinguish and separate the two classes of quantum and classical algorithms. A quantum function similar to the *qh(p, i)* above *h* can exist to determine whether any classical program *p* will halt on any classical input *i* or not. The above contradiction can be avoided if the quantum halting cannot take as an argument the modified program, which is now a quantum program.

To investigate the decidability of the Turing halting problem in the framework of quantum computability, consider the Diophantine equations and Hilbert's tenth problem.

At the turn of the last century, David Hilbert listed 23 important problems among which the problem number ten is the only decision problem and could be rephrased as:

Given any polynomial equation with any number of unknowns and with integer coefficients, device a universal process, according to which it can be determined by a finite number of operations whether the equation has integer solution.

Eventually, this problem was finally shown to be undecidable in 1970 by Matiyasevich [7] and the result was formulated as follows:

the Hilbert's tenth problem could be solved if and only if the Turing halting problem could also be solved.

As the Turing halting problem is not solvable (as it was shown above), the Hilbert's tenth problem is undecidable.

Among the alternative models of quantum computation the quantum adiabatic process is the recent proposal to employ for computation. The idea is to encode the solution of some problem to be solved into the ground state $|g\rangle$, of some suitable hamiltonian, H_1. But as it is easier to implement the hamiltonian than to obtain the ground state, we should start the computation in a different and

readily available initial ground state, $|g_0\rangle$, of some initial hamiltonian H_0, then deform this hamiltonian in a time *T* into the hamiltonian whose ground state is the desired one, through a time-dependent process

$$H\left(\frac{t}{T}\right)=\left(1-\frac{t}{T}\right)H_0+\frac{t}{T}H_1$$

The adiabatic theorem of quantum mechanics states that if the deformation time is sufficiently slow compared to some intrinsic time scale, the initial ground state will evolve into the desired ground state with high probability – the longer the time, the higher the probability.

The quantum algorithm with the ground-state oracle is thus clear:

one starts, for example with a hamiltonian

$$H_0=\sum_{i=1}^{k}(a_i^*-\alpha_i^*)(a_i-\alpha_i)$$

where a_i^* and a_i are the creation and anihilation operators for the corresponding Fock space. This hamiltonian admits the readily achievable coherent state $|g_0\rangle=|\alpha_1\cdots\alpha_k\rangle$ as its ground state. Then one forms the slowly varying hamiltonian *H(t/T)* which interpolates in the time interval [0, *T*] between the initial H_0 and H_1. According to the quantum adiabatic theorem, the initial ground state with certain probability will evolve into our desired ground state up to a phase.

In contrast with the classical algorithm, the quantum algorithm above will terminate in principle, because the time interval *T* is always finite (even though sometimes it can be very long) and give us the decision result for the Hilbert's tenth problem.

One of the generalizations of the above described adiabatic approach to the quantum computation is the *holonomic quantum computation* [9]. In this approach, the information is encoded in a degenerate eigenspace of a parametric family of hamiltonians and manipulated by the associated holonomic gates. These are realized in terms of non-abelian berry connection and are obtained by driving the control parameters along adiabatic loops. For a specific model (which in fact is universal) it is possible to explicitly determine the loops generating any desired logical gate, thus producing the universal set of unitary transformations.

The noncommutative approach to this construction allows us to involve the classical (discrete) computing in the general model of holonomic quantum computing. In this model the classical finite state automata can be considered as a discrete differential calculus on some oriented graph [8] and the transition function can be considered as a connection on some fiber bundle over this graph. Therefore, the calculation procedure (algorithm) consists of a motion along a finite number of nodes of the base graph and the holonomy corresponding to the connection on the fiber bundle gives us the desired unitary transformation (or recursive function) as a composition of the unitary transformations corresponding to the nodes of the graph.

Bibliography

1. B. Jack Copeland. The Church-Turing Thesis. // Stanford Encyclopedia of Philosophy 1095-5054 (1996).
2. A. M. Turing. Intelligent Machinery. // National Physical Laboratory Report (1948).
3. E. L. Post. Finite Combinatory Processes – Formulation 1. // Journal of Symbolic Logic 1, 103-105 (1936).
4. D. Deutsch. Quantum Computational Networks. // Proceedings of the Royal Society. London A 439, 553-558 (1989).
5. Paul Benioff. Models of Quantum Turing Machines. // LANL Archive quant-ph/9708054 (1995).
6. D. Deutsch. Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer. // Proceedings of the Royal Society. London, A 400, 97-117.
7. Y. V. Matiyasevich. Hilbert's Tenth Problem. // MIT Press (1993).
8. A. Dimakis, F. Muller-Hoissen. Discrete Differential Calculus, Graphs, Topologies and Gauge Theory. // LANL Archive hep-th/9404112.
9. P. Zanardi, M. Rasetti. Holonomic Quantum Computation. // LANL Archive quant-ph/9904011.