UDC 004.38:004.43
Original Scientific Paper

An Approach to Source Code Conversion of Classical Programming Languages into Source Code of Quantum Programming Languages

Alan Bojić *alan.bojic@in2.hr*
IN2 Ltd.
Marohniceva 1/1, HR-10000 Zagreb, Croatia

Abstract

Basic principles of quantum computer ensure more computing power than the principles of classical computers. Quantum computers use quantum mechanical effects such as entanglement and superposition to speed up computing processes compared to classical computers. If the quantum computers one day becomes a commercially available product, then it is very important to have a converter that will convert the source code of programming languages for classical computers into source code of programming languages for quantum computers because it will speed up the translation process and ensure that developers of classical computers can develop programs for quantum computer using programming languages for classical computers. In this paper, the basic approach to automatic conversion of the source code for classical computer into a source code of quantum computer is introduced.

Keywords: quantum computer, classical computer, source code, conversion approach

1. Introduction

The field of quantum computing was first introduced by Yuri Manin in his paper "Vychislimoe i nevychislimoe"[18] in 1980 and Richard Feynman in his paper "Simulating physics with computers"[7] from 1982. The quantum computer is a computing device that uses quantum mechanical effects such as superposition and quantum entanglement to make computation faster comparing to the computation logic of classical computer [23]. Quantum superposition is a property of a particle to occupy all its possible quantum states *0* and *1* at the same time [21]. Quantum entanglement is the physical resource which can be measured and it is associated with correlations between separated quantum systems [14]. The Bell states represent four special states of two-qubit quantum systems which are maximally *2* [21].

A quantum computer has significantly more computing power than classical computer. Till today, the best proof of quantum computer's computing power is Shor's algorithm that factorizes integer in polynomial time [26] which is exponentially faster of the fastest algorithm for integer factorization for classical computer. Also, one good example of the quantum computer's computing power is Grover's algorithm that finds a specific element in an) [13], where *N* is the number of elements in an unsorted list.

Most known quantum complexity class is BQP (bounded error quantum polynomial time) which includes problems that can be solved in quantum computer with the maximum error *13.*

Scientists believe that the relations between BQP class and other well known complexity classes are as shown in the figure bellow, but it must be said that none of the shown relationships has been proved yet [20].

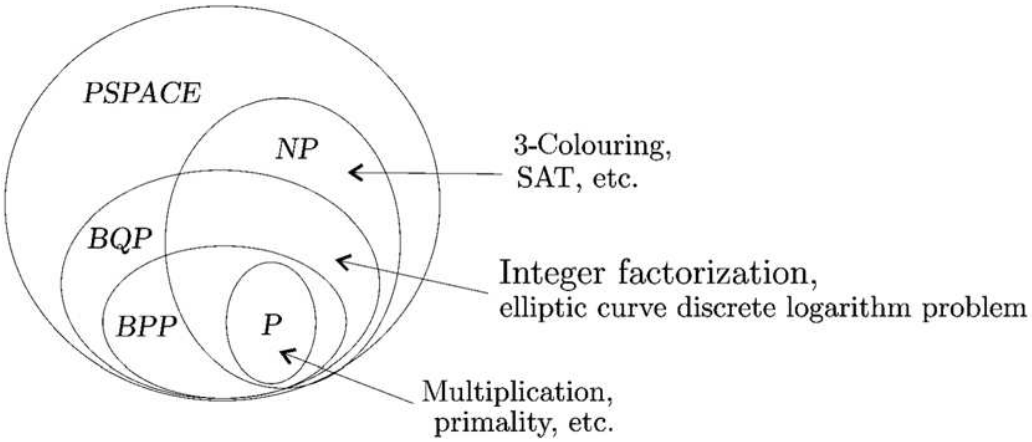

Figure 1. Complexity classes [16]

- P - complexity class which includes problems that can be solved in polynomial time on deterministic Turing machine.
- NP - complexity class which includes problems that can be solved in polynomial time on nondeterministic Turing machine.
- BPP - complexity class which includes problems that can be solved in polynomial *13.*
- PSPACE - complexity class which includes problems that can be solved using polynomial memory space on deterministic Turing machine.

The quantum computer is questioning the Church-Turing thesis which says that any algorithm on a physically achievable computing machine can be simulated on the Turing machine with at most a polynomial slowdown[3], what until today has not been the case with the Shor's algorithm.

