Multiple-Valued Quantum Circuits and Research Challenges for Logic Design and Computational Intelligence Communities

Marek A. Perkowski
Portland Quantum Logic Group
Department of Electrical and Computer Engineering, Portland State University, USA.

Abstract. **The paper is an overview of basic quantum circuits for researchers with background in logic design and computational intelligence. After intuitive review of quantum circuits, we discuss the issues in simulation, synthesis and test for them, pointing to the similarities with classical computing. Quantum Computational Intelligence is presented as a new area that uses quantum ideas in the framework of Computational Intelligence, especially machine learning. The goal of the paper is to demonstrate that with a very limited introduction to quantum fundamentals, the specialists from other fields can use their knowledge and skills to become productive researchers in this new quantum-related area of computer engineering.**

1. Introduction

Quantum computing (QC) is a very promising and flourishing research area [13,14,18,20,30,35,41]. QC and quantum communication give high promises of dramatically faster, smaller and more powerful systems for computing, communication and widely understood intelligent robotics, including atomic-scale robotics. Unfortunately, to understand the research published in the physics journals requires familiarity with concepts and notations of quantum mechanics, the domain that computer scientists and engineers are rarely familiar with. Hopefully, using the metaphor of a "network", the fundamental concepts of quantum computing are quite easy to understand for someone who is familiar with classical digital computing. After being able to understand how the quantum "circuit" operates, the researchers are immediately ready to use their skills and intuitions to design quantum circuits and algorithms and develop new software tools for simulation, synthesis, testing and analysis of them. Because the quantum concepts generalize standard computer notions such as combinational circuits, state machines, games or spectral transforms, every traditional computing concept can be generalized to its counterpart quantum concept, the usefulness of which should be next analyzed with respect to computational complexity and new attractive computational features. Thus a specialist in DSP may design quantum spectral transforms superior to classical transforms, neural network researcher would create their quantum counterparts, test engineer would analyze how quantum networks can be tested, CAD developer would create efficient software for quantum CAD, fuzzy logic specialist would generalize concepts to quantum fuzzy logic, and others would work on development of quantum algorithms or models of learning based on quantum phenomena in these networks. Quantum is the whole new world to investigate, but we have already many guidelines from the history of computing to follow!

Theoretically, QC allows designers to build much more efficient computers than the existing classical ones. For example, some problems that can't be solved in polynomial time using classical computers can be solved in polynomial time using quantum computers [41]. In part, this is because quantum circuits are inherently able to perform massive parallel computations [41]. In this paper we will not deal with building physical quantum computers but only with the minimal and useful from the engineering standpoint mathematical description of their operation. Although it is too early to use quantum circuits and algorithms for practical applications (now quantum computers with not more than 10 qubits can be build), their correctness can be verified using quantum simulators (some circuits up to 45 qubits have been simulated). Formally, quantum algorithms are also (combinational - no loops or memory) circuits, but with hierarchical structure composed of many circuit block levels. While most of the results in quantum literature are for binary quantum computing, the multi-valued (MV) and hybrid quantum computing are new and exciting research areas in which not many results are known. Observe first, that in classical digital circuit design binary rather than multi-valued logic is a natural choice for a variable (bit) because of the physics of transistor's operation. In case of a quantum realization, however, various logics with higher than two radix can be realized. Moreover, quantum phenomena allow naturally the realization of the so-called hybrid quantum circuits in which a two-input gate may have for instance one binary input and one ternary input. Here we give background for such logic, which allows also to generalize and unify the presentation without making it more complex. Below, few modern research areas fundamental to the success of future quantum computing are briefly presented. Some paragraphs include sections with suggested immediate research areas for people with computer engineering or Computational Intelligence backgrounds. Numerous provided references should aid the reader as a starting point in individual study.

2. Fundamentals of Multiple-Valued Quantum Circuits

In multi-valued (MV) Quantum Computing, the unit of memory (information) is qudit. MV quantum logic operations manipulate qudits, which are microscopic entities such as a photon's polarization or atomic spin. For instance, ternary logic values of 0, 1, and 2 are represented by a set of distinguishable different basis states of a qutrit. These states can be a photon's polarizations or an elementary particle's spins. After encoding these distinguishable quantities into multiple-valued values, qutrit states are represented by basis states $|0\rangle$, $|1\rangle$ and $|2\rangle$, respectively. A qubit, used for Quantum Computing uses only two basis states, $|0\rangle$ and $|1\rangle$, binary quantum circuits are a special case from a formal point of view nothing more than k-valued quantum circuits for a special case of k = 2. Qubit and qutrit are then special cases of qudits. Qudits exist in a linear superposition of states, and are characterized by a wave function ψ. As an example (d=2), it is possible to have light polarizations other than purely horizontal or vertical, such as slant 45^o corresponding to the linear superposition

in Hilbert space, $\psi = \frac{1}{2}\left[\sqrt{2}|0\rangle + \sqrt{2}|1\rangle\right]$. In ternary logic, the notation for the superposition is $\alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$ where α, β, and γ are complex numbers. These intermediate states cannot be distinguished, rather a measurement will yield that the qutrit is in one of the basis states $|0\rangle$, $|1\rangle$, or $|2\rangle$. The probability that a measurement of a qutrit yields state $|0\rangle$ is $|\alpha|^2$, state $|1\rangle$ is $|\beta|^2$, and state $|2\rangle$ is $|\gamma|^2$. The sum of these probabilities is one. The absolute values are required since, in general, α, β and γ are complex quantities. Recall that $|a|^2 = a \cdot a^*$ where $\cdot$ is the operation of multiplication of complex numbers and a* denotes the complex conjugate of number a. By + we denote addition of complex numbers. Pairs of qutrits are capable of representing nine distinct states $|00\rangle$,$|01\rangle$,$|02\rangle$,$|10\rangle$,$|11\rangle$, $|12\rangle$,$|20\rangle$,$|21\rangle$, and $|22\rangle$, as well as all possible superpositions of these states. This property is mathematically described using the Kronecker product (tensor product) operation $\otimes$. The Kronecker product of two matrices is defined as follows:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \otimes \begin{bmatrix} x & y \\ z & v \end{bmatrix} = \begin{bmatrix} a\begin{bmatrix} x & y \\ z & v \end{bmatrix} & b\begin{bmatrix} x & y \\ z & v \end{bmatrix} \\ c\begin{bmatrix} x & y \\ z & v \end{bmatrix} & d\begin{bmatrix} x & y \\ z & v \end{bmatrix} \end{bmatrix} = \begin{bmatrix} ax & ay & bx & by \\ az & av & bz & bv \\ cx & cy & dx & dy \\ cz & cv & dz & dv \end{bmatrix}$$

