A Hierarchical Approach to Computer-Aided Design of Quantum Circuits

Marek Perkowski,+* Martin Lukac,* Mikhail Pivtoraiko,* Pawel Kerntopf, & Michele Folgheraiter ^,
Dongsoo Lee, + Hyungock Kim,+ Woong Hwangbo, Jung-wook Kim+ and Yong Woo Choi.+

*Department of Electrical and Computer Engineering, Portland State University, Portland, OR, 97201. USA, +Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, 373-1, Guseong-dong, Yuseong-gu, Taejeon, 305-701, Republic of Korea, & Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland, ^ Dipartimento di Elettronica ed Informazione (DEI), Politecnico di Milano, Piazza Leonardo da Vinci 32, Milano, Italy.

Abstract:
A new approach to synthesis of quantum logic circuits has been presented in this paper. This approach produces better results than the previous approaches based on classical reversible logic and can be easier tuned to any particular quantum technology such as nuclear magnetic resonance (NMR). The approach is based on the following principles: (1) design of (complex) gates using approach that links evolutionary, combinatorial and human interaction approaches, (2) design of circuits with these gates using evolutionary and search methods, (3) macro-generation of complex gates to lower level quantum primitives, (4) applying technology-related optimizing tautological transforms that take realistic quantum layout constraints, (5) calculating realistic costs related to quantum realization, (6) synthesis of incompletely specified functions.

1. Introduction

While quantum mechanics and quantum computing are established research areas, automated quantum circuit synthesis is still only at the beginning of its exploration [9,30,45,48,49,50,51]. In quantum computation we use quantum bits (qubits) instead of classical binary bits to represent information. This gives the advantage of being able to perform massively parallel computations in one time step. The design of quantum circuits of practical size is still technologically impossible (the maximum number of qubits in year 2002 is 7), but the progress is fast and there are no arguments based on physics against the possibility of building powerful quantum computers in the future. Therefore quantum computing area of research is recently flourishing.

Finding an effective and efficient method of designing quantum circuits can have two immediate application areas:
(1) Optimizing quantum circuits for NMR [19,20,21,22,42,43], ion trap, quantum dot, cavity quantum electrodynamics or Si-based nuclear spin quantum computer technologies that exist already in practice. Each of these technologies has different minimization requirements and all of them require minimizing both the width and the length of scratchpad register (number of qubits processed). Width is absolutely critical and length is also important because of decoherence. Circuit reduction is very important for present technology, so exact or sub-minimal methods should be developed for small circuits, less than 7 variables. This is a very small number from the standard CAD algorithms point of view, but the synthesis problem for quantum logic is much more difficult.
(2) Modeling quantum computers in FPGA-based reconfigurable hardware for speeding-up computations that are very inefficient on standard computers [31]. Since for current FPGA technologies only small quantum circuits can be emulated or simulated, the requirements are similar to point (1).

2. Quantum circuit synthesis is not the same as standard reversible logic synthesis

The major difference between quantum logic and binary logic is the concept of the information itself. While the classical (binary or multi-valued) representations of information are precise and deterministic, in Quantum Computing (QC) the concept of bit is replaced by the qubit. Unlike classical bits that are realized as electrical voltages or currents present on a wire, quantum logic operations manipulate qubits [50]. Qubits are microscopic entities such as a photon or atomic spin. Boolean quantities of 0 and 1 are represented by a pair of distinguishable different states of a qubit. These states can be a photon's horizontal or vertical polarization denoted by $|\leftrightarrow\rangle$ or $|\updownarrow\rangle$, or an elementary particle's spin denoted by $|\uparrow\rangle$ or $|\downarrow\rangle$ for spin up and spin down, respectively. After encoding these distinguishable quantities into Boolean constants, a common notation for qubit states is $|0\rangle$ and $|1\rangle$. Qubits exist in a linear superposition of states, and are characterized by a wavefunction ψ. As an example, it is possible to have light polarizations other than purely horizontal or vertical, such as slant 45° corresponding to the linear superposition of $\psi=\frac{1}{2}[\sqrt{2}|0\rangle+\sqrt{2}|1\rangle]$. In general, the notation for this superposition is $\alpha|0\rangle+\beta|1\rangle$. These

intermediate states cannot be distinguished, rather a measurement will yield that the qubit is in one of the basis states, $|0\rangle$ or $|1\rangle$. The probability that a measurement of a qubit yields state $|0\rangle$ is $|\alpha|^2$, and the probability is $|\beta|^2$ for state $|1\rangle$. The absolute values are required since, in general, α and β are complex quantities. Pairs of qubits are capable of representing four distinct Boolean states, $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$, as well as all possible superpositions of the states. This property is known as "entanglement", and may be mathematically described using the Kronecker multiplication (tensor product) operation $\otimes$ [59]. As an example, consider two qubits with $\psi_1=\alpha_1|0\rangle+\beta_1|1\rangle$ and $\psi_2=\alpha_2|0\rangle+\beta_2|1\rangle$. When the two qubits are considered to represent a state, that state ψ_{12} is the superposition of all possible combinations of the original qubits, where

$$\psi_{12}= \psi_1\otimes\psi_2 = \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \alpha_2\beta_1|10\rangle + \beta_1\beta_2|11\rangle. \qquad (1)$$

Superposition property allows qubit states to grow much faster in dimension than classical bits. In a classical system, n bits represents 2^n distinct states, whereas n qubits corresponds to a **superposition** of 2^n states. Observe also that in the above formula some coefficient can cancel, so there exist a constraint bounding the possible states in which the system can exist.

