Assessment of the IBM-Q quantum computer and its software environment

Zuzanna Chrząstek[1,2], Marian Bubak[1],
Tomasz Stopa[2] and Katarzyna Rycerz[1]

[1]Department of Computer Science, AGH Krakow, Poland
[2]IBM Software Laboratory, Krakow, Poland

zuzanna.chrzastek@ibm.com, {bubak,kzajac}@agh.edu.pl, tomasz.stopa@pl.ibm.com

CGW'18, 22 October 2018

Outline

- Motivation and objectives
- Related work
- Hardware of quantum computers
- IBM-Q devices and software environment
- QASM to the QuIDE converter
- Implementation of a quantum walk algorithm
- Summary and future work

Motivation

- Recently, quantum computing is becoming more and more popular [1].
- IBM has built a quantum computer called IBM-Q, a universal quantum computing system for business and science [2].
- There are other companies that build their own quantum computers, e.g. Google, D-Wave Systems, Rigetti Computing.
- It is impossible to transfer the style of programming from classical computers to their quantum counterparts [3] and therefore the role of quantum computers simulators rises.

[1] Jop Briët, Simon Perdrix. “Quantum Computation and Information”. ERCIM News 112 (Jan. 2018), pp. 8, 9.
[2] “IBM Q Experience Library”. http://research.ibm.com/ibm-q/qx/.
[3] Andris Ambainis, Harry Buhrman, Elham Kashefi, Adrian Kent, Iordanis Kerenidis, Frederik Kerling, Noah Linden, Ashley Montanaro, Floor van de Pavert and Thomas Strohm “Quantum Software Manifesto” (2017) http://www.qusoft.org/quantumsoftware-manifesto/.

Objectives

- Present physical background of quantum computing in IBM-Q.
- Gather information about architecture of IBM-Q.
- Analyze software environment of IBM-Q.
- Propose extensions to the IBM-Q software environment.
- Validate proposed solutions with different quantum algorithms.
- Assess the IBM-Q with a quantum walk algorithm.

Methods

- Surveys for obtaining background information and for assessment of quantum computers and simulators.
- Case studies for analysis of quantum algorithms execution.
- Computer experiments for validation and evaluation of usability.

Related work

Basic papers presenting physical background are:

- *John Clarke and Frank K.Wilhelm*. "**Superconducting quantum bits**". Nature 453 (2008), pp. 1031–1042
  Hans Mooij. "**Superconducting quantum bits**". Physics World 17.12 (2004), p. 29.

Solutions related to building software environments are presented in:

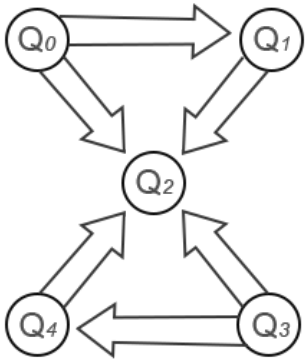
- *Thomas Häner, Damian S Steiger, Krysta Svore, and Matthias Troyer*. "**A software methodology for compiling quantum programs**". Quantum Science and Technology 3.2 (2018), p. 020501
  Joanna Patrzyk, Bartłomiej Patrzyk, Katarzyna Rycerz, and Marian Bubak. "**Towards A Novel Environment for Simulation of Quantum Computing**". Computer Science 16.1 (2015), p. 103.

IBM-Q requires an enhancement with an advanced quantum simulator, that would show inner quantum states during execution of a quantum algorithm. This would help debugging quantum programs.

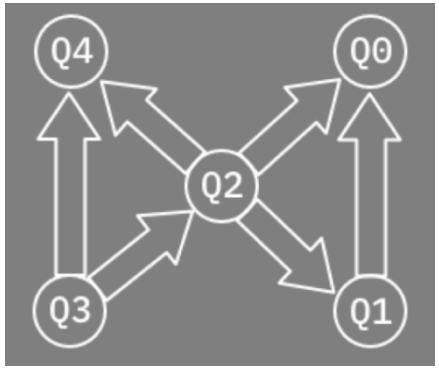
IBM-Q architectures

IBM-Q platform contains three types of backend architecture.