Observe that expression such as ax is a multiplication of complex numbers (symbols of multiplication are avoided). The 2*2 matrix corresponds to single-qubit binary gate, thus the 4*4 matrix above corresponds to a quantum circuit in which two single-qubit gates operate in parallel. As always in reversible logic, the numbers of inputs and outputs of a quantum gate are the same. Similarly to the tensor product of 2*2 matrices above, one can define tensor products for any sizes of matrices and in particular for vectors representing superposed states. As an example, consider two qutrits with $\psi_1 = \alpha_1|0\rangle + \beta_1|1\rangle + \gamma_1|2\rangle$ and $\psi_2 = \alpha_1|0\rangle + \beta_1|1\rangle + \gamma_1|2\rangle$. When the two qutrits are considered together to represent a state, that state ψ_{12} is the superposition of all possible combinations of the original qutrits, where

$$\psi_{12} = \psi_1 \otimes \psi_2 = \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \alpha_1\gamma_2|02\rangle + \beta_1\alpha_2|10\rangle + \beta_1\beta_2|11\rangle + \beta_1\gamma_2|12\rangle + \gamma_1\alpha_2|20\rangle + \gamma_1\beta_2|21\rangle + \gamma_1\gamma_2|22\rangle$$

The notation above is called Dirac notation. The Heisenberg notation uses matrices for operators that act on states and uses vectors to represent states. Both notations are useful and we will illustrate both of them. The superposition property allows the qubit states to grow much faster in dimension than classical bits, and the qudits faster than qubits [39]. In a classical system, n bits represent 2^n distinct states, whereas n qubits correspond to a superposition of 2^n states and n qutrits correspond to a superposition of 3^n states. Because we are multiplying in general complex numbers, in the above formula some coefficient can be equal to zero, so there exists a constraint bounding the possible states in which the system can exist. For instance, the state $\gamma|00\rangle + \varepsilon|11\rangle$ is the so-called entangled state that when measured will produce $|00\rangle$ with probability $|\gamma|^2$ and $|11\rangle$ with probability $|\varepsilon|^2$. Observe that states $|01\rangle$ and $|10\rangle$ are never measured in this example! Observe also that the entangled state $\gamma|00\rangle + \varepsilon|11\rangle$ cannot be factored back to a Kronecker product. This kind of calculation is not possible by a physical system in macro-world. It exists only in quantum world and is the base of efficiency of quantum algorithms and testing of quantum circuits [4]. As observed in [40] - "Allowing d to be arbitrary enables a tradeoff between the number of qudits making up the quantum computer and the number of levels in each qudit". Because in contemporary quantum technologies every qubit is costly, higher radices than 2 give an advantage of improved processing and storage power at the same realization cost. This is just one of strong arguments for the realization of multi-valued logic in quantum circuits. In addition to standard advantages of MV logic, quantum MV logic may be superior to binary because of different nature of entanglement [40]. The study of entanglement is very important to understand the essence of quantum phenomena and algorithms [41]; new quantum algorithm use entanglement in a new creative way; entanglement is counterintuitive and requires computationally difficult calculations. Thus many methods are used and potentially methods of Computational Intelligence such as neural nets or evolutionary programming should be used as well. Future automatic synthesis of quantum algorithms will be based on good understanding of entanglement.

Unitary matrix U is one that $U \cdot U^+ = I$, where U^+ is an adjoint matrix of U and I is the identity matrix. An adjoint U^+ is a conjugate transpose matrix of U. A gate or a sub-circuit of a quantum circuit is described as a unitary matrix, from now we will not distinguish between them. In Heisenberg notation, an output of a gate is obtained by multiplying the unitary matrix of this gate by a vector of Hilbert space corresponding to this gate's input state. Quantum gates can be connected in series or in parallel. If they are connected in parallel, the resultant matrix is obtained by Kronecker multiplication of their matrices. If they are connected in series, one uses standard matrix multiplication of gates' matrices. Various quantum notations, such as the Dirac notation above or matrix (Heisenberg) notation shown below, contribute to the difficulty in understanding the concepts of quantum computing. These difficulties limited until recently successes of non-physicists in creating efficient analysis, simulation, verification and synthesis algorithms for QC. Generally, however, it is our experience that once the minimal amount of formalism is understood, researchers with engineering background and familiar with any types of networks can quickly contribute to new circuits, algorithms and software, since much can be learnt and re-used from the Electronic Design Automation, DSP, evolutionary computing and MV logic theory and design. The achievements and methodologies of these well-developed areas should be now applied to develop new concepts and design efficient software tools for quantum computing. Also, to introduce quantum-inspired ideas to these former disciplines [24]. The benefits will be thus mutual. Here we include the absolute minimum amount of formalism sufficient to start independent software development by people who have sufficient background in any classical area such as particularly; EDA tools, algorithms, search, machine learning, evolutionary programming or fuzzy-neural systems.

In terms of logic operations, anything that changes a vector of qudit states to another qudit vector and satisfies the measurement probability properties can be considered as a quantum operator (unitary matrix). These phenomena can be modeled using the analogy of a "quantum circuit" (called also quantum array or quantum network). In a quantum circuit, a wire does not carry ternary values but corresponds to a 3-tuple of complex values, α, β, and γ. Quantum logic gates of the circuit map the complex values on their inputs to complex values on their outputs. When the operation of a quantum gate is described by a unitary matrix, any quantum circuit, being a composition of parallel and serial connections of blocks, from small to large, can be described by operations on these matrices. Small blocks correspond to quantum gates that are easily directly realizable (like Pauli rotations) or are very simple and require just few basic quantum operations such as the Feynman gates [19,41] in binary, or the Stroud/Muthukrishnan gates [40] in multi-

ple-valued quantum logic. Theoretically, as shown above, the analysis, simulation and verification are easy and can be based on matrix algebra and software. Practically they are tough because the dimensions of the matrices grow exponentially. All these become much easier when one deals only with permutative matrices, which are equivalent to multi-output truth tables of completely specified functions. In such matrices there is exactly one "1" in every row and in every column. An active research area is to represent operations on unitary matrices (in particular, the permutation matrices) by new efficient data structures, algorithms and hardware.