In terms of logic operations, anything that changes a vector of qubit states can be considered as an operator. These phenomena can be modeled using the analogy of a "quantum circuit". In a quantum circuit wires do not carry Boolean constants, but correspond to pairs of complex values, α and β. Quantum logic gates of this circuit map the complex values on their inputs to complex values on their outputs. Operation of quantum gates is described by matrix operations. Any quantum circuit is a composition of parallel and serial connections of blocks, from small to large. Serial connection of blocks corresponds to multiplication of their (unitary) matrices. Parallel connection corresponds to Kronecker multiplication of their matrices. So, theoretically, the analysis, simulation and verification are easy and can be based on matrix methods. Practically they are tough because of the problem dimension -exponential growth of matrices.

Synthesis problem can be formulated as decomposing hierarchically a given unitary matrix to serial and parallel connections of smaller matrices, until basic directly realizable quantum primitives are reached. This problem is very difficult in such basic formulation and therefore several special methods are being developed. Probabilistic calculations based on this representation are used in only very small quantum computers so far, but it was verified that information can be represented as a superposition of states of single qubits, and that in one time step operations can be performed on several qubits. Moreover it was shown [50] that any quantum computing has to be reversible, which affects all synthesis methods. These all contribute to difficulties in understanding the concepts of quantum computing and creating efficient analysis, simulation, verification and synthesis algorithms for QC. Generally, however, we believe that much can be learned from the history of Electronic Computer Aided Design and the lessons learned should be used to design efficient CAD tools for quantum computing.

3. Quantum CAD – a hierarchical approach.

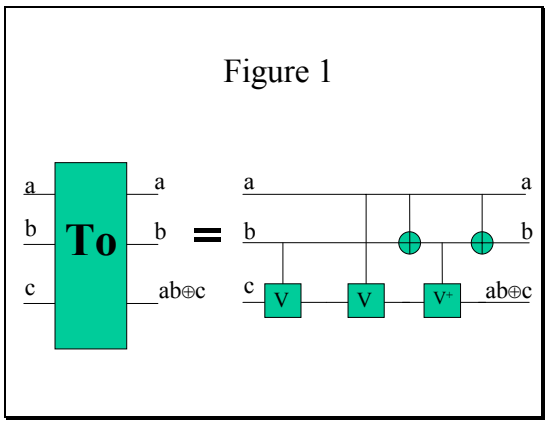
Building quantum computers becomes more and more technical rather than only scientific issue, and the methods developed to design them, such as formal representation, modeling and synthesis will have applications not only to quantum computing but also to DNA and other nano-technologies because of their reversible nature. In this paper we focus on the synthesis of arbitrary quantum circuits (and quantum gates in particular) of small size, less than seven variables. As a special case, we design circuits from a class of permutation gates which have unitary matrices being permutation matrices. Such circuits correspond to classical reversible Boolean functions, but in contrast to previous approaches our methods deal with incompletely rather than completely specified Boolean functions. The reader has, however, to keep in mind that our design methods are for general quantum circuits, with arbitrary unitary matrices. The presented methods have been specialized to some classes such as f-controlled-phase-shift gate circuits [22], and generalized to multiple-valued quantum logic and mixed multiple-valued logic [1,2,17,39,40,41].

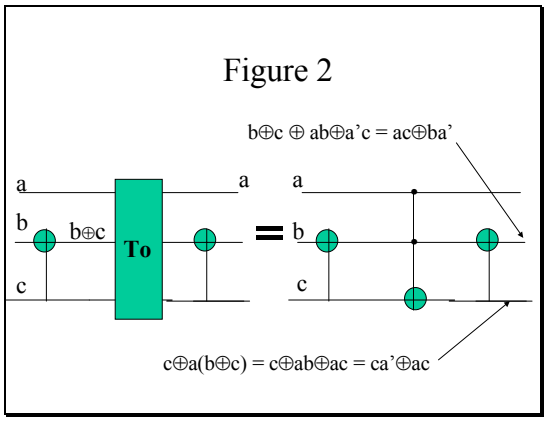
While most of synthesis papers deal with general reversible logic [13,25-27,46] here we are interested in quantum realizations. **We want to demonstrate and emphasize that it is not true that current binary reversible logic synthesis methods can be directly applied to permutation quantum circuits.** This way we want to create new improved methods that will be more practical for optimizing quantum sequences in current existing technologies such as NMR [19-22,42,43]. We thus show (see also [52]) that Toffoli (CCNOT) and Fredkin are not always the best gates for quantum computing. Multi-input Toffoli gate looks simple in a diagram, but takes a lot of gates when macro-generated to 3*3 gates with auxiliary constants – such gates are not primitives. Therefore we propose a hierarchical approach – design of powerful gates first. Synthesis should be performed on one hand at a lower level of quantum primitives such as CNOT (Feynman), controlled square-root-of-not and Hadamard (or even lower), and on the other hand at the level of more powerful reversible gates, such as the complex gates introduced in [1,2,6,14-18,25,29,36-41,52]. Using diverse and more complex gates simplifies search. The transformation from