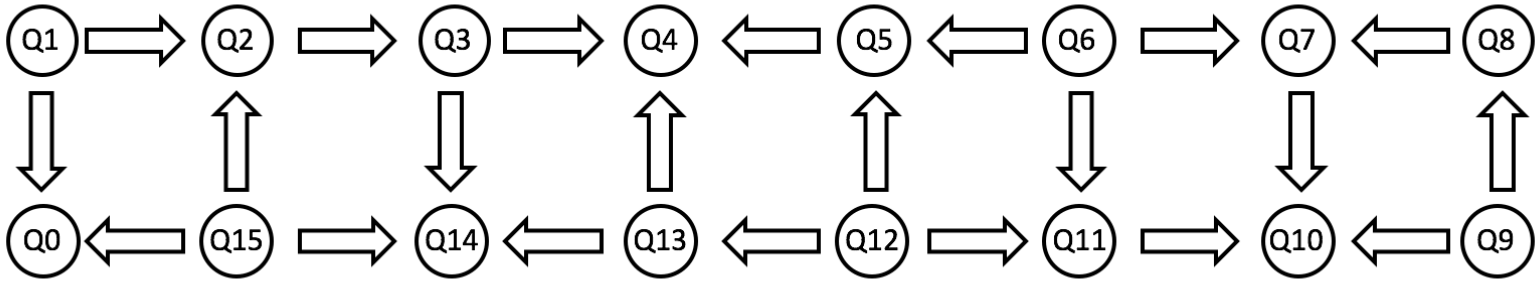
Circles represent qubits while arrows show which qubit in a pair is a control qubit and which is a target qubit.



IBMQX2



IBMQX4



IBMQX5

Basic parameters of IBM-Q

Parameter	General value	IBM-Q's value
Coherence time T_{coh}	100 μs	90 μs
Gate length T_G	10-100 ns	10 ns
Number of quantum operations	$\frac{T_{coh}}{T_G}$	About 10^4
$E_1 - E_0$	4-6 GHz	5 GHz
$E_2 - E_1$	About 5% different than $E_1 - E_0$	

Parameter	IBM-Q's value
Temperature of operation	15-20 mK
Material	Aluminium
Critical temperature of aluminium	1.2 K
Energy of Cooper pair for aluminum	5 GHz
Dimensions of the device simulating one qubit	About 500μm

The **quantum volume V** measures the useful amount of quantum computing done by a device in space and time:

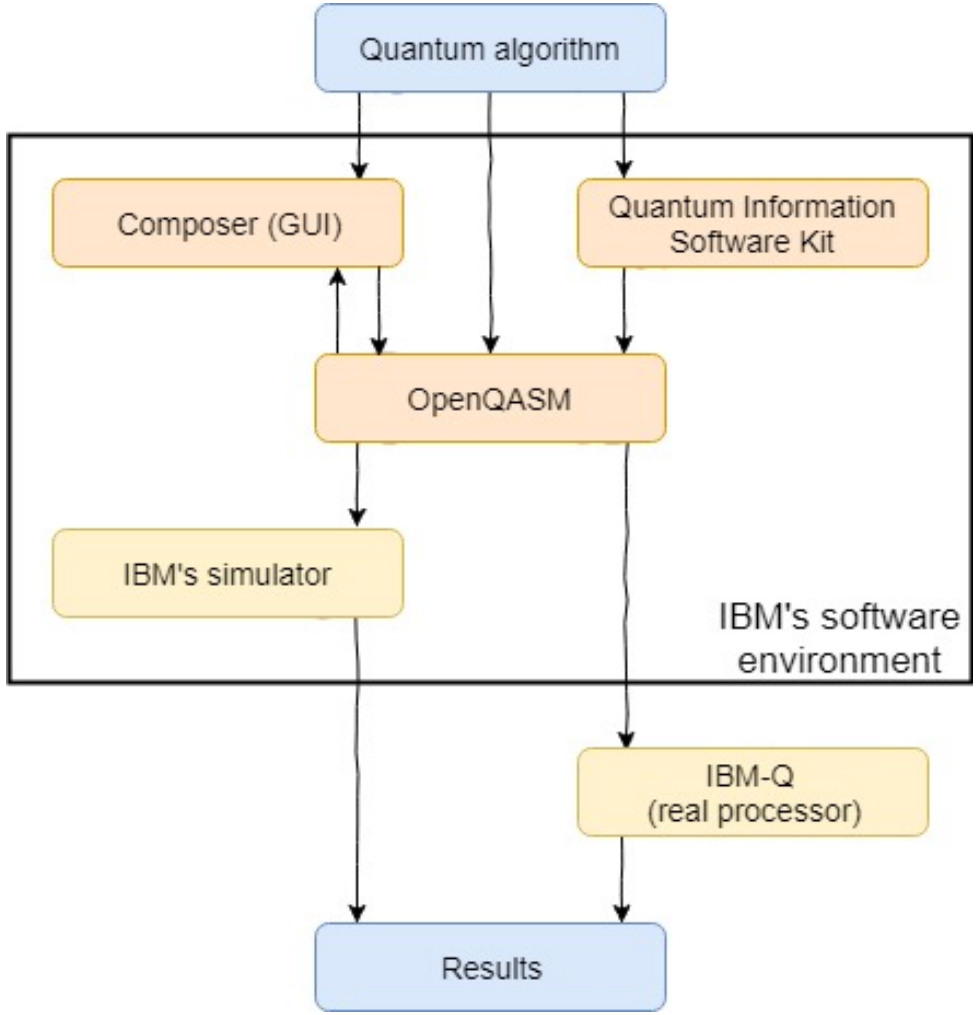
$$V = \max_{n\prime \leq n} \min \left[n', \frac{1}{n' \epsilon_{eff}(n')} \right]^2$$

Topology	Error rate	Quantum volume
IBM QX 5Q	$5x10^{-2}$	16
4x4	10^{-2}	36
4x4	10^{-3}	256
7x7	10^{-3}	256
7x7	10^{-4}	1296

Software environment of IBM-Q

There are several ways to create a quantum program to be executed on IBM quantum computer:

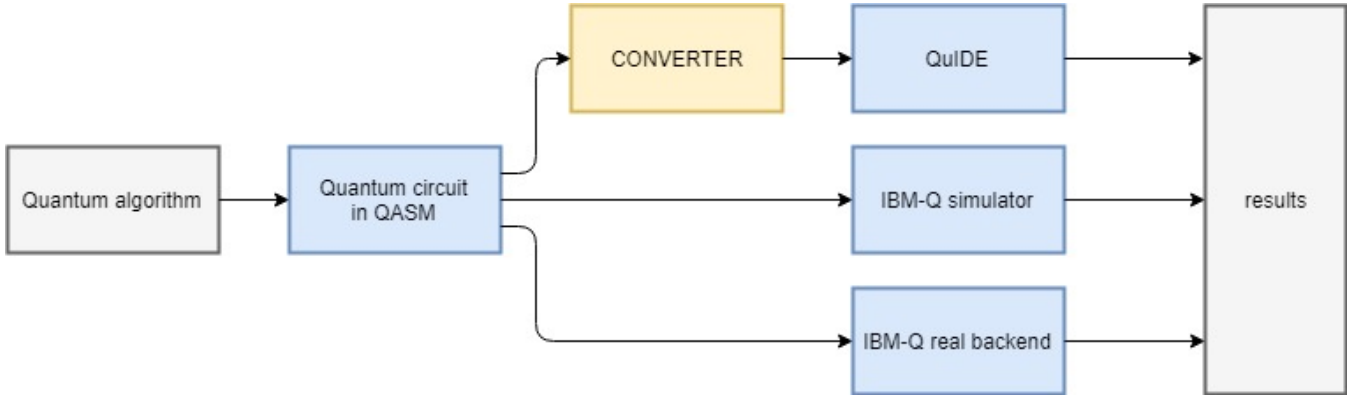
- graphical user interface (GUI),
- software development kit called Quantum Information Software Kit (QISKit),
- both circuit elaborated with the GUI and QISKit code can be translated into QASM code,
- directly in QASM.