An important unitary matrix property is that of a full rank. This property implies that quantum gate matrix rows and columns are orthonormal. Therefore, past results from spectral methods for classic digital logic are directly applicable to quantum logic synthesis. Unitary quantum transforms can be build and applied in all areas where spectral transforms are now used, such as in Image Processing. Furthermore, since quantum logic gates are represented using unitary orthonormal matrices, they represent logically reversible gates. These observations mean that the single input/output quantum logic gates are rotation matrices characterized by some particular rotation angle θ, where, for example, a = cosθ, b = sinθ, c = -sinθ and d = cosθ. With this viewpoint, it can be seen that there exist, in fact, an infinite number of single input/output qubit gates. However, three elementary gates can be used to generate any rotation [41]. These are the R, S, and T gates described in matrix notation by

$$R=\begin{bmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{bmatrix}\quad S=\begin{bmatrix}1 & 0\\ 0 & e^{i\pi}\end{bmatrix}\quad T=\begin{bmatrix}e^{i\pi} & 0\\ 0 & e^{-i\pi}\end{bmatrix} \quad (1)$$

One of the most important quantum gates is the quantum XOR gate (called also Feynman gate or Controlled-Not gate - CNOT). This gate allows input states of $|00\rangle$ and $|01\rangle$ to appear unchanged at the outputs, but interchanges the states $|10\rangle$ and $|11\rangle$. This is seen in the 4*4 permutative matrix in equation (2) below. Columns (from left to right) correspond to input states of the gate, and rows (from top to bottom) correspond to output states -- $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. Thus the transition from basis input state $|11\rangle$ to basis output state $|10\rangle$ is represented as a "1" at the intersection of the fourth column and third row. The basis state vector of state $|10\rangle = [0\ 0\ 1\ 0]^T$, is a Kronecker product of states $|1\rangle = [0\ 1]^T$ and $|0\rangle = [1\ 0]^T$. (T stands for vector transpose). When only basis states are used we remain in the realm of classical reversible logic. When vectors represent arbitrary superpositions, we are in quantum domain. Let us analyze operation of this gate on basis input state state $|10\rangle$. By multiplying the matrix of CNOT by the input state $|10\rangle = [0\ 0\ 1\ 0]^T$ we obtain vector $[0\ 0\ 0\ 1]^T$, as follows:

$$\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}0\\0\\1\\0\end{bmatrix}=\begin{bmatrix}0\\0\\0\\1\end{bmatrix} \quad (2)$$

In this example, the input is $|10\rangle = (0)|0\rangle+(1)|1\rangle) \otimes ((1)|0\rangle+(0)|1\rangle)$, and the input vector is represented by the coefficients shown in parentheses. Logically the CNOT gate can be described by two equations: P = a, Q = a⊗b, where bit a is the control bit and bit b is the controlled or data bit. Input variables are a and b, output variables are P and Q. Dirac notation would be $|a, a \otimes b\rangle$. Remember however that this gate operates on arbitrary complex states not only on basis states. By multiplying the matrix of CNOT by the input state $[\alpha\ \beta\ \gamma\ \delta]^T$ we obtain vector $[\alpha\ \beta\ \delta\ \gamma]^T$, illustrating operation of a permutative gate on arbitrary states.

It is a significant fact that the unitary gates described by Equations (1) and (2) can realize any quantum logic function [7] (including standard binary). Every non-reversible Boolean or multiple-valued circuit C_1 (with arbitrary numbers of inputs and outputs) can be transformed to a reversible circuit C_2 (described by a permutative matrix). This requires adding bits to C_2. Circuit C_2 is equivalent to C_1 in the sense that on its original bits it produces the same input-output mapping as C_1. Bits added to circuit C_2 are the so-called ancilla bits, initialized to constants, they are always necessary to convert a non-reversible logic function to a reversible function on more variables, but we want to add as few of them as possible. C_2 has also non-used output bits, called garbage. Conversion methods are looked for to minimize garbage and ancilla bits for arbitrary, especially incompletely specified functions. There are several strong similarities of quantum logic, and especially its subset of reversible circuits, to classic digital circuit design using XOR gates, called sometimes XOR logic or Reed-Muller logic [4,31,32]. Researchers with background in this logic as well as spectral approaches to logic synthesis are at the advantage when they work on quantum circuits because many ideas can be adapted.

Observe in unitary matrix for CNOT gate (Eqn. 2) that when the control bit a=P (the first one from top) has value $|0\rangle$ the gate does nothing, the controlled bit b (the lower bit) has the same value on input and on output. When the first bit a has however value $|1\rangle$, the controlled bit b is inverted (since 1⊗b = b'). Therefore this gate is called Controlled-NOT, which means that the controlling bit controls the NOT operation. Similarly a 3-qubit (universal) Fredkin gate is called a Controlled-Swap, because when the first bit P = a (control) is $|0\rangle$ the gate does nothing, which means that the lower two bits just transfer input values; Q = b, R = c, and when the control bit is $|1\rangle$, these bits are swapped; Q = c, R = b. Thus the Fredkin gate realizes on basis states the following reversible function, P = a, Q = (a' ∧ b) v (a ∧ c), R = (a' ∧ c) v (a ∧ b). A binary Toffoli gate (Figure 1) is described by equations: P = a, Q = b, R = (a∧b)⊗c. Symbol ∧ denotes Boolean AND, symbol v Boolean OR and symbol ⊗ Boolean XOR. Toffoli is also called Controlled-Controlled-NOT, since NOT is controlled by a product of inputs a and b. Feynman, Fredkin and Toffoli gates, named after their early inventors, are examples of controlled gates. Controlled gates are created also for multi-valued and hybrid quantum circuits. In hybrid logic a ternary bit A may control a binary bit B or a binary bit may control a quaternary bit, but every wire is for one radix logic only, binary, ternary or quaternary, throughout the entire circuit. Such multi-valued and hybrid controlled gates are fundamental to quantum circuits and continue to be areas of active study.

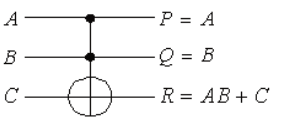

Figure 1. A standard binary 3*3 Toffoli gate. Symbol + is a Boolean XOR. It would be a modulo-3 addition in ternary Toffoli gate. The circuit should be read from left to right (representing time) but is reversible.