optimal reversible circuit (on any level of gates) to the minimal quantum sequence for NMR programming is far from being trivial, and so far no combinatorial optimization algorithms have been created for them. For instance in NMR every gate must be constructed from 2-qubit gates, but some pairs of (quantum) wires of a (quantum) array cannot be inputs/outputs of a gate and thus swap gates must be added. One swap gates costs three Feynman gates so it is costly. A good synthesis method should then involve not only logic synthesis but also "Quantum Layout Synthesis", which in this particular case means transforming circuit in such a way that gates are not connecting the forbidden pairs of quantum wires. This leads to problems of gate ordering, input variable ordering, local transformations, removal of swap gates (planarization) and others. This way, the gate cost assumptions that can be found in general quantum papers (and not in specialized papers about NMR technology) are far too approximate and can lead to highly non-optimal sequences.

Here we propose methods that are tuned to real technology. These optimization problems are quite similar to technology mapping and physical design areas in classical CAD. So far, no systematic CAD approaches have been proposed for them and they are only solved *ad hoc* by physicists who build NMR computers [22]. In our CAD tools, the evolutionary, combinatorial search, and human-oriented interactive methods are combined. We were able to synthesize completely automatically more complex circuits than those found by previous programs, for instance the EPR circuits and Toffoli or Margolus gates.

4. New Quantum Gates and their Cost Functions for the Optimization Algorithms



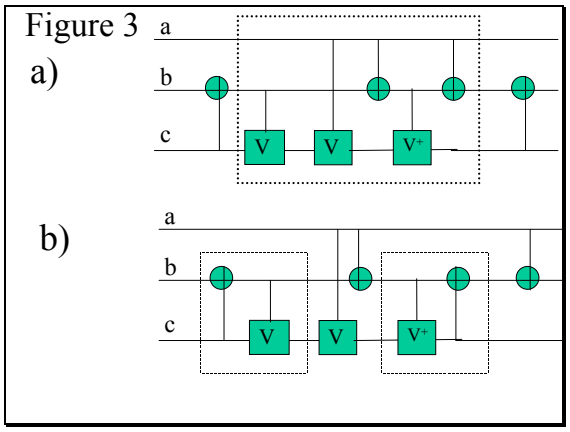


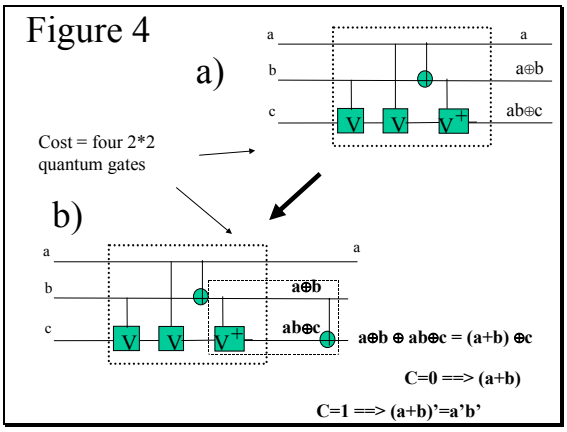
An important problem, not discussed so far by other authors, is the selection of the cost functions to evaluate the quantum circuit designs. Although the detailed costs depend on any particular realization technology of quantum logic, so that the cost relations between for instance Fredkin and Toffoli gates can differ in NMR and ion trap realizations, the assumptions used in several previous papers; that each gate costs the same, or that the cost of a gate is proportional to the number of inputs/outputs, are both far too approximate. In this paper we illustrate more precise cost functions for gates that are used in our optimization algorithms (even more accurate costs can be found in [39,40]). Here, we follow the footsteps of previous papers in quantum computing [3,19-22,42,43,47] and will realize all gates from 1* 1 and 2*2 gates. Moreover, according to [47] we assume that the cost of every 2*2 gate is the same. We assume also that a 1*1 gate costs nothing, since it can be always included to arbitrary 2*2 gate that precedes or follows it. Thus, in first approximation, every permutation quantum gate will be build from 1*1 and 2*2 quantum primitives and its cost calculated as a total sum of 2*2 gates used. Using the well-known realization of Toffoli gate with truly quantum 2*2 primitives, shown in Figure 1 [47], the cost of Toffoli gate is five 2*2 gates, or simply, 5. In Figure 1, V is a square-root-of-NOT gate (unitary matrix V) and V^+ is its hermitian. Thus V V creates a unitary matrix of NOT gate and V V+ = I (an identity matrix, describing just a quantum wire). The reader can analyze correctness of this construction by analyzing all possible values of inputs signals (the generalization of this gate to n-inputs without constant wires is shown in [3]). Now we will realize the Fredkin gate from the Toffoli gate. Using evolutionary synthesis methods discussed below, we synthesize the Fredkin gate using two Feynman and one Toffoli gate as in Figure 2. Substituting the Toffoli design from Figure 1 to Figure 2 we obtain the circuit from Figure 3a. After using the obvious EXOR-based transformation shown in the right (change of order of Feynman gates), the final circuit from Figure 3b is obtained. Observe that a cascade of two 2*2 gates is another 2*2 gate, so following [47] we obtain a circuit from Figure 3b with the cost of 5. (Each sequence in dotted lines has a cost of 1). Thus, the cost of Toffoli gate is exactly the same as the cost of Fredkin gate, and not half of it, as some authors assumed and as may be suggested by classical binary schemata of such gates, where the Toffoli gate includes a single Davio gate, while the Fredkin gate includes two multiplexers.