Comparison of IBM-Q and QuIDE simulators

QuIDE	IBM-Q GUI and QASM	IBM-Q QISKIT
Non-elementary quantum gates	Only elementary quantum gates	Non-elementary quantum gates
C# (.NET Framework)	Graphical interface and assembler	Python
Ability to create quantum programs with code (C# - QuIDE library) as well as with Circuit Designer	Ability to create quantum programs with code (QASM or Python QISKit framework) as well as with Composer	
Shows values of registers, probability and amplitude after each step of circuit execution	Shows results after execution of the whole circuit	
One library	Multiple frameworks	
Transformation from code to circuit and vice versa		
Ability to create reusable subroutines		

QASM to QuIDE converter

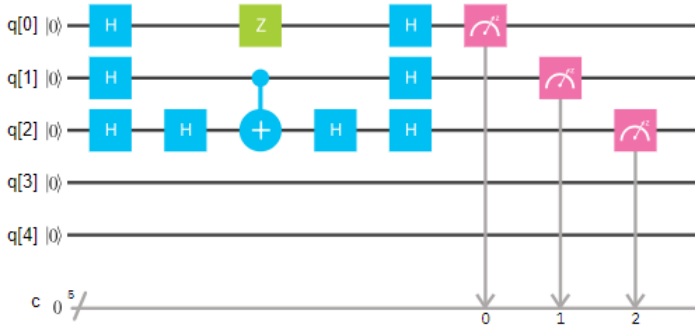


- The converter converts QASM code (transcript of a quantum circuit) to C# code that can be run on QuIDE simulator.
- QuIDE gives the user a possibility to preview the internal state of a quantum system at each stage of the computation. This is impossible on real quantum systems as well as on IBM's simulator.

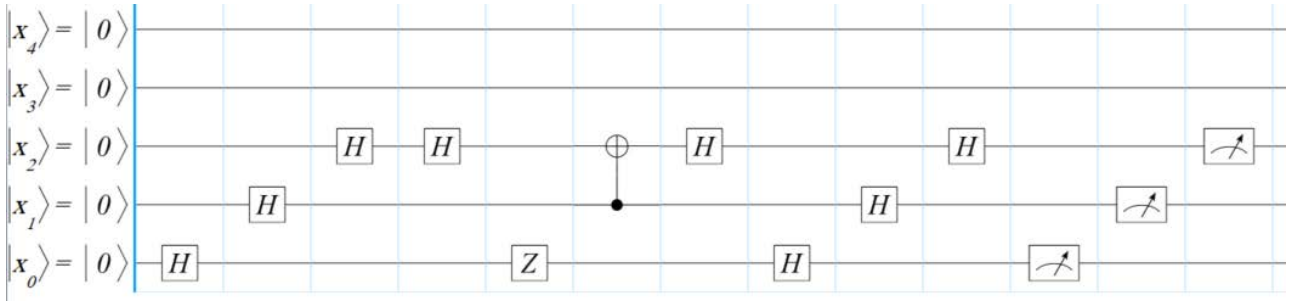
Validation of QASM to QuIDE converter

The converter was validated with the Deutsch-Jozsa algorithm, the Grover algorithm and the Shor algorithm in the following way:

- The existing circuits were transformed in the IBM-Q into a QASM code which was then converted into a C# code in the converter.
- From the C# code a circuit in the QuIDE was generated.
- We compared the circuit from the IBM-Q and the circuit from the QuIDE. If they were identical, it was a proof that the conversion is correct.
- The next step was to check internal states in the key stages of the examined algorithm.
- Finally, we compared results from both simulators as well as from the real IBM-Q processor.