We will illustrate now one way to realize multi-valued logic using binary quantum computing. The normalization $|\alpha|^2 + |\beta|^2 = 1$ admits the parameterization $\alpha = \cos(\theta/2)\, e^{i\gamma}$, $\beta = \sin(\theta/2)\, e^{i\delta}$. $|\Psi\rangle = e^{i\alpha}(\cos(\theta/2)|0\rangle + e^{i\varphi}\sin(\theta/2)|1\rangle)$. Since the global phase of $|\Psi\rangle$ has no observable effect, we may write $|\Psi\rangle = \cos(\theta/2)|0\rangle + e^{i\varphi}\sin(\theta/2)|1\rangle$. The angles θ and φ define a point on the surface of a unit sphere - the Bloch sphere, see Figure 2. The Bloch sphere provides an excellent tool to visualize the state vector of a qubit.

The identity matrix and three Pauli matrices: $I=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} X=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} Y=\begin{bmatrix}0 & -i\\ i & 0\end{bmatrix} Z=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}$ form a basis for the 2*2 density matrices. We associate with every 1-qubit state $p = \frac{1}{2}(I + a_x X + a_y Y + a_z Z)$ vector (a_x, a_y, a_z). If $p = |\Psi\rangle\langle\Psi|$ for a state $|\Psi\rangle = e^{i\alpha}(\cos(\theta/2)|0\rangle + e^{i\varphi}\sin(\theta/2)|1\rangle)$. Then the corre-

sponding vector is (a_x, a_y, a_z) = (sinθ cosψ, sinθ sinψ, cosθ). It can be easily derived that the vectors (a_x, a_y, a_z) satisfy $|a_x|^2+|a_y|^2+|a_z|^2 = 1$, which means that all pure states are located on the surface of the Bloch Sphere [41]. When there are many identical quantum circuits working together they are described by density matrices and the (mixed) states may lay inside the sphere, not on the surface.

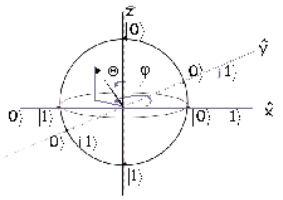

Figure 2. Bloch Sphere with 6 values shown.

Figure 2 shows the location of 6 points (quantum states), that may correspond to logic values used in some multi-valued quantum algebras. For binary logic we use $|0\rangle$ and $|1\rangle$. For quaternary logic we use $|0\rangle, |1\rangle, |0\rangle+|1\rangle$, and $|0\rangle-|1\rangle$. For 6-valued logic we use additionally points $|0\rangle+ j|1\rangle$ and $|0\rangle - j|1\rangle$. This assumes only 90 degree angle rotations. A rotation or a combination of rotations leads from one value to any other value. These are single qubit operations, which we want to reduce to the minimum number of elementary "gates" being physically realizable "rotations", such as Pauli rotations. Observe that the Bloch Sphere visualizes all possible values of a single qubit in Hilbert space and all operators on a single qubit. It has a big didactic value to explain operation of quantum circuits. Because global phase does not count as it cannot be distinguished in measurement, the T gate can be also written as follows:

$$T(\Pi/8) = \begin{bmatrix} 1 & 0 \\ 0 & e^{j\pi/4} \end{bmatrix}$$

H denotes the important Hadamard gate:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

The Hadamard and the Π/8 gate can be used to approximate any given single-qubit unitary operation with arbitrary accuracy. On the Bloch sphere, T and HTH are rotations by an angle Π/4 radians around the z- and x-axes, respectively. The composition of these two operations gives a rotation by an angle Θ, which is defined by $\cos\Theta/2 = \cos^2\Theta/8$, around an axis n, which is defined by n = (cosΠ/8, sinΠ/8, cosΠ/8). Since Θ is irrational, any rotation around the Θ-axis can be build, to arbitrary precision, from T and HTH. Furthermore, since for α arbitrary H $R_n(\alpha)$ H = $R_m(\alpha)$ with m = (cosΠ/8, - sinΠ/8, cosΠ/8) not collinear n, there are angles α, β, γ such that any given U can be written U = $R_n(\alpha)$ $R_m(\beta)$ $R_n(\gamma)$. It can be also shown that any given 2-qubit gate can be composed from CNOT and a single qubit gate. Similarly, other universal sets of 1-qubit gates can be found and illustrated using Bloch Sphere. This Sphere is also useful to find operator identities (quantum generalizations of Boolean logic rules like "Not (Not B) = B"). The quantum circuit can be simplified and transformed using these identities. Finding useful sets of identities for binary, multi-valued and hybrid quantum logic and creating search software to optimize circuits based on applying these identities remain outstanding open problems in quantum circuit design.

The reason that multi-valued logic is not yet a mainstream in quantum computing has perhaps mainly historical reasons, since realization of MV circuits requires technologically exactly the same basic quantum operations as in case of the binary quantum logic; it means, single qubit 90° Pauli rotations and two-qubit controlled gates in which binary qubits control arbitrary single-qubit 90° rotations. Observe that all these operations already exist in various quantum computer realization technologies such as Nuclear Magnetic Resonance (NMR) [41]. Above we showed how multiple-valued logic can be encoded in binary quantum computing. Quaternary logic requires two binary measurements. The first measurement distinguishes states $|0\rangle$ and $|1\rangle$, and the second measurement uses additional rotation gates to distinguish between states $|0\rangle+|1\rangle$, and $|0\rangle - |1\rangle$. It can be shown that the logic with 2^n values requires n measurements. Another approach to multi-valued quantum circuits requires measurement gates that measure more than two basis states. Such gates also start to appear. Study of universality and power as well as quantum realization costs of multi-valued gates are still active research areas. Investigations of various multi-valued and hybrid gates, their equivalence transformations, physical realizations, synthesis and testing methods just started in recent years. Very little is known on quantum algorithms using such gates, communication schemes that use them and spectral transforms in MV and hybrid logics. Applications in Computational Intelligence are also very new. Now that the reader is familiar with basic concepts and formalisms, we will list some new areas of research where substantial contributions can be done by building new software tools for quantum computing and inventing new quantum concepts that would draw from the existing engineering areas.

