Development of a “Quantum Parallel For Loop” Algorithm For Performing Iterated Parallel Quantum Calculations

Pronin Cesar Borisovich

Ostroukh Andrey Vladimirovich

MOSCOW AUTOMOBILE AND ROAD CONSTRUCTION STATE TECHNICAL UNIVERSITY (MADI). 64, Leningradsky prospect, Moscow, Russia

Abstract: This article highlights the development of a “Quantum Parallel For Loop” algorithm for performing iterated parallel quantum calculations. The developed algorithm’s potential real life applications and potential efficiency were analyzed and compared to its classic counterpart.

Key words: quantum for, quantum parallel for, quantum loop, quantum cycle, Grover’s algorithm, oracle function, quantum computing, qubit, superposition, quantum gate.

Introduction

Loops play an irreplaceable role in programming, repeating a certain set of operations until a certain condition is met. Iterations of a loop can be split between different compute units in order to perform them in parallel, significantly speeding up calculations. Based on the optical operating principles of universal quantum computers we proposed that in a quantum computer, controlled operations would perform in parallel, if triggered by simultaneously existing control signals (fig. 1).

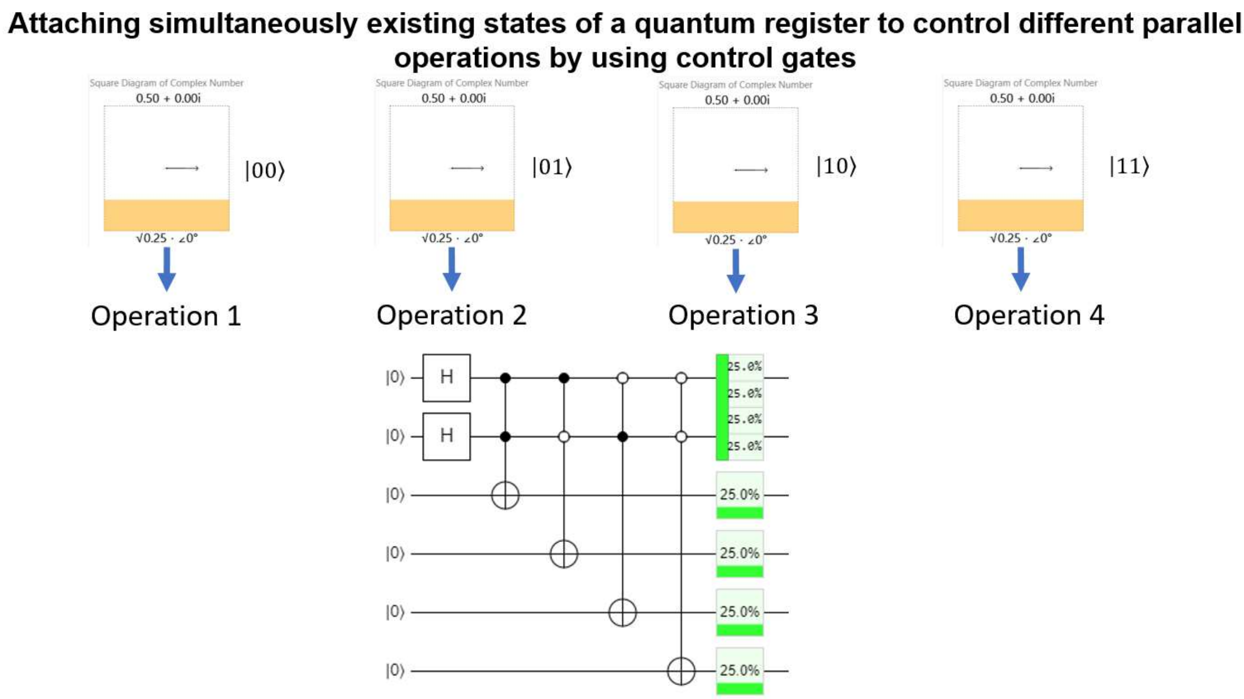

Fig. 1 Attaching states of a quantum register to control different parallel operations by using control gates

The circuit on fig. 2 is based on the Grover's algorithm with a custom oracle function. Let's imagine that we have a task of analyzing a graph. Each hypothetical path of our graph has its index called *"Path"* inside this implementation of Grover's algorithm. If a condition (C1-C3) triggers a control attached to a path's index $(1_{10}, 2_{10},\ 4_{10})$, then the amplitude of that index will be amplified by the Grover's algorithm. The path's index is coded by using different combinations of control elements for the Pauli-Z gate in the oracle function on the top three qubits in the circuit.