Deutsch-Jozsa quantum circuit in IBM-Q Composer



Deutsch-Jozsa quantum circuit in QuIDE Circuit Designer

Quantum random walk algorithms (1/4)

- Stochastic process that describes a path derived from a series of random steps on some mathematical space [1].
- A walker is placed at 0 on a line of integers and a coin is flipped. Depending on which side it lands, the walker moves one unit to the right or one unit to the left.

$$|\psi\rangle = \sum_k [a_k|0\rangle_c + b_k|1\rangle_c]|k\rangle_p,$$

where

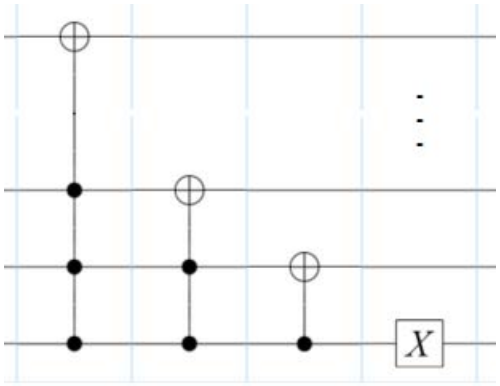
$|0\rangle_c, |1\rangle_c$ are the coin state components,

$|k\rangle_p$ are the walker state components.

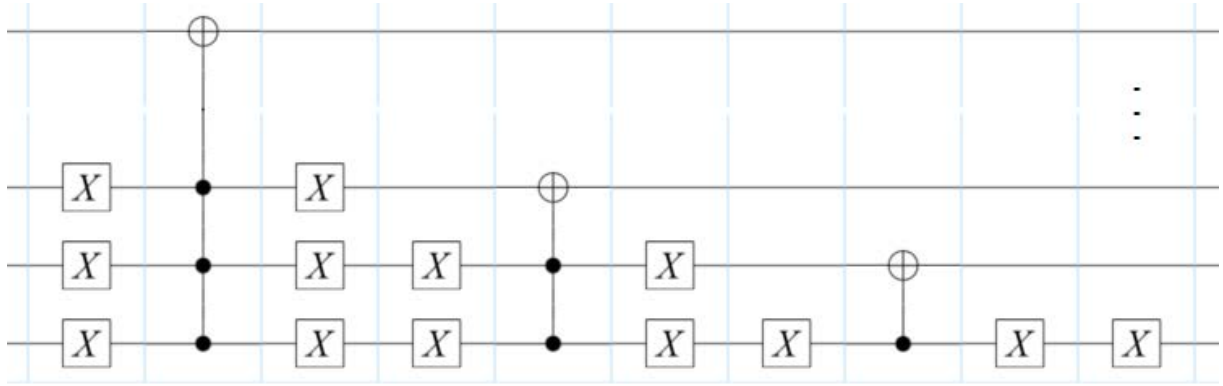
[1] Radhakrishnan Balu, Daniel Castillo, and George Siopsis. “Physical realization of topological quantum walks on IBM-Q and beyond”. Quantum Science and Technology 3.3, 2018, p. 035001

Quantum random walk algorithms (2/4)

Basic gates



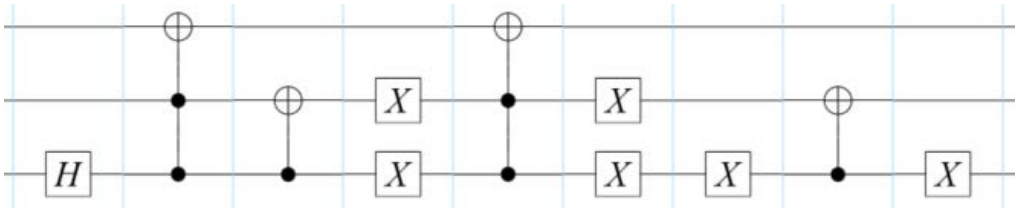
Increment gate



Decrement gate

Quantum random walk algorithms (3/4)

- Implementing this algorithm in QISKit was not trivial, because the cccNOT gate had to be decomposed to elementary gates that QISKit would be able to read.