3. Quantum Circuit Simulation and the Role of Good Representation

Simulation of quantum circuits plays absolutely fundamental role in many areas of quantum physics and engineering. Similarly as in classical circuit design, simulation is used to verify correctness of the design or algorithm, to analyze its properties and to find some interesting aspects that cannot be found by "hand and pencil" methods. It is amazing that the first quantum algorithms were invented without quantum simulators, but now the researchers routinely use quantum simulators to help them with the designed by them algorithms (circuits) and to verify their design guesses. Quantum simulators are used to simulate a good quantum circuit and a circuit with inserted quantum faults for test generation and fault localization in quantum circuits [4,42,44]. Observe that many search-based synthesis and optimization methods used in Computational Intelligence require simulation as part of the fitness function calculation. This is also true in quantum applications, especially for synthesis. When the exhaustive search, genetic algorithms, genetic programming, bacteria foraging, particle swarm optimization, simulated annealing or heuristic search do not use deeper knowledge of circuit structure and properties, simulation is the only way (used as part of the fitness function) to direct the search towards a circuit that satisfies given requirements. The results of the simulation are compared with the circuit specification many times in the loop of the search program. The same is true for quantum fault simulations. As we see, in all these applications the simulation of quantum circuits must be very fast and the computer memory should be large. On the other hand, using standard matrix representations, matrix operations on unitary matrices are slow, thus new methods and representations should be found to allow for very fast simulation that does also not consume too much memory. This is attempted by two fundamental validation methods: (1) acceleration of standard operations by using special FPGA-based hardware emulators, parallel computers or processor networks for simulation, (2) creating new advanced data structures such as quantum decision diagrams for software simulators on standard computers to represent quantum data more efficiently. These data structures, such as QUIDDs from [53] allow for implicit parallelism when executing Kronecker multiplications on them. QUIDDs are based on the decision diagrams used in CAD of digital

logic: ADDs and MTBDDs, being both generalizations of the famous Binary Decision Diagrams (BDDs). Similarly as the classical decision diagrams utilize similarity patterns in parts of truth tables (cofactors), the efficiency of decision diagrams comes from recognizing and reusing identical rectangular sub-patterns in unitary matrices. It is well-known that in classical design automation many breakthroughs were obtained because of using efficient data structures such as Binary Decision Diagrams. It is believed that in future other decision diagrams may be used to represent and manipulate quantum circuits. This will lead to efficient simulators, programs for formal verification, test generation, and synthesis. Finally, new quantum algorithms will be analyzed or even automatically created thanks to the expected power of these tools. It may be also predicted that basic software engines used successfully in classical CAD (such as for instance Satisfiability (SAT) or Automatic Test Pattern Generation (ATPG) methods) may be used to deal with quantum circuits. Also, the fast simulators based on new types of decision diagrams, such as [53] should be in future parallelized and possibly accelerated in Field Programmable Gate Array technology (FPGA) based boards. Even before quantum computers will become available, their emulations on standard computers and ASIC/FPGA may prove useful to solve some practical problems. The availability of powerful simulation, synthesis and analysis software tools will be unavoidable for the success of creating new Computational Intelligence models as well. All existing tools are now far insufficient for these tasks.

4. Synthesis, Testing and Diagnosis of Quantum Circuits

Several issues related to synthesis of quantum circuits have been already mentioned above. Important problem is that of designing permutative circuits [45] since they have application in oracle design, for instance for Grover algorithm. These circuits are synthesized with or without ancilla bits, starting from either Boolean or multiple-valued specification, reversible or not. If the initial function is not reversible, it should be first converted to reversible or this conversion is a part of the synthesis algorithms. These various conditions lead to several types of algorithms. So far, the synthesis methods included standard search approaches such as exhaustive search, A*, simulated annealing, Genetic Programming, Genetic Algorithm, Bacteria Foraging and others, [36,49]. Research has been also done on adapting XOR logic methods such as Fixed Polarity Reed-Muller expression, Exclusive Sum of Products and Galois Field logic minimization [31,32]. The most studied so far is the MMD algorithm for permutative circuits [15] for which convergence can be proved. There is some work on applications of group theory [55] and SAT-based approaches [29]. Interesting new research is on synthesis arbitrary unitary matrices to arbitrary quantum gates [7,46,54]. Truly quantum circuits (those for arbitrary unitary matrices) are also designed using above mentioned stochastic and search algorithms. The problem of their design is much harder than designing permutative quantum circuits. There are no published results in the area of realization of special types of functions, systematic synthesis methods for spectral transforms or arithmetic circuits, classification of reversible and quantum circuits and formal verification of such circuits. There is also some work on synthesizing circuits from high-level quantum languages, being counterparts of register-transfer level descriptions. Pioneering work in the area of testability of reversible logic [42] showed that such circuits are much better testable than irreversible circuits. This is because every test covers half faults and every fault is covered by half tests. The reversible circuits are then "transparent" to faults, making them well observable and controllable. It was also shown [44] that fault localization in reversible circuits is easier. The preliminary results on testing binary quantum circuits are in [4] and on fault localization of quantum circuits in [44]. Extremely high testability of quantum circuits was demonstrated for specific types of oracles; for instance a Positive-Polarity Reed-Muller like quantum circuit oracle can be tested in just five tests [4]. In general, the basic idea is to generalize classical approach to test. The good circuit is simulated. Next every possible quantum fault is inserted and the circuit with fault is simulated in Hilbert space (no measurement). The fault model is inserting arbitrary matrix in place of fault, this allows to simulate many different types of faults, such as Pauli rotations, missing controls in Controlled gates and other [4]. All possible measurement values are calculated with their probabilities. The comparison of a measurement from the unitary matrix of a correct circuit and a circuit with fault determines which input combinations (tests) give different measurements. Testing may be done in Walsh spectrum domain when superposed tests are created using quantum BIST preprocessor before the circuit and quantum BIST after. Some BIST circuits improve linearity of the circuit in order to allow disentanglement of outputs. Observe that in contrast to standard testing and reversible circuits testing, there are three types of faults in quantum domain: (1) faults that can be detected deterministically, (2) faults that cannot be detected (like global phase faults), and (3) faults that can be detected by repeated application of tests, possibly with special measurement gates. In some cases these faults are detected only with certain probabilities. Thus, in general, quantum testing is probabilistic testing.