Encouraged with this observation, we calculated costs of other known gates [39,40,52] and came to similar

observations - the cost of Miller's Gate is 7 and not 9, as might be expected from its binary schematics using Feynman and Toffoli gates. Another realization of Miller's gate has an even smaller cost of 6. There are similar gates of cost 5. Similarly the costs of 3*3 gates by Kerntopf [14,15,36], Margolus [14], De Vos [15], Khan [16], and Maslow [6,25], costs of all 4*4 Perkowski's gates [18], and other gates from [15] can be calculated. Next observe that a new permutation quantum gate with equations: P = a, Q = a ⊕ b, R = ab ⊕ c can be realized with cost 4. It is just like a Toffoli gate from Figure 1 but without the last Feynman gate from right. This is the ***cheapest quantum realization known to us of a complete (universal) permutation gate*** and it is thus worthy further investigations. We found that the equation of this gate was known to Peres [14], but it has been not used in reversible or quantum computing.

Observe that algorithms from [13,46], when given the equation of Peres gate, would return the solution composed from Toffoli and Feynman gates, which would lead to clearly non-minimal quantum sequence. However, if the post-processing stage of macro-generation and next equivalence-based optimizing transformations were applied to the results from [13,46], then the minimum solution would be found. It is now a challenge for other researchers to find a cheaper universal 3*3 gate, that would use only 2*2 and 1*1 gates (it is known that several such universal sets of gates exist. [3]). Concluding, the synthesis of NMR circuits should be not restricted to Toffoli, Feynman and NOT gates as in [13,25-27,46].





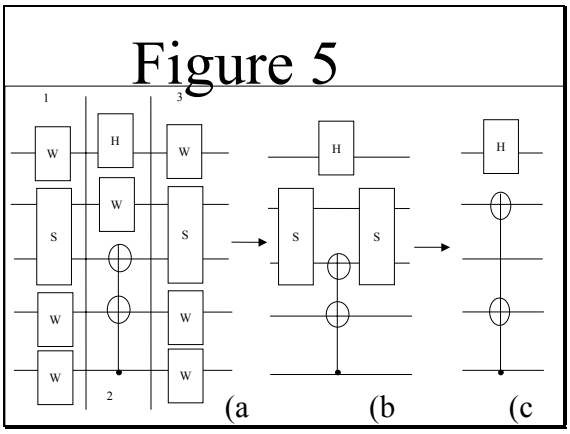
5. Frame-based search and evolutionary gate generator s

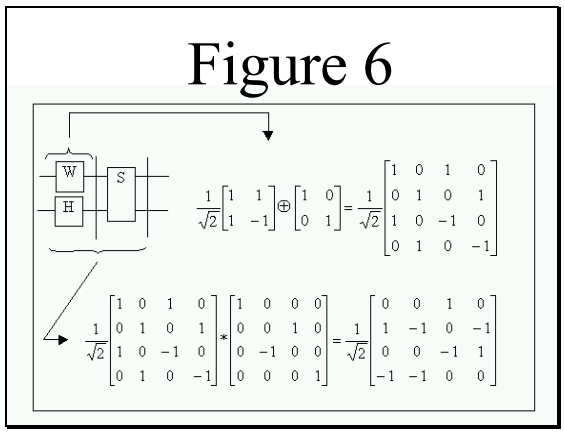
Now observe in Figure 4, that the Peres Gate (Fig. 4a) with Feynman gates located in all possible pairs of terminals at its periphery creates powerful and inexpensive new gates. For instance, by locating Feynman gate with EXOR up on output wires b and c, generates a gate with equations: a = a, b = (a+b) ⊕ c, c = ab ⊕ c. Again, the Feynman gate can be combined with the V^+ gate so the cost of the combined new gate is again only four. Another possibility of connecting Feynman gate to the outputs b, c of the Peres gate is shown in Figure 4b. There exist also two possibilities of connecting Feynman gate to inputs b, c, which similarly lead to two new gates, and four possibilities of connecting Feynman gates to both inputs b, c and outputs b, c, which leads to four new gates. Several other new gates can be created for another input/output terminal pairs in Figure 4. There exist very many combinatorial possibilities of connecting Feynman (and other) gates to inputs and outputs of basic gates, as was shown in section 4.