One of the major challenges in constructing a big scale quantum computer is quantum decoherence. Quantum decoherence is the loss of coherence or ordering of the phase angles between the components of a system in a quantum superposition and it takes place when the quantum system is not isolated from the system which surrounds it [24]. To date, the biggest quantum computer uses 300 qubits [1], but governments around the world invest their resources in the technology to achieve a large scale quantum computer. NSA already uses a quantum computer for cryptography purposes [2].

All these facts show that quantum computing is a promising computing paradigm and indeed one day quantum computers might be commercially available.

There are few models of quantum computer usage in terms of its collaboration with the classical computer. It is most likely that in the beginning quantum computers will be controlled by classical computers and it will be used for such problems that can be solved much faster by quantum computers than classical computers (i.e. integer factorization). Such model is called QRAM (Quantum Random Access Machine) [20]. One of the disadvantages of that hybrid system is that two technologies must be maintained at the same time but with time the quantum technology will probably prevail because the technology for classical computers is at the edge and parts for classical computers became smaller and smaller, resulting in the circumstance that quantum mechanical effects must be taken into account. We can expect that the technology for quantum computer will continuously develop in future, just like it was the case with the technology for classical computer. If quantum computers will be commercially available, the need for automatic converter of programs of classical computer into programs of quantum computer will arise. Also, one of the consequences of quantum computer expansion is that software developers will have to switch their thinking to quantum logic to program quantum computers. To facilitate that mind switch, converters will ensure that all developers can program quantum computers using classical programming language.

Today, there are many converters which convert source code of one classical programming language into source code of another classical programming language. For instance, Instant C# converts Visual Basic source code into C# source code [15], J2Eif converts Java source code into Eiffel source code [31], C2Eif converts C source code into Eiffel source code [30], Google Web Toolkit converts Java source code into the Javascript source code [11]. Unfortunately, today doesn't exist any published approach for source code conversion between classical programming languages and quantum programming languages so this paper is actually the first paper regarding that subject. In this paper I want to introduce the basic approach for the conversion of source code for a classical computer into source code for a quantum computer. The described approach is a good starting point for research regarding this topic.

The approach for the conversion is described in section 2, whereas section 3 describes a concrete suggestion for implementation, and section 4 provides the conclusion and reference to future research.

2. Proposed approach for source code conversion

The figure below shows the steps according to the proposed approach that must be taken in order to convert the source code of classical programming languages into the source code of quantum programming languages.

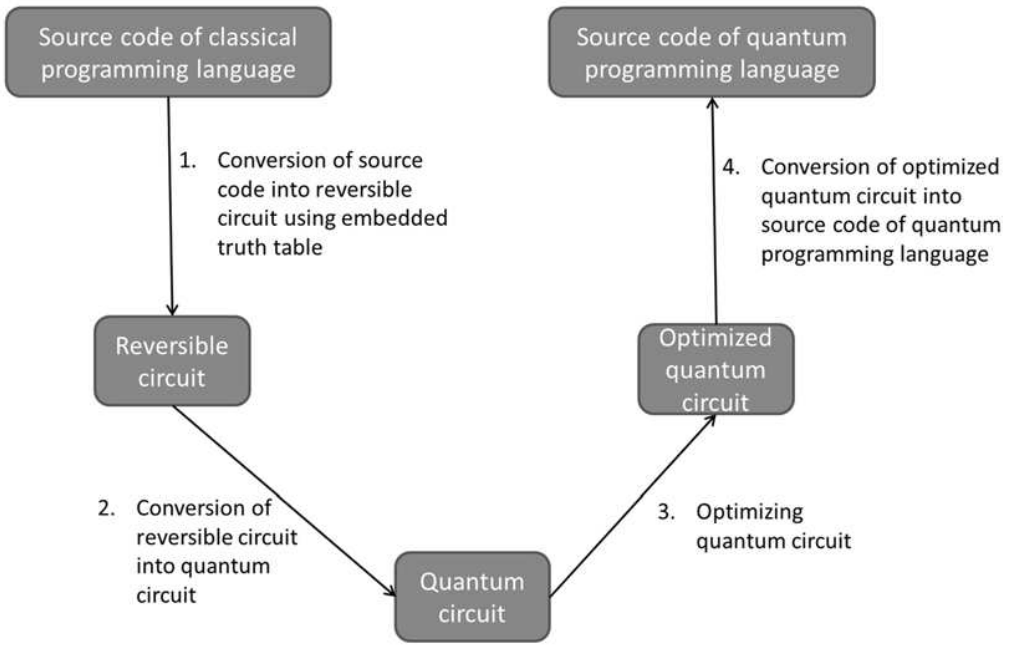

Figure 2. Proposed approach for source code conversion

2.1. Conversion of source code into reversible circuit using embedded truth table

Reversible circuit is a circuit which consists of reversible gates which means that it can be undone. Based on circuit's output we can conclude what was the circuit's input [25]. One example of reversible gate is a NOT gate. On the other side, AND and OR gates are not reversible. To make irreversible circuit reversible the first step is to represent its function by truth table, and then embed it into a reversible truth table. One of such methods for embedding irreversible truth table is well described in the Rolf Drechsler's and Robert Wille's paper "From Truth Tables to Programming Languages: Progress in the Design of Reversible Circuits"[6]. Now we need to somehow convert reversible truth table of irreversible circuit