5. Quantum Computational Intelligence (QCI)

The two most famous quantum algorithms to date were created by Peter Shor [47] and Lov Grover [20,21]. The first algorithm is for factoring integers and it produces an exponential computational speedup over classical algorithms, thus can break the RSA encryption techniques. The Grover's algorithm searches an unordered list of data, to find a particular item. It has a provable quadratic speedup over the best classical algorithm. It is like looking for name of a person in yellow pages knowing only his telephone number. In contrast to Shor's algorithm that has only few applications, Grover's algorithm can be used to speed-up arbitrary search problem from worst case complexity of N to $N^{½}$. It is just required that the algorithm designer builds a quantum permutative circuit representing an appropriate oracle for the problem. Usually, oracle gives only yes/no response. An interesting research topic is this, "How can Shor, Grover and other quantum algorithms be applied in the field of Computational Intelligence?" Because quantum computing is in every particular instance at least as powerful as standard computing, it is very reasonable to look for quantum counterparts for all concepts created in past in algorithm design, artificial intelligence, machine learning, computational intelligence or soft computing. Many quantum algorithms such as quantum counting, maximum, minimum and search algorithm other than Grover can speed up many NP-complete and NP-hard problems [10]. The name NP means non-deterministically polynomial, because there are no deterministic algorithms to solve NP problems in polynomial time (w.r.t the size of the problem). Any problem in the NP-complete class can be transformed into any other problem in this NP-com-

plete class using polynomial number of steps. The algorithm invented by Lov K. Grover for searching an unstructured database is of little use in its original formulation, but it started many practical applications because of the generality of its main idea - phase amplification. For instance, Grover himself extended his algorithm [9,22,23] for the structured search problem, one of the main tough research issues in AI, with a multitude of important and practical applications, including all mentioned here. Many interesting papers about quantum search using problem structure were written by Hogg and collaborators [26,27,28]. Boyer developed bound for quantum searching algorithms [5]. The class of NP problems includes graph coloring, satisfiability, planning, set covering, combinatorial optimization, tautology verification, finding best fixed polarity Reed-Muller form, and many other problems that are useful for instance to solve the synthesis and optimization problems from section 4 of this paper. The quantum search algorithms can be used to solve the "constraint satisfaction problems" into which all other NP-complete algorithms can be reduced [10]. In a constraint satisfaction problems (SAT is the simplest example, graph coloring is another one) we deal with multi-valued variables and constraint rules on value relations between values of subsets of variables (relations like, "two adjacent nodes in a graph should have different colors"). In other words, one has to find assignment of values to all variables so that all constraints are satisfied. All such problems can be reduced theoretically to SAT, but this is not necessarily the best way to solve them. A polynomial speedup over classical algorithms may be enough to solve many currently intractable problem instances [35].

Because quantum circuits, the main concept of quantum computing, are a powerful generalization of circuits in classical computing, researchers are systematically generalizing all the fundamental concepts of computing to involve quantum mechanics formalism-based or just "quantum inspired" ideas in one way or another. And thus; a quantum circuit is a generalization of a combinational Boolean circuit, multiple-valued circuit or continuous logic (fuzzy) circuit, quantum automata (various formalizations) generalize finite state machines, quantum turing machine generalizes turing machines and probabilistic turing machines, and so on. The same tendency is seen in Computational Intelligence. Its concepts and algorithms are being generalized to those of the Quantum Computational Intelligence (QCI). And thus; quantum neural networks [16,34], quantum associative memories [51], quantum Bayesian nets [50], quantum games, quantum markets, quantum agents, quantum formulas [6], quantum fuzzy networks, quantum spectral transforms and networks, quantum evolutionary algorithms, and many others have been created and are actively investigated both theoretically, using software simulators, hardware emulators and in real quantum circuits. Observe that a common point to all these generalization is the concept of a network (circuit, machine), which can be systematically designed, adapted, learned or evolved. Computational Intelligence area gives us many powerful metaphors that can be translated to quantum computing concepts in a straightforward way, leading to more powerful (so far only mathematically) systems.

Because laws of quantum mechanics proved useful to improve algorithmic performance of some NP problems, there is a high probability that more problems will find efficient solutions in quantum domain. Quantum-Neural Algorithms have been introduced, including Quantum Associative Memories of Ventura and Martinez [51], Competitive Learning in Quantum System by Ventura [52] and Perus [43]. While neural net processes real values, quantum NN processes complex values. It includes therefore standard NN and binary computers as special cases, but thanks to superposition and entanglement can do much more. Weights that are complex values will allow to express much more and higher order information. Totally new algorithms can be invented for learning and for using such nets. QuAM is analogous to a linear associative memory but all neurons are quantum mechanical gates. Because previous work on computational learning and particularly constructive induction designs arbitrary structures from arbitrary gates, it is applicable also to quantum structures. New algorithm to synthesize quantum circuits from examples can be created, such as those that generalize Ashenhurst Curtis decomposition where the problem is represented by a strongly unspecified function or relation. QuAMs are worse than classical algorithms on generalization, and decomposition based algorithms are very good in generalization for classical data. Therefore it is possible that by extending model of QuAMs, a more general quantum structures will be found that will have good properties of QuAMs such as storing exponential number of patterns but will be also good in generalizing. It is well known that there exist animals with very few neurons, such as nematode worms. Still they can exhibit much more complex behaviors that a robot controlled by few neurons. The neuron used in NN theory is thus a big simplification of real neuron, and it is possible that quantum computing is used in brains of animals. In any case, the fact that actual neurons are more powerful than their current models is a powerful argument to investigate generalized models of neurons - especially quantum neurons. The results of simulating quantum Genetic Crossover operators suggest that indeed quantum computation can speed up the search for solutions to the traveling salesman problem. Several successful experiments of various variants of Quantum-inspired GA have been described for several applications [24]. In [17] quantum algorithms for searching trees are discussed, there are examples of trees for which the classical algorithm requires time exponential in n, but for which the quantum algorithm succeeds in polynomial time. Spectral Associative Memories (SAMs) are classical networks inspired by quantum mechanics and proposed by Spencer [48]. They are quantized frequency domain formulations of conventional Contents Addressable Memories (CAMs). Non-local connectivity is made virtually by spectral convolution. In classical CAMs attractors scale quadratically or polynomially. In contrast, SAMs scale linearly with memory dimension. One model of the neuron [37,38] is based on quantum holography [12]. Phase is not only the essential parameter of physical significance, as in the postulated model of quantum neural information processing, but the essential means by which holograms i.e. the 3-dimensional representations of objects may be encoded, decoded or transmitted. Concluding, there are dual influences of CI and quantum computing; the quantum ideas can be used to create powerful quantum-inspired algorithms to solve many types of problems in EDA and robotics. On the other hand the ideas and algorithms from many classical computer science areas can be now used in quantum domain [33] or transformed and extended to quantum domain. Although several ideas have been already published, there is very little operational software packages that use them and very little has been published on comparison of these methods among themselves and with respect to classical methods.