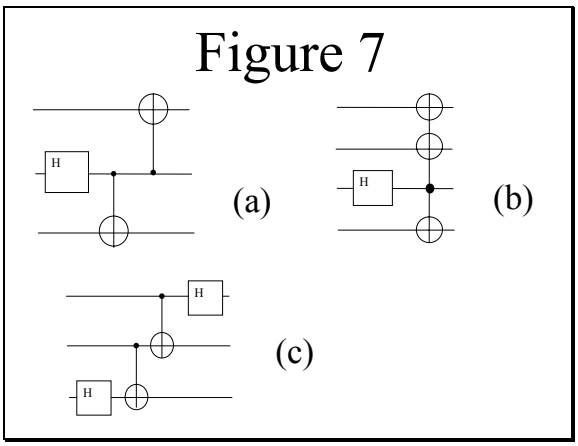
The above illustrated procedure of systematic adding gates to structures has been automated [39,40]. Our frame-based search generator allows exhaustive and partial search, and it is interactively called by the user, who declares structure, gates, constraints and search parameters like depth, search type (depth first, breadth first, OR versus AND/OR, etc), number of node children, etc. (reader interested in pseudo-code and details is referred to [7,33-35]. The user selects some seed gate and defines symbolically all possible structures that can be build by connecting additional gates to it. The types of these gates, the terminals of their abutment and the names of optimizing (combining) transformations to be applied, as well as the costs of gates, are the parameters of the program. These gates are like slots in a frame structure known from AI programming. The sequence of slots creates a GA's chromosome. This new method, which we call informed GA, combines evolutionary programming and frames, as well as the user being part of the design loop. The details about evolutionary algorithms for both truly quantum circuits and permutation circuits can be found in [23] and [24], respectively. All automatically created new gates, or gates satisfying some criterion (like small cost or (partial) satisfaction of unitary matrix) are displayed on screen to the user, together with their cost functions. This way, a library of powerful complex permutation gates is created. Some gates in this library are geared towards structured design methods (such as Kerntopf gate used in

regular structures called Nets and other gates used in Lattices [2,15,36,38]). All gates have good realizations (short and completely layout-realizable sequences) in a given particular quantum technology, such as especially NMR.

In all our programs we use similar encodings of quantum arrays. Also, the same matrix-operation based verification scheme from the fitness evaluation subroutine of GA is used. An example of our encoding is shown in Figure 5. It illustrates a transformation of a QC from the encoded chromosome (5a) to a final quantum circuit notation representation of this QC (5c). Here S is a Swap gate, H is a Hadamard gate, W is a wire. In Fig 5b there is one CCNOT (Toffoli) gate. In Figure 5a it is shown how the circuit from Figure 5c is encoded. As can be seen, there is no free space in the proposed encoding. Each place in the circuit is presented as a symbol of a unitary matrix of certain elementary quantum gate. A wire has a unitary identity matrix representation. While evaluating the fitness function, Kronecker products (tensor products) are executed on matrices of parallel gates (blocks, circuits), and standard matrix multiplications are performed on serial connections of gates. Each QC is parsed in parallel blocks, evaluated separately and finally multiplied together to give the final unitary matrix representation of the QC. This final matrix is next compared with the target matrix to evaluate their distance as a part of fitness function calculation. The example shown in Figure 5 illustrates a common inconvenience of encoding quantum circuits for genetic algorithms. A quantum gate CCNOT can be placed over three different arbitrary quantum wires in a quantum circuit. However with the encoding used, there is no information indicating what gates are connected to what wires, beside the order of the gates. To solve this problem we insert two Swap gates (one before and one after) the CCNOT. This implies that outside of the Swap gates the CCNOT seems like being on wires 2, 4 and 5, but the real CCNOT gate uses wires 3, 4 and 5. In order of being able to encode a QC without any additional parameters, the circuit is split into parallel blocks where each block can be evolved or interactively manipulated separately. Figure 5 illustrates also the Quantum Layout problem. Assuming that quantum wires 2 and 4 are a non-realizable pair, two swap gates as shown in Figure 5b should be inserted to make the circuit from Figure 5c realizable in this particular NMR technology. The cost of the circuit should then include the total cost of all swap gates, which can be very high in terms of 2-qubit gates (see [16,17,39]). Figure 6 illustrates how parallel and serial blocks correspond to Kronecker products $\oplus$ and matrix products * while cost (fitness) functions are calculated – this stage is a slowdown of the programs.







6. Synthesis, Macro-generation and Optimizing Transformations.

Synthesis of permutation circuits can be also performed using methods from [18,28,32,39,40]. Next the macro-generations from higher order permutation quantum gates (3*3, 4*4, and higher) to quantum gate primitives are applied. They are followed by tautology and equivalence transformations not on the level of permutation gates (as in [13,46]) but on the level of truly quantum 1*1 and 2*2 primitives. This allows to optimize the quantum circuits further. For instance, an n-input Toffoli gate can be build without constant inputs using only 2*2 primitives as a result of macro-generation [3]. These transformations generalize (but are in principle similar to) those presented in detail in [7,13,19-22,28,32,39,40,42,43,46] and will be not discussed here. Observe, however, that our transformations are more general than those from [13,46] because: (1) they apply to incomplete functions, (2) they exhaust in a sense space of transformations [28,32], (3) they are used for complex gates and macro-generation, (4) they apply also to truly quantum (non-permutative) gates [27,28], (5) they apply to general Galois Logic and not only to GF(2) [40]. The control of the transformations is based on methods from [7,33-35], thus the user can experiment with various transformation strategies. Many of them are based on obvious rules of EXOR algebra, such as some illustrated above, but other use Galois Field or Lee algebras. For instance, it can be shown how a Unified Complex Hadamard Transform of Rahardja and Falkowski [44], not investigated so far in quantum computing, can be optimized using non-permutation Hadamard gates, other gates and quantum primitives from [20,42].