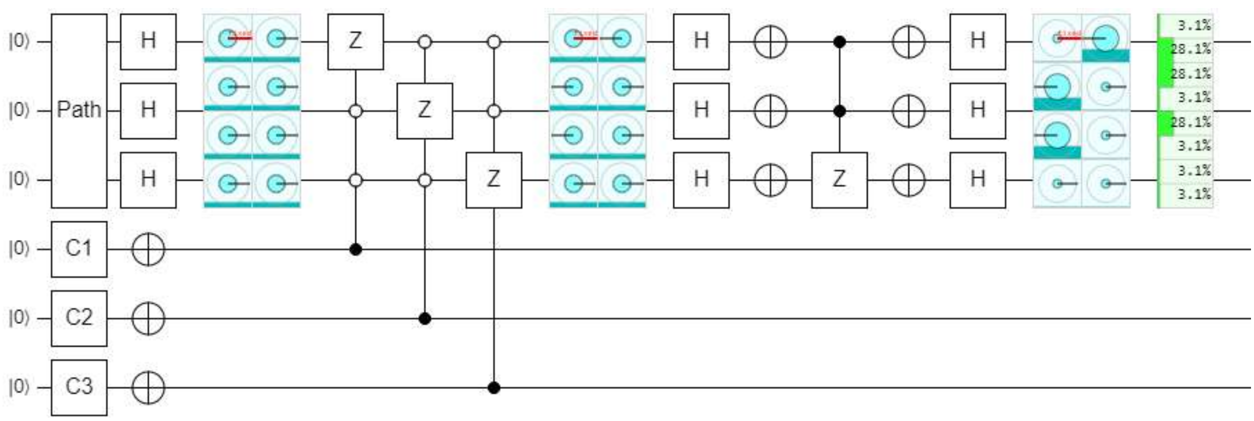

Fig. 2 "Quantum parallel for loop" index triggering concept circuit

This pattern for coding and triggering indexes (fig. 2) was used to develop a quantum version of a *parallel “for” loop* (fig. 3), based on the Grover’s algorithm and the principles shown above.