6. Conclusions

This paper is related to what is an emerging area of systematically designing quantum circuits and algorithms.

Similarly as in classical computing, there will appear sub-areas of quantum algorithm development, quantum computer architecture, high level quantum synthesis, logic level quantum synthesis, quantum physical design, quantum test, quantum verification, quantum simulation, quantum software-hardware co-design, quantum automatic learning from examples, quantum neural networks, data mining, and so on. We outlined some subjective choice of recent papers and research directions as a potential base of future research in quantum computer engineering. It was our goal to show that the conventional logic synthesis, test and machine learning methods, for both binary and multiple-valued logic, form a powerful base of new approaches. Similarly the data structures like decision diagrams or fundamental algorithms such as satisfiability or reachability analysis continue to have their role. Many CI methods will be used to develop, design and optimize future quantum computers.

References

[1] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Schor, T. Sleator, J. Smolin and H. Weinfurter, Elementary Gates for Quantum Computation, Physical Rev (A), no. 52, pp. 3457-3467, March 1995.

[2] C. H. Bennett et al., Strengths and Weaknesses of Quantum Computing, SIAM J. Computing, Vol. 26, pp. 1510-1523, 1997.

[3] C. H. Bennett and D. P. DiVincenzo, Quantum Information and Computation, Nature, vol. 404, pp. 247-255, March 2000.

[4] J. Biamonte, M. Perkowski, A Quantum Test Algorithm, Quantum Physics, quant-ph/0501108.

[5] M. Boyer et al., Tight Bounds on Quantum Searching, Proc. Workshop Physics and Computation (PhysComp96), New England Complex Systems Inst., Cambridge, Mass., pp. 36-43, 1996.

[6] N. Bshouty and J. Jackson, Learning DNF over the Uniform Distribution Using a Quantum Example Oracle, Proc. Eighth Ann. Conf. Computational Learning Theory, ACM Press, NY, 1995, pp. 118-127.

[7] S.S. Bullock, D.P. O'Leary, G.K. Brennen, "Asymptotically Optimal Circuits for d-level Systems," Physical, Review Letters, vol. 94, 230502 (2005). Also quant-ph/0410116.

[8] A.V. Burlakov, M.V. Chekhova, O.V. Karabutova, D.N. Klyshko, and S.P. Kulik, Polarization state of a biphoton: quantum ternary, Phys. Rev. A 60, R4209, 1999.

[9] N. J. Cerf, L. K. Grover, C. P. Williams, Nested Quantum Search and NP-Hard Problems, Appl. Algebra Eng. Commun. Comput. 10, (4/5), pp. 311-338. 2000

[10] P. Cheeseman, R. Kanefsky, and W.M. Taylor, Where the Really Hard Problems Are, Proc. Int'l Joint Conf. AI, Morgan Kaufmann, San Francisco, pp. 331-337, 1991.

[11] I. L. Chuang, N. Gershenfeld, and M. Kubinec, Experimental Implementation of Fast Quantum Searching, Physical Rev. Letters, Vol. 80, pp. 3408-3411, 1998.

[12] B. E. P. Clement et al., The Brain as a Huygens Machine, Informatica, 23(3), 1999

[13] D. Deutsch, Quantum Computational Networks, Proc. Roy. Soc. Lond. A425, pp.73-90., 1989.

[14] D. P. DiVincenzo, Quantum Computation, Science, vol. 270, pp. 255-261, October 1995.

[15] G. Dueck, D. Maslov, and D. M. Miller, A Transformation Based Algorithm for Reversible Logic Synthesis, Proc. DAC. 2003, Anaheim, CA, pp. 318-323

[16] A. Ezhov and D. Ventura, Quantum Neural Networks, in Future Directions for Intelligent Systems and Information Science (Ed. N. Kasabov), Physica-Verlag,

[17] E. Farhi and S. Gutmann, "Quantum Computation and Decision Trees," Physical Rev. A, Vol. 58, No. 2, 1998, pp. 915-928.

[18] R.P. Feynman, QED: The Strange Theory of Light and Matter, Princeton Univ. Press, Princeton, 1985.

[19] E. Fredkin and T. Toffoli, Conservative Logic, International Journal of Theoretical Physics, vol. 21, Nos. 3-4, pp. 219-253, 1982.

[20] L. Grover, A Fast Quantum Mechanical Algorithm for Database Search, Proc. ACM Symp. Theory of Computing, ACM Press, New York, 1996, pp. 212-219.

[21] L.K. Grover, Quantum Mechanics Helps in Searching for a Needle in a Haystack, Physical Rev. Letters, Vol. 78, pp. 325-328, 1997.

[22] L. K. Grover, Rapid sampling though quantum computing. STOC. 1998, pp. 618-626.

[23] L. K. Grover, Quantum Search on Structured Problems, QCQC 1998, pp. 126-139.

[24] K-H. Han, J-H. Kim, Quantum-inspired evolutionary algorithm for a class of combinatorial optimization, IEEE Trans. Evolutionary Computation, 6 (6), pp. 580-593, 2002.

[25] S. Haroche and J.-M. Raimond, Quantum Computing: Dream or Nightmare? Physics Today, Vol. 49, pp. 51-52, August 1996.

[26] T. Hogg, B.A. Huberman, and C.P. Williams, eds., Frontiers in Problem Solving: Phase Transitions and Complexity, Artificial Intelligence, Vol. 81, 1996.

[27] T. Hogg, Single-Step Quantum Search Using Problem Structure, Los Alamos preprint quant-ph/9812049, Los Alamos Nat'l Lab, Albuquerque, N.M., 1998.

[28] T. Hogg, Solving Highly Constrained Search Problems with Quantum Computers, J. Artificial Intelligence Research, Vol. 10, 1999, pp. 39-66; http://www.jair.org/ abstracts/hogg99a.html.