7. Experimental results

The results are quite encouraging. In every case the GA found the requested gate, however in no case the automatically created chromosome was better than the circuit for which the corresponding target unitary matrix was created. Summary of results is shown in Table 1.

Number of inputs per q-gate	Number of generations	pM	pC	Real time (average 20 runs)	pM<0.2 Number of generations	Real time (average 20 runs)	Population size
1 - input	<50	0.4	0.6	< 30 seconds	<100	< 1 minute	50
2 - inputs	<50	0.6	0.4	< 30 seconds	<100	< 1 minute	50
3 - inputs	50 - 200	0.6	0.6	<1 minutes	<200	<3 minutes	60

Table 1: Results of experiments. Due to the similarity of results we grouped the results by the number of inputs/outputs of the requested q-gate. PM and PC are probability of mutation and crossover.

All circuit evolved were exact copies or at least had same number of wires, of the searched circuit. For small gates the convergence is logically faster, because of the restricted recombination between different gates. A gate with more inputs than the number of wires in the circuit was not tested. The 3 input gates test shows the same result, however with exponential time. The results are measures of average values over 20 runs. Depending on the circuit we were looking for, the times are, as predicted, increasing very fast with the increase of the number of QC inputs. Two configurations were tested. First, high probabilities of mutation (0.4-0.7) and crossover (0.4-0.6) were applied. The results are surprising because of so high mutation probability. The size of the GA population was set in range [50,100]. The local very high probability of mutation allows a fast dangerous search. However the runs were stopped as soon as a good solution was found and the best individual examined. A large random search with mutation used on a recombination problem seems to have a positive effect in a restricted search. The solution was found also when the mutation was of a small order, however the time of search raised as well.

Next step was to test composite circuits proposed by [45, 49]. We selected from literature three benchmarks shown in Figure 7. The first two are both circuits to produce EPR as in [45], the last is the "send" circuit originally proposed by [4] and evolved in [49].

The results are shown below in Table 2. We were able to find all searched circuits, however in this part of experimentation the starting set of gates was open. Our GA found for all benchmarks at least similar, if not better, results compared to the published results of the studied cases. Even if the number of generations grows exponentially, the real time still remains reasonable.

Number of inputs per q-gate	Number of generations	pM	pC	Real time (average 20 runs)	pM<0.2 Number of generations	Real time (average 20 runs)	Population size
3 - input	<150	0.4	0.6	< 1 minutes	<300	< 2 minute	50
4 - inputs	<350	0.6	0.4	< 2 minutes	<900	< 3 minute	50

Table 2: Results of benchmark tests for assembled circuits.

The 3- and 4- input circuit search was made under similar conditions as the first part of experiments. Results from both tables shows that GA can be quite successfully used to synthesize circuits. Using this algorithm, we were not able to find less expensive quantum realizations of any 3*3 permutation gates than Smolin-like realizations, well-known solutions and our hand designs, but we found the same or new gates of the same cost (the optimized version of the "send" circuit found by Williams, Toffoli and Margolus gates). For instance, only 99 individuals were created in the genetic algorithm pool to find the optimal solution to Margolus gate. Many new gates, some interesting and of small costs, have been found as a by-product of searching for known gates. These results will be analyzed elsewhere.

8. Conclusion

In contrast to previous published work, we produce a sequence of quantum operations that is practically realizable. Our hierarchical stages can be compared to standard gate library design, generic logic synthesis and technology mapping stages of classical CAD systems, respectively. The designer can be a part of feedback loop in evolutionary program and the search is not for circuits of known specifications, but for any gates with high processing power and small cost for given constraints. In contrast to previous approaches, our methodology allows synthesis of both: small quantum circuits of arbitrary type (gates), and permutation class circuits that are well realizable in particular technology.

We have shown that the hierarchical approach can be used for automated QC synthesis. Our evolutionary program found one new circuit that was earlier came across by Williams [507], three circuits located by Rubinstein [45] and several new circuits. In all cases that we studied the program was faster than the results previously published. In contrast to previous works that concentrated on some particular types of circuits such as teleportation [50] and entanglement [45] our approach is fully general. It is also hierarchical and allows to include humans in the design loop. The algorithm from [23] is applied to permutation circuits such as those used in the famous Grover's Quantum Search Algorithm [11]. Although all benchmarks proved the convergence of our GA and results were better than previous ones, the goal of our approach is not only to benchmark a GA, but mainly to explore various evolutionary and other approaches to reversible and quantum circuit synthesis. We created a benchmark library to be used by other interested QC researchers: www.ece.pdx.edu/~lukacm/q-bench.html