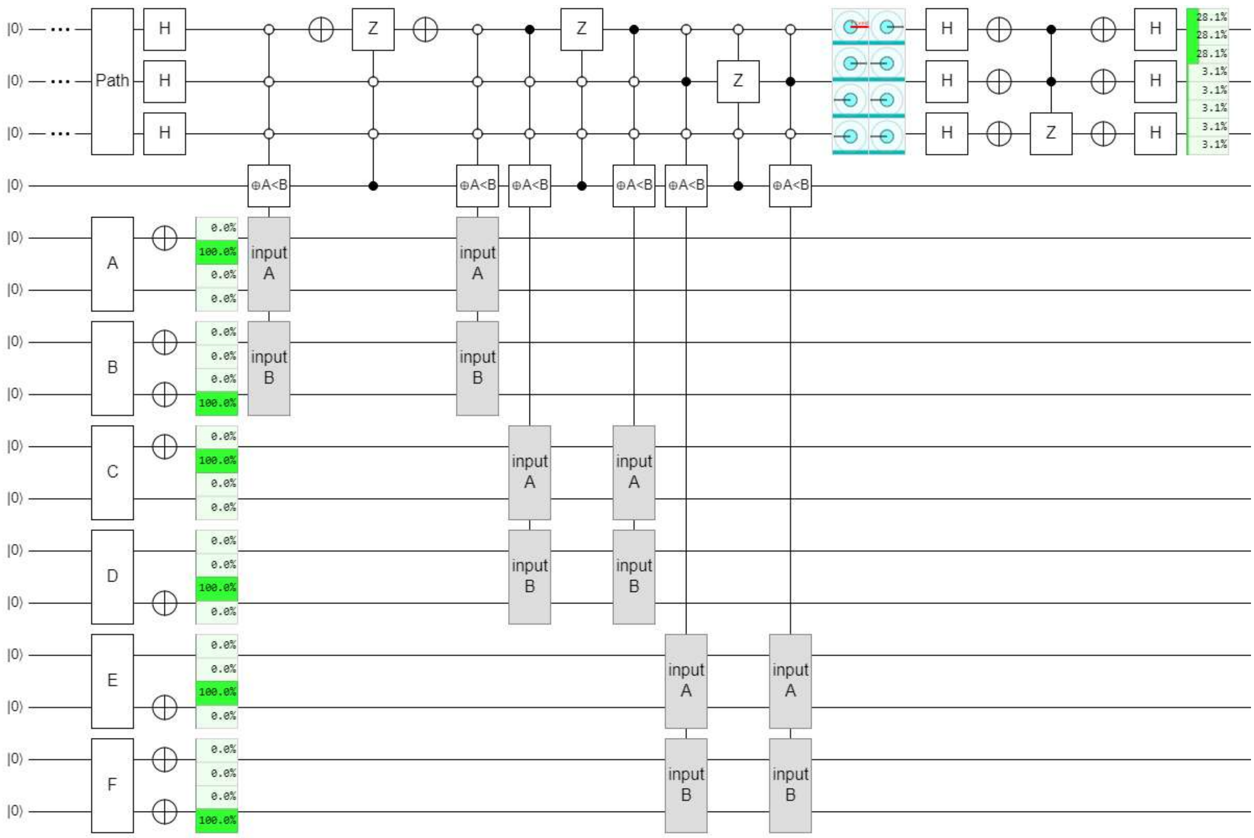

Fig. 3 “Quantum parallel for loop” circuit concept for comparing numbers in pairs

The circuit on figure 3 performs a parallel search for the minimal number in pairs of six numbers: A and B, C and D, E and F. It is given that indexes 0, 1, 2 are attached to the conditions, which detect - if numbers A, C and E are the smallest in their pairs respectively, in these cases, amplitudes of indexes (states) 0, 1, 2 will be selected and amplified by the Grover’s algorithm.

In case if the algorithm did not affect indexes 0, 1 and/or 2 then it means that, the smallest or equal numbers in their respective pairs are B, D and/or F.

Using the proposed "Quantum Parallel For Loop" algorithm to speed up the process of finding the shortest route in a graph

Dijkstra's algorithm is a classical algorithm for finding the shortest route between nodes in a graph, which was invented by a Dutch scientist Edsger Wybe Dijkstra in 1959. The algorithm is based on the principle of sequentially finding the shortest paths from one (source) node to every other and only works for graphs with non-negative paths.

Big logistic companies use algorithms for finding optimal routes that fit a certain search criteria. These analyses use sufficient computing resources due to operating with large cargo amounts, many transport units and complex delivery maps. Because of that, their optimization is an important and relevant task.

This concept (fig. 4) of attaching controlling states to paths was made as an example for analyzing graph ABCD.