Quantum walk on three qubits, with a 2-qubit walker.

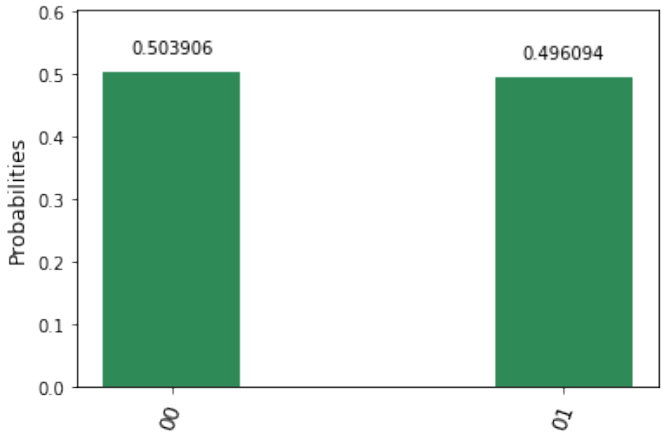
Parameters of U3 gates are $\frac{\pi}{4}, -\frac{\pi}{2}, \frac{\pi}{2}$ converted to double.

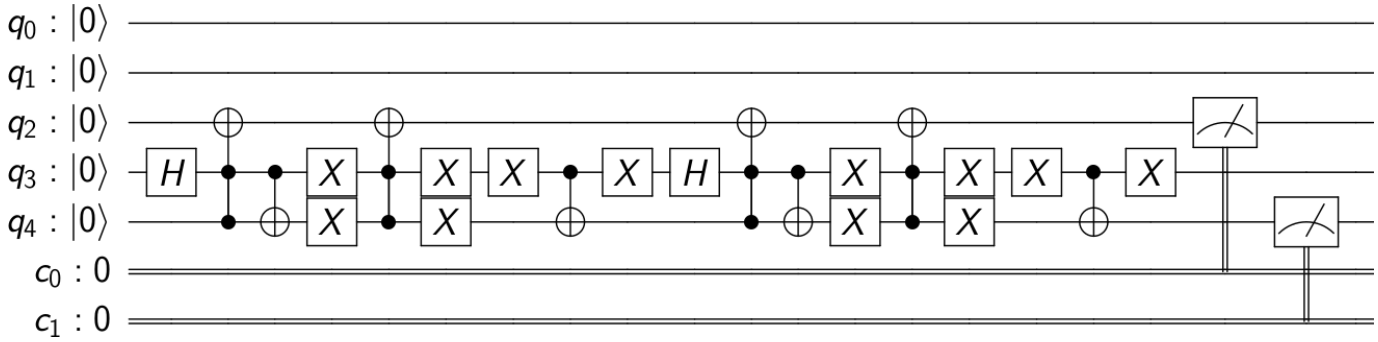
One step of a quantum walk with a 3-qubit walker, implemented in the IBM-Q simulator.

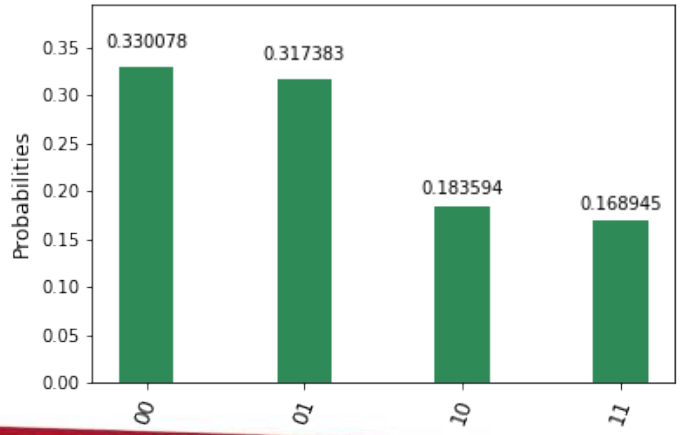
Quantum random walk algorithms (4/4)

- Implementation on both the **IBM-Q simulator** and the **QuIDE simulator** as well as on the **IBM-Q real backend**.
- Results from the IBM-Q simulator are very similar to the results from the QuIDE.
- The results from the IBM-Q simulator and the real backend differ significantly. The reason for that is decoherence.