9. References

1. A. Al-Rabadi, L.W. Casperson, M. Perkowski and X. Song, "Multiple-Valued Quantum Logic," *Booklet of 11th Post-Binary Ultra Large Scale Integration (ULSI)' 2002 Workshop,* pp. 35-45, Boston, Massachusetts, 15th May 2002.
2. A. Al-Rabadi, "Novel Methods for Reversible Logic Synthesis and Their Application to Quantum Computing," *Ph. D. Thesis*, Portland State University, Portland, Oregon, USA, October 24, 2002.
3. A. Barenco et al., "Elementary Gates For Quantum Computation", *Phys. Review A* **5**2, 1995, pp. 3457-3467.
4. G. Brassard., S.L. Braunstein, and R. Cleve, "Teleportation as Quantum Computation", *in Proc. 4th Workshop on Physics and Computation*, New England Complex System Institute.
5. D. Deutsch, "Quantum computational networks," *Proc. Roy. Soc. Lond*. A. 425, 1989, pp. 73-90.
6. G.W. Dueck, and D. Maslov, "Reversible Function Synthesis with Minimum Garbage Outputs," *Proc. RM-2003.*
7. K.Dill and M. Perkowski, "The Creation of a Cybernetic (Multi-Strategic Learning) Problem-Solver: Automatically Designed Algorithms for Logic Synthesis and Minimization," *submitted to a journal, 2003.*
8. E. Fredkin and T. Toffoli, "Conservative logic," *Intern. J. Theoretical Physics*, 21, pp. 219-253, 1982.
9. Y. Z. Ge, L. T. Watson, and E. G. Collins. Genetic algorithms for optimization on a quantum computer. In

Unconventional Models of Computation, pp. 218-227.
10. D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning *Addison Wesley*, 1989
11. L.K. Grover, "A Framework for Fast Quantum Mechanical Algorithms," *Proc. STOC*, 1998.
12. T. Hogg et al., "Tools for Quantum Algorithms", http://arxiv.org/abs/quant-ph/9811073, 1998.
13. K. Iwama, Y. Kambayashi, and S. Yamashita, "Transformation Rules for Designing CNOT-based Quantum Circuits," *Proc. DAC 2002,* New Orleans, Louisiana.
14. P. Kerntopf, "Maximally efficient binary and multi-valued reversible gates," *Proc. ULSI Workshop*, Warsaw, Poland, May 2001**,** pp. 55-58.
15. P. Kerntopf, "Synthesis of multipurpose reversible logic gates," *Proceedings of EUROMICRO Symposium on Digital Systems Design*, 2002, pp. 259-266.
16. M. H. A. Khan, and M. Perkowski , "Multi-Output ESOP Synthesis with Cascades of New Reversible Gate Family," *Proc. RM 2003*.
17. M.H.A. Khan, and M. Perkowski, "Multi-Output Galois Field Sum of Products (GFSOP) Synthesis with New Quantum Cascades*," submitted to ISMVL 2003*.
18. A. Khlopotine, M. Perkowski, and P. Kerntopf, "Reversible logic synthesis by gate composition," *Proc. IWLS* 2002. pp. 261 – 266.
19. J. Kim, J-S Lee, and S. Lee, "Implementation of the refined Deutsch-Jozsa algorithm on a three-bit NMR quantum computer*", Physical Review A*, Volume 62, 022312, 2000.
20. J. Kim, J-S. Lee, and S. Lee, "Implementing unitary operators in quantum computation," *Physical Review A,* Volume 62, 032312, 2000.
21. J. Kim, "A Study on Ensemble Quantum Computers," *Ph.D. Thesis*, Korea Advanced Institute of Science and Technology, May 23, 2002.
22. J-S. Lee, Y. Chung, J. Kim, and S. Lee, „A Practical Method of Constructing Quantum Combinational Logic Circuits," arXiv:quant-ph/9911053v1, 12 Nov. 1999.
23. M. Lukac and M. Perkowski, "Evolving Quantum Circuits Using Genetic Algorithm," *Proc.* of *NASA/DOD Workshop on Evolvable Hardware*, Washington, D.C. July2002, pp. 177 -185.
24. M. Lukac, M. Pivtoraiko, A. Mishchenko, and M. Perkowski, "Automated Synthesis of Generalized Reversible Cascades using Genetic Algorithms," *Proc.5th Intern Workshop on Boolean Problems*, Freiberg, Germany, September 19-20, 2002. pp 33-45
25. D. Maslov, and G. W. Dueck, "Garbage in Reversible Designs of Multiple-Output Functions," *Proc. RM-2003*.
26. D.M. Miller, "Spectral and Two-Place Decomposition Techniques in Reversible Logic," *Proc. Midwest Symposium on Circuits and Systems,* on CD-ROM, August 2002
27. D. M. Miller and G.W. Dueck, *"* Spectral Techniques for Reversible Logic Synthesis," *Proc. RM 2003*.
28. A. Mishchenko and M. Perkowski, ``Fast Heuristic Minimization of Exclusive Sums-of-Products," *Proc. RM'2001 Workshop,* August 2001.
29. A. Mishchenko and M. Perkowski, "Logic Synthesis of Reversible Wave Cascades", *Proc. IWLS,* June 2002. pp. 197 – 202.
30. M. Nielsen & I. Chuang, *Quantum Computation and Quantum Informatio*n, Cambridge Univ. Press, September 2000.
31. G. Negovetic, M. Perkowski, M. Lukac, A. Buller, "Evolving quantum circuits and an FPGA-based Quantum Computing Emulator," *Proc. Intern. Workshop on Boolean Problems,* 2002. pp. 15-22.
32. N. Song, and M. Perkowski, "Minimization of Exclusive Sums of Multi-Valued Complex Terms for Logic Cell Arrays," *Proc. ISMVL 98,* Fukuoka, Japan, May 1998.
33. M. Perkowski**,** "The method of solving combinatorial problems in the automatic design of digital systems". *Ph.D. Thesis,* Institute of Automatic Control, Technical University of Warsaw, 1980.
34. M. A. Perkowski, P. Dysko, B. J. Falkowski, "Two Learning Methods for a Tree-Search Combinatorial Optimizer," *Proc. of IEEE International Phoenix Conference on Computers and Communication,* pp. 606 - 613, Scottsdale, Arizona, March 1990.
35. M. Perkowski, J. Liu, and J. Brown, "Quick Software Prototyping: CAD Design of Digital CAD Algorithms," In G. Zobrist, (ed.), "Progress in Computer Aided VLSI Design," Vol. 1., *Ablex Publishing Corp*., pp. 353-401, 1989.
36. M. Perkowski, P. Kerntopf, A. Buller, M. Chrzanowska-Jeske, A. Mishchenko, X. Song, A. Al-Rabadi, L. Jozwiak, A. Coppola, and B. Massey, "Regularity and symmetry as a base for efficient realization of reversible logic circuits," *Proceedings of IWLS 2001*, pp. 90-95, 2001.
37. M. Perkowski, L. Jozwiak, P. Kerntopf, A. Mishchenko, A. Al-Rabadi, A. Coppola, A. Buller, X. Song, M. M. H. A. Khan, S. Yanushkevich, V. Shmerko, and M. Chrzanowska-Jeske, "A general decomposition for reversible logic," *Proceedings of RM 2001*. pp. 119 – 138.
38. M. Perkowski, P. Kerntopf, A. Buller, M. Chrzanowska-Jeske, A. Mishchenko, X. Song, A. Al-Rabadi, L. Jozwiak, A. Coppola, and B. Massey, "Regular realization of symmetric functions using reversible logic,"