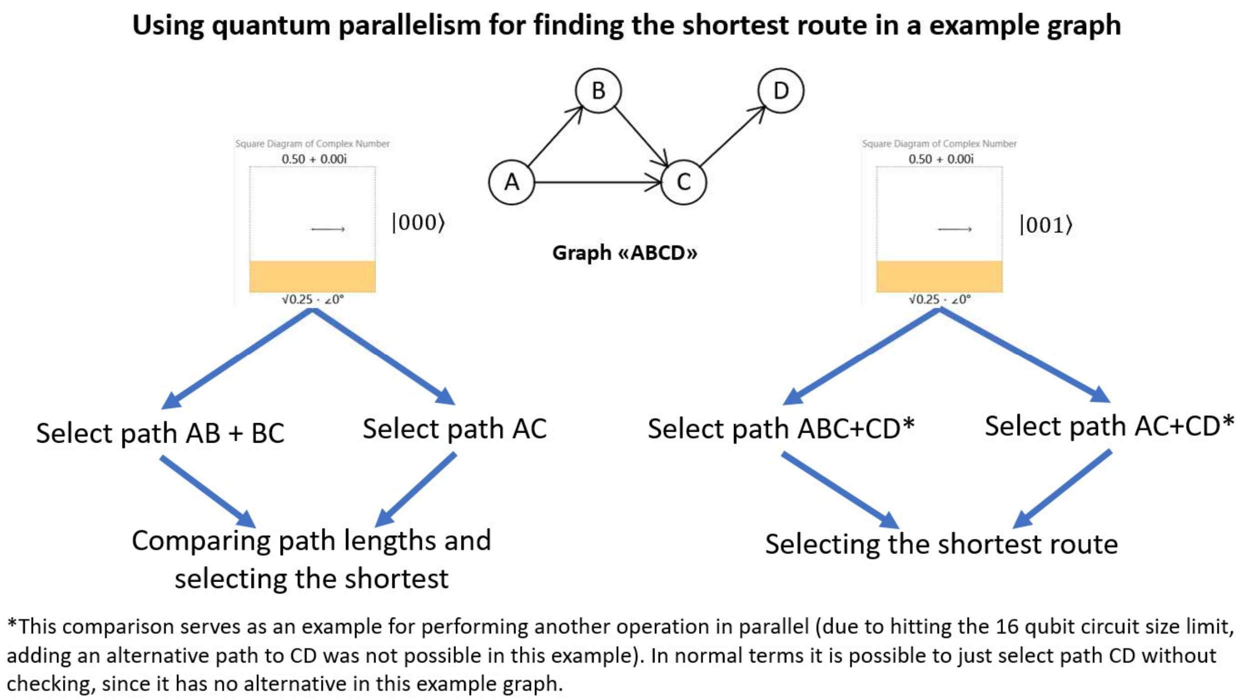

Fig. 4 Concept for attaching path selection to different controlling states in a quantum circuit

The main advantage of this proposed method for performing parallel quantum calculations is that it does not obfuscate (or merge) data-related parts of a quantum circuit and doesn't require compiling complex combined rotation matrices. This keeps the circuit open to necessary changes.

To decrease the size of the circuit, the amplitude amplification step of the Grover's algorithm was compiled into one element called Ampl. This could be described as merging multiple rotation matrices for different operations into one combined rotation matrix (in terms of quantum simulation).

The final quantum circuit for finding the shortest route from A to D is shown on figure 5.