into reversible circuit which consists of reversible gates. For that we can use transformation based approach introduced by D. M. Miller, D. Maslov and G. W. Dueck in their paper "A transformation based algorithm for reversible logic synthesis" [19].

In this step, the goal is to convert the source code of classical programming language into truth table, embed that table and construct reversible circuit using algorithms mentioned above.

2.2. Conversion of reversible circuit into quantum circuit

All quantum gates are reversible and all reversible gates can be easily represented as quantum operators and implemented on a quantum computer. That is one way of looking at the conversion between reversible circuit and quantum circuit. The other way is to use Toffoli gate. The Toffoli gate is a universal gate in reversible circuit logic [21]. That means that every reversible circuit can be implemented using only Toffoli gates. Because the Toffoli gate can be implemented on a quantum computer we can conclude that we can convert a reversible circuit into quantum circuit by using only Toffoli gates. It is important to emphasize that the Toffoli gate is not a universal quantum gate and that vice versa is not possible in such manner.

Input			Output		
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Table 1. Toffoli gate truth table

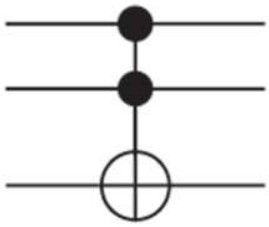

Figure 3. Toffoli gate symbol

2.3. Optimizing quantum circuit

After step two, we will have a quantum circuit, but it will possibly not be optimized in the manner of quantum cost. Quantum cost is the cost of a quantum circuit and it is calculated knowing the number of primitive reversible logic gates (1x1 or 2x2) required to realize the circuit [27].

The goal in this step is to optimize quantum circuit regarding quantum cost. For that purpose we can use various unitary matrix decomposition algorithms such as QR decomposition [8] or CS decomposition [32] which uses the unitary matrix for its input and gives a quantum circuit realizing the input's unitary matrix as its output. There are also other optimization algorithms such as Banerjee's and Pathak's algorithm [5] or Maslov's, Dueck's,

Miller's and Negrevergne's algorithm [19] for minimizing quantum cost which takes unoptimized quantum circuit as its input and gives optimized quantum circuit as its output.

2.4. Conversion of optimized quantum circuit into source code of quantum programming language

The final step is to convert optimized quantum circuit into the source code of programming language for quantum computer. Most of the today's quantum computer languages are on quantum circuit level so in this step only mapping between quantum gates and proper commands should be done.

3. Possible implementation of approach

There are many programming languages for classical computers and several programming languages for quantum computer. The full list of quantum languages and quantum computer simulators can be found on http://www.quantiki.org/wiki/List_of_QC_simulators [17]. In this section the possible implementation of the described approach will be described.

The idea is to convert the source code of VHDL [4] program code of classical computers into the QCL [22] program code of quantum computer. VHDL is suitable for the conversion because its logic is at circuit level, just like in QCL.

The steps of conversion are described in more detail in the following figure.