Value	Qubits	Probability	Amplitude
|0>	|000>	P = 0.25	0.50 + 0.00i
|1>	|001>	P = 0.25	-0.50 + 0.00i
|4>	|100>	P = 0.25	0.50 + 0.00i
|5>	|101>	P = 0.25	0.50 + 0.00i







Assessment of usability of IBM-Q

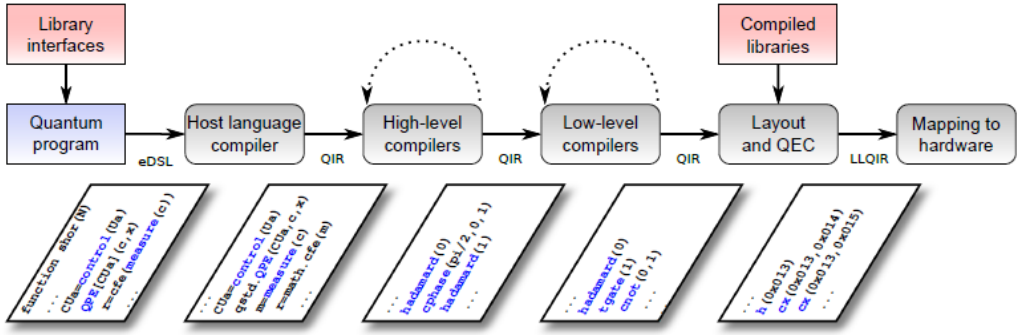
- After a number of gates **decoherence** significantly interrupts executing of algorithms.
- **Lack of tools** that can help with mapping quantum algorithms to topologies of the IBM-Q backends.
- The **graphical interface** is quite limited.
- QISKit provides **more gates** than the graphical interface, nevertheless we needed cccNOT gate which is not available in QISKit.
- In QISKit we can also use **standard Python instructions** such as loops, which significantly speeds up the process of implementation.
- QISKIT provides tools for creating histograms. For debugging purposes, it is useful to visualize the quantum state.

Summary

- We presented **realization of qubits** in superconducting quantum computing.
- **Architectures** of the IBM-Q and its parameters were delineated.
- We described **software environment of the IBM-Q**, its features and their applications.
- We presented our idea for an **extension of the IBM's simulator**.
- We implemented and executed **a quantum walk** on both the IBM-Q and the QuIDE simulators as well as on the real quantum processor of IBM.
- The thesis showed the **importance of quantum simulators**, validated the IBM-Q quantum computer and reviewed some available quantum algorithms.

Future work

- **Decoherence analysis**, especially its impact on the state of the system during usage of multiqubit gates.
- Investigation of optimal placement of selected quantum algorithms on the **IBM-Q architectures**.
- Paper [1] presents a concept of a software architecture for transformation quantum programs from a high-level language program to hardware-specific instructions. This may be considered as a program for further investigations.



[1] Thomas Häner, Damian S Steiger, Krysta Svore, and Matthias Troyer. “A software methodology for compiling quantum programs”. Quantum Science and Technology 3.2 (2018), p. 020501

Final remarks

- This research has been done in collaboration of the Department of Computer Science AGH with the IBM Lab – Krakow

- We acknowledge Dr Katarzyna Rycerz and Prof. Piotr Gawron (IITiS PAN Gliwice) for their suggestions and discussions

- The thesis and this presentation are available at http://dice.cyfronet.pl/publications/filters/filter_MSc_Theses

- We plan to present the results of this Thesis at the CGW'2018 as two short papers:
  - *Interoperability of the IBM-Q and QuIDE simulators*
  - *Implementing and running a quantum walk on the IBM-Q*