Proceedings of EUROMICRO Symposium on Digital Systems Design, 2001**,** pp. 245-252.
39. M. Perkowski, P. Kerntopf, A. Al-Rabadi, and M.H.A. Khan, "Multiple-Valued Quantum Computing. Issues, Open Problems, Solutions," Technical Report, KAIST, December 2002.
40. M. Perkowski, M. Pivtoraiko, M. Lukac, C. Huang, S. Ahmad, et al, "Technology mapping for Quantum CAD", *in preparation.*
41. M. Perkowski, A. Al-Rabadi, P. Kerntopf, "Multiple-Valued Quantum Logic Synthesis", *Proc. 2002 Intern. Symp. on New Paradigms VLSI Computing*, December 12-14, Sendai, Japan, pp. 41-47.
42. M. D. Price, S.S. Somaroo, A.E. Dunlop, T. F. Havel, and D. G. Cory, "Generalized methods for the development of quantum logic gates for an NMR quantum information processor," *Physical Review A*, Vol. 60, Number 4, October 1999, pp. 2777-2780.
43. M.D. Price, S.S. Somaroo, C.H. Tseng, J.C. Core, A.H. Fahmy, T.F. Havel and D.G. Cory, "Construction and Implementation of NMR Quantum Logic Gates for Two Spin Systems," *Journal of Magnetic Resonance,* 140, pp. 371- 378, 1999.
44. S. Rahardja and B.J. Falkowski, "Family of Unified Complex Hadamard Transforms," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing, Vol*. 46, No. 8, pp. 1094-1100, 1999.
45. B.I.P. Rubinstein, "Evolving quantum circuits using genetic programming", *Proc. CEC2001,* pp. 144-151.
46. V.V. Shende, A.K. Prasad, I.L. Markov, J.P. Hayes, "Reversible Logic Circuit Synthesis," *Proc. IWLS,* 2002, pp. 125 – 130.
47. J. Smolin, D. P. DiVincenzo, "Five two-qubit gates are sufficient to implement the quantum Fredkin gate." *Physical Review A*, Vol. 53, no. 4, April 1996, pp. 2855-2856.
48. L. Spector, H. Barnum, H. J. Bernstein, and N. Swamy, "Finding a better-than-classical quantum AND/OR algorithm using genetic programming," In *Proc. 1999 Congress on Evolutionary Computation,* volume 3, pp. 2239-2246, Washington D.C., 6.-9. July 1999, IEEE, Piscataway, NJ.
49. C.P. Williams, A.G. Gray, "Automated Design of Quantum Circuits", *QCQC '98,* Springer-Verlag, pp. 113-125, 1999.
50. C.P. Williams, S.H. Clearwater "Explorations in Quantum Computing", *Springer-Verlag*, N Y. 1998.
51. T. Yabuki and H. Iba. "Genetic algorithms and quantum circuit design, evolving a simpler teleportation circuit," In *Late Breaking Papers at the 2000 Genetic and Evolutionary Comp. Conf.* pp. 421-425, 2000.
52. G. Yang, W.N.N. Hung, X. Song and M. Perkowski, "Majority-Based Reversible Logic Gate," *Proc. RM 2003*.