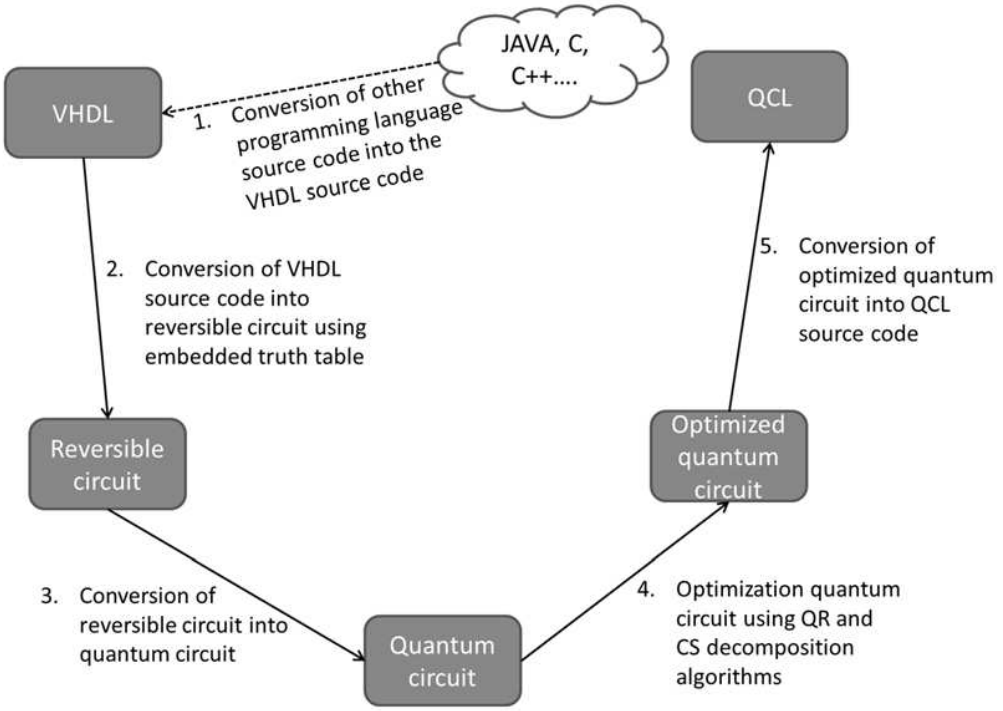

Figure 4. Proposed implementation of source code conversion

3.1. Conversion of other programming language source code into the VHDL source code

For many popular programming languages for classical computer (i.e. Java, C) there is a converter into the VDHL source code. One of such converters is SPARK [29] which converts C code into VHDL. If we take advantage of such existing converters and we successfully implement VHDL -> QCL converter, we will automatically have a converter from other classical computer languages into the QCL.

3.2. Conversion of VHDL source code into reversible circuit using embedded truth table

VHDL stands for VHSIC Hardware Description Language, and it is used to describe digital systems [4][12]. We can easily convert VHDL source code into the truth table using testing benchmarks and simulation tool such as GHDL [10]. As mentioned above, truth table can be embedded into reversible logic and easily converted into the reversible circuit.

The third and fourth points are similar as second and third points in the approach described above. We can use the mentioned algorithms to create a quantum circuit based on truth table and optimize it.

3.3. Conversion of optimized quantum circuit into QCL source code

QCL stands for Quantum computer language. It is one of the most advanced programming languages for Quantum computer. It provides a quantum simulator and utilized syntax derived from C. It is user friendly and it has some of the High level programming features like user defined operators and functions [20][28]. Because QCL has logic on circuit level, we can easily do the mappings between quantum gates in optimized quantum circuit and commands inside the QCL's source code.

4. Conclusion and Future Work

In this paper I described a possible approach for source code conversion of classical programming languages into source code of quantum programming languages. Besides the approach, a proposal for concrete implementation using that approach has also been introduced. The described approach has its limitations and it has room for further improvements. It's obvious limitation is performance issue caused by large truth tables and unitary matrixes which sizes will exponentially grow as number of input and output bits of converted program grows. Improvements regarding that issue can be done, for instance we can improve the implementation of the approach by calculating truth table of circuit by taking into account logic gates which it contains, not using the brute force benchmark method for it or we can recognize design patterns inside the program code of classical computer and use mapped design patterns in quantum computer code for that pattern. We can even expand the implementation of the approach by converting QCL source code into the source code of other Quantum computer programming languages. Such achievement would ensure source code conversion of many classical computer languages into source code of many quantum computer languages. Future work will be focused on the application of a converter using the described approach and on the conclusion as to its limits and how it can be improved. Also, concrete implementation of an approach will give some answers on the efficiency of decomposition algorithms regarding cost criteria.

References

[1] 300 atom quantum simulator smashes qubit record. http://www.zdnet.com/300-atom-quantum-simulator-smashes-qubit-record-4010026044/, downloaded: July, 23rd 2014.

[2] A description of the Penetrating Hard Targets project. http://apps.washingtonpost.com/g/page/world/a-description-of-the-penetrating-hard-targets-project/691/, downloaded: July, 17th 2014

[3] Arora, S.; Barak B. *Computational Complexity: A Modern Approach*, Cambridge University Press, New York, 2009.

[4] Ashenden, P.J.; *The Designer's Guide to VHDL*, Morgan Kaufmann Publishers Inc., San Francisco, 2008

[5] Banerjee, A.; Pathak, A. An algorithm for minimization of quantum cost, quant-ph 0910.2129v2, pages 1-9, 2007

[6] Drechsler, R.; Wille R. From Truth Tables to Programming Languages: Progress in the Design of Reversible Circuits, *Int'l Symp. on Multi-Valued Logic*, pages 78 - 85, 2011

[7] Feynman, R. P. Simulating physics with computers, *International Journal of Theoretical Physics*, Volume 21(6), pages 467–488, 1982.