[29] W.N.N. Hung, X. Song, G. Yang, J. Yang, and M. Perkowski, Quantum Logic Synthesis by Symbolic Reachability Analysis, Proc. 41 DAC, San Diego, California, June 2004. pp. 838-841

[30] S.C. Kak, Quantum neural computing. Adv. Im. and Electr. Physics, 94, pp. 259-313, 1995.

[31] F. Khan, M. Perkowski, Synthesis of Ternary Quantum Logic Circuits by Decomposition, Proceedings of 7th International Symposium on Representations and Methodology of Future Computing Technologies, RM 2005, University of Tokyo, September 5-6, 2005, pp. 114-118.

[32] M.H.A. Khan, M.A. Perkowski, and P. Kerntopf, Multi-Output Galois Field Sum of Products Synthesis with New Quantum Cascades, Proc. 33rd ISMVL, Tokyo, May 16-19, 2003, pp. 146-153.

[33] I. E. Lagaris, A. Likas and D.I. Fotiadis, Artificial neural network methods in quantum mechanics, Computer Physics Communications, vol. 104, pp. 1-14, 1997.

[34] H. J. Levy and T. C. McGill, A feedforward artificial neural network based on quantum effect vector-matrix multipliers. IEEE Transactions on Neural Networks, 4(3): pp. 427--433, May 1993.

[35] I. Levner, Quantum Computing (Recent Developments in Quantum Computational Intelligence). Preprint.

[36] M. Lukac, M. Perkowski, H. Goi, M. Pivtoraiko, C.H. Yu, K. Chung, H. Jee, B-G. Kim, and Y-D. Kim, Evolutionary approach to Quantum and Reversible Circuits synthesis, Artificial Intelligence Review, 20, pp 361-417, 2003. Kluwer Academic Publishers.

[37] P. Marcer, The Neuron: a Computational Model Utilizing Quantum Devices, Nanobiology, 1, pp. 289-291, 1992.

[38] P. Marcer, A Quantum Leap to Advances in Pattern Recognition, Proc. ICAPR'98, 23-25 Nov. Plymouth, UK, ed. Singh S., pp. 375-384, 1998.

[39] G. J. Milburn, Quantum Optical Fredkin Gate, Phys. Rev. Lett, vol. 62, no. 18, pp. 2124-2127, May 1989.

[40] A. Muthukrishnan, and C. R. Stroud, Jr., Multivalued logic gates for quantum computation, Physical Review A, Vol. 62, No. 5, Nov. 2000, 052309/1-8.

[41] M. Nielsen and I. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000.

[42] K.N. Patel, J.P. Hayes and I. Markov, "Fault testing for reversible circuits," Proc. VLSI Test Symp. (VTS 03), Napa, CA, pp. 410-416, April 2003.

[43] M. Perus, Neuro-Quantum Parallelism in Brain-Mind and Computers, Informat., 20, pp. 173-183, 1996.

[44] D. Pierce, J. Biamonte, M. Perkowski, Test Set Generation and Fault Localization Software for Reversible Circuits, Proceedings of 7th International Symposium on Representations and Methodology of Future Computing Technologies, RM 2005, University of Tokyo, September 5-6, 2005, pp. 42-49.

[45] V.V. Shende, A.K. Prasad, I.L. Markov, J.P. Hayes, Reversible Logic Circuit Synthesis, Proc. 11th IEEE/ACM Intern. Workshop on Logic Synthesis, 2002, pp. 25 - 130.

[46] V. V. Shende, S. S. Bullock, I. L. Markov, ``A Practical Top-down Approach to Quantum Circuit Synthesis,'' Proc. Asia and South Pacific Design Automation Conference, pp. 272-275, Shanghai, China, 2005, quant-ph/0406176.

[47] P.W. Shor, Algorithms for Quantum Computation: Discrete Logarithms and Factoring, Proc. 35th Symp. Found. On CS, IEEE Computer Soc. Press, Los Alamitos, Calif., pp. 124-134, 1994.

[48] R. Spencer http://amesp02.tamu.edu/~rspencer/

[49] L. Spector, H. Barnum, H.J. Bernstein, and N.Swamy, Finding a better-than-classical quantum AND/OR algorithm using genetic programming, Proc. 1999 Congress on Evolutionary Computation, Vol. 3, pp. 2239-2246, Washington DC, 6-9 July 1999, IEEE, Piscataway, NJ.

[50] R. Tucci, Quantum Bayesian Nets, Int'l J. Modern Phys, Vol. B9, 1995, pp. 295-337.

[51] D. Ventura, and T. Martinez, A Quantum Associative Memory Based on Grover's Algorithm, International Conference on Artificial Neural Networks and Genetic Algorithms, April 1999.

[52] D. Ventura, Implementing Competitive Learning in a Quantum System, Proc. Int'l Joint Conf. Neural Networks, IEEE Computer Soc. Press, Los Alamitos, Calif., 1999.

[53] G.F. Viamontes, I. L. Markov and J. P. Hayes, Graph-based simulation of quantum computation in the density matrix representation, arXiv:quant-ph/0403114V2, 16 Mar 2004.

[54] J. J. Vartiainen, M. Möttönen, M. M. Salomaa, Efficient decomposition of quantum gates, quant-ph/0312218, Phys. Rev. Lett. 92, 177902 (2004).

[55] G. Yang. X. Song, M. Perkowski and J. Wu, Realizing ternary quantum switching networks without ancilla bits,Journal of Physics A: Mathematical and General, 2005.

Marek A. Perkowski got his Ph.D. in automatic control in the Institute of Automatics at the Electronic Engineering Department at Technical University of Warsaw, Warsaw, Poland. He served on the Faculty of Warsaw Technical University, University of Minnesota. Since 1984 he was an associate professor and was promoted in 1994 to full professor at the Department of Electrical and Computer Engineering, Portland State University, Portland, Oregon. He was a consultant and programmer in the areas of logical synthesis and FPGA design in EDA industry, working for Intel, Sharp Microelectronics, GTE, Cypress Semiconductor and other companies. In 1994 he worked for Wright Laboratories of U.S. Air Force in Dayton, Ohio in the area of pattern recognition. His main research areas include logic synthesis, computer aided design, efficient algorithms, intelligent robotics, Artificial Intelligence and pattern recognition. Since 1998 Dr. Perkowski is interested mainly in quantum computing and its applications to computational intelligence and robotics.