[8] Gadner, W. Algorithms for the QR-Decomposition, Research report 80-021, Eidgenoessische Technische Hochschule, Zürich, 1980

[9] Garipelly, R.; Kiran, P. M.; Kumar, A. S. A Review on Reversible Logic Gates and their Implementation, *International Journal of Emerging Technology and Advanced Engineering*, Volume 3(3), 2013

[10] Ghdl - Summary. https://gna.org/projects/ghdl/, downloaded: July, 3rd 2014.

[11] Google Web Toolkit. http://www.gwtproject.org/, downloaded: November, 16th 2014.

[12] Goyal, D.; Sharma, V. VHDL Implementation of Reversible Logic Gates, International Journal of Advanced Technology & Engineering Research, Volume 2(3), 2012

[13] Grover, L.K. A fast quantum mechanical algorithm for database search. In *28th Annual ACM Symposium on the Theory of Computing*, pages 212 – 219, Philadelphia, USA, 1996.

[14] Horodecki, R.; Horodecki, P.; Horodecki, M; Horodecki, K. Quantum entanglement. *Rev. Mod. Phys*. Volume 81, pages 865–942, 2009.

[15] Instant C# – VB to C# Converter. http://www.tangiblesoftwaresolutions.com/Product_Details/Instant_CSharp.html, downloaded: November, 16th 2014.

[16] Kaye, P.; Laflamme R.; Mosca M. *An Introduction to Quantum Computing Modern Approach*, Oxford University Press, New York, 2007.

[17] List of QC simulators. http://www.quantiki.org/wiki/List_of_QC_simulators, downloaded: August, 25th 2014.

[18] Manin, Yu. I. Vychislimoe i nevychislimoe , Sov.Radio. pages 13–15, 1980

[19] Miller, D. M.; Maslov, D.; Dueck, G. W. A transformation based algorithm for reversible logic synthesis, *Design Automation Conference*, pages. 318–323, 2003.

[20] Miszczak, J.A. Models of quantum computation and quantum programming languages, *Bulletin of Polish Academy of Sciences: Technical Sciences*, Volume 59 (3), 2011.

[21] Nielsen, Michael A.; Chuang, Isaac L. *Quantum Computation and Quantum Information: 10th Anniversary Edition*, Cambridge University Press, New York, 2011.

[22] QCL - A Programming Language for Quantum Computers. http://tph.tuwien.ac.at/~oemer/qcl.html, downloaded: November, 15th 2014.

[23] Quantum computer - Wikipedia. http://en.wikipedia.org/wiki/Quantum_computer, downloaded: July, 1th 2014.

[24] Quantum decoherence - Wikipedia. http://en.wikipedia.org/wiki/Quantum_decoherence, downloaded: August, 18th 2014.

[25] Saeedi, M.; Markov, I. Synthesis and Optimization of Reversible Circuits - A Survey, *ACM Computing Surveys*, Volume 45(2), 2013.

[26] Shor, P. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM Journal on Computing*, Volume 26(5), pages 1484-1509, 1997.

[27] Smoline J.; DiVincenzo, D.P.; Five two qubit gates are sufficient to implement the quantum fredkin gate, *Physics Review A*, Volume 53 (4), pages 2855 - 2856, 1996.

[28] Sofge, D. A Survey of Quantum Programming Languages: History, Methods, and Tools, Proceedings of the Second International Conference on Quantum, Nano, and Micro Technologies, IEEE Computer Society, pages 66-71, 2008.

[29] SPARK: A Parallelizing Approach to the High-Level Synthesis of Digital Circuits. http://mesl.ucsd.edu/spark/, downloaded: June, 11th 2014.

[30] Trudel M.; Furia C.; Nordio M.; Meyer B.; Oriol M. C to O-O Translation: Beyond the Easy Stuff, *Proceedings of the 19th Working Conference on Reverse Engineering*, 2012

[31] Trudel M.; Oriol M.; Furia C.; Nordio M. Automated Translation of Java Source Code to Eiffel, Objects, Components, Models, Patterns, TOOLS 2011, Zurich, Switzerland, 2011

[32] What is the CS Decomposition, and how does one use it to do quantum compiling? http://www.ar-tiste.com/csd-intro.pdf, downloaded: August, 29th 2014.