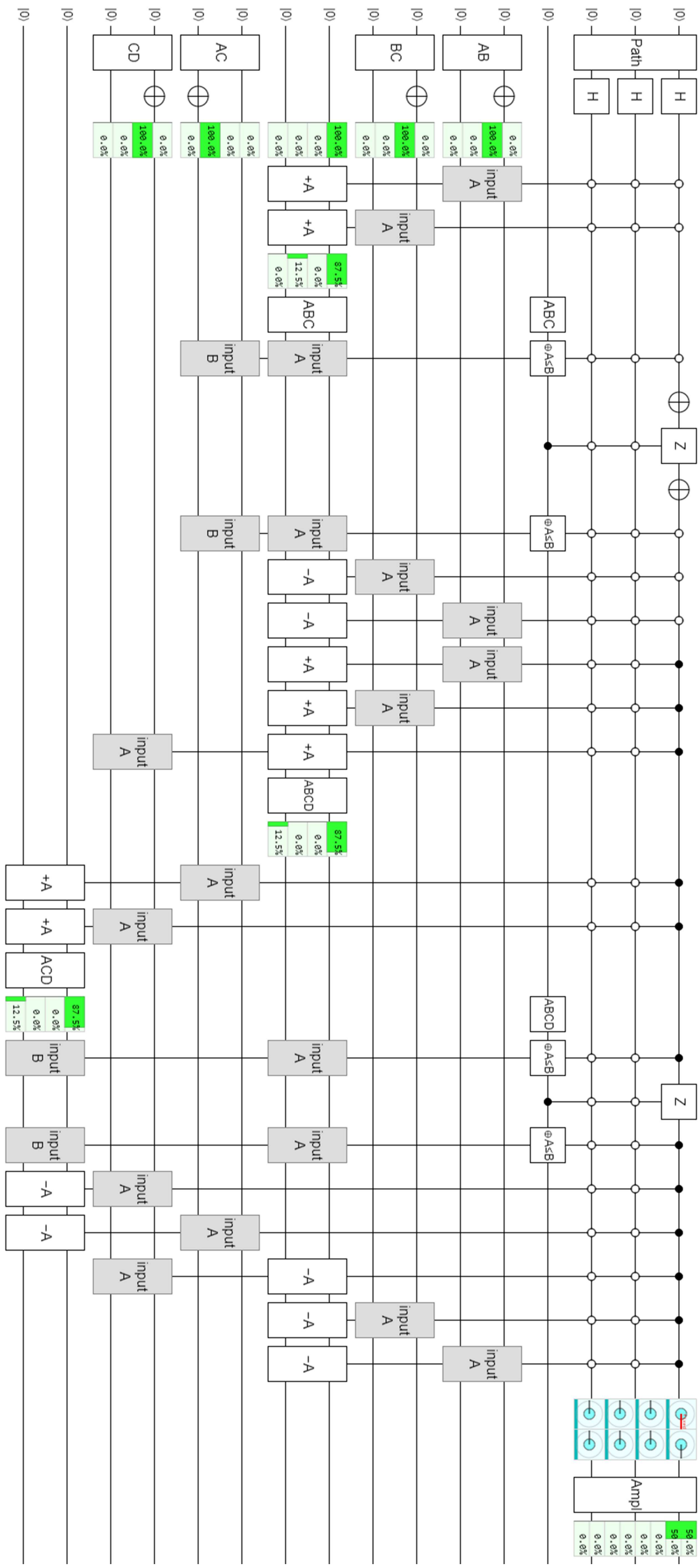

Fig. 5 The proposed circuit for finding the shortest route in the example graph ABCD

Comparing the required “quantum parallel for” iterations with classical linear search

As of this date, universal quantum computers exist mostly in their prototype stage. Because of that, for us, it is only possible to compare the required number of iterations for both sorting algorithms. This is not the only required value for marking the exact difference in performance between different sorting algorithms on different computing architectures. However, the difference in required iterations could give an important estimate for the potential benefit of using quantum computing to solve more complex problems. For comparing all different paths, using Dijkstra’s algorithm the maximum number of linear search iterations N_D is equal to the number of paths N_{Pi} to each of the target nodes N_{tn}.

$$N_D = \sum_{i}^{N_{tn}} N_{Pi}$$

For the example graph “ABCD” on fig. 4:

$$N_{D\ for\ ABCD} = 1 + 2\ + 1 = 4$$

Our version of the shortest route algorithm is based on the Grover’s algorithm, so the number of iterations increases proportional to the number of qubits in its quantum register 2^n and decreases with the number of solutions *l*. In our circuit, the number of solutions *l* is equal to the number of paths taken on the way from A to D.

Because of that, in the best case, only one Grover iteration N_G is need for our example graph (N_G is rounded to its lowest value (floor)):

$$N_G = \left[\!\left[\frac{\pi}{4} * \sqrt{\frac{2^n}{l}} \right]\!\right] = \left[\!\left[\frac{\pi}{4} * \sqrt{\frac{8}{2}} \right]\!\right] = [\![1{,}57]\!] = 1$$

The worst case is 2 iterations, this happens if routes, alternative to the ones reported by the circuit, are selected. In this case *l=0 or l=1* (if *l=0*, then the number of iterations is counted as if *l=1*, but all the output states of the algorithm will have the same probability of remaining after measurement):

$$N_G = \left[\!\left[\frac{\pi}{4} * \sqrt{\frac{2^n}{l}} \right]\!\right] = \left[\!\left[\frac{\pi}{4} * \sqrt{\frac{8}{1}} \right]\!\right] = [\![2{,}22]\!] = 2$$

$$where\ [\![x]\!] - is\ the\ floor\ of\ x$$

With a larger amount of available qubits, the circuit on fig. 5 could be modified to always report all paths on the shortest route, thus making the number *l* maximum possible, further decreasing the number of required Grover iterations.

The difference in the number of required iterations between a linear search algorithm (classic "for" loop) and search algorithms based on the Grover's algorithm is calculated via this formula:

$$N_\Delta = \frac{N}{N_G} = \frac{N}{\frac{\pi}{4} * \sqrt{\frac{N}{l}}} = \frac{4 * \sqrt{N * l}}{\pi}$$

Table 1

Comparing the required number of iterations for analyzing all states of a register by using linear search and Grover's algorithm with l = 1

Number of qu/bits	Number of linear search iterations	Number of Grover iterations	Grover's algorithm benefit (N_Δ)
2	4	1	4
3	8	2	4
8	256	12	21
16	65 536	201	326
32	4 294 967 296	51 471	83 444
64	18 446 744 073 709 551 616	3 373 259 426	5 468 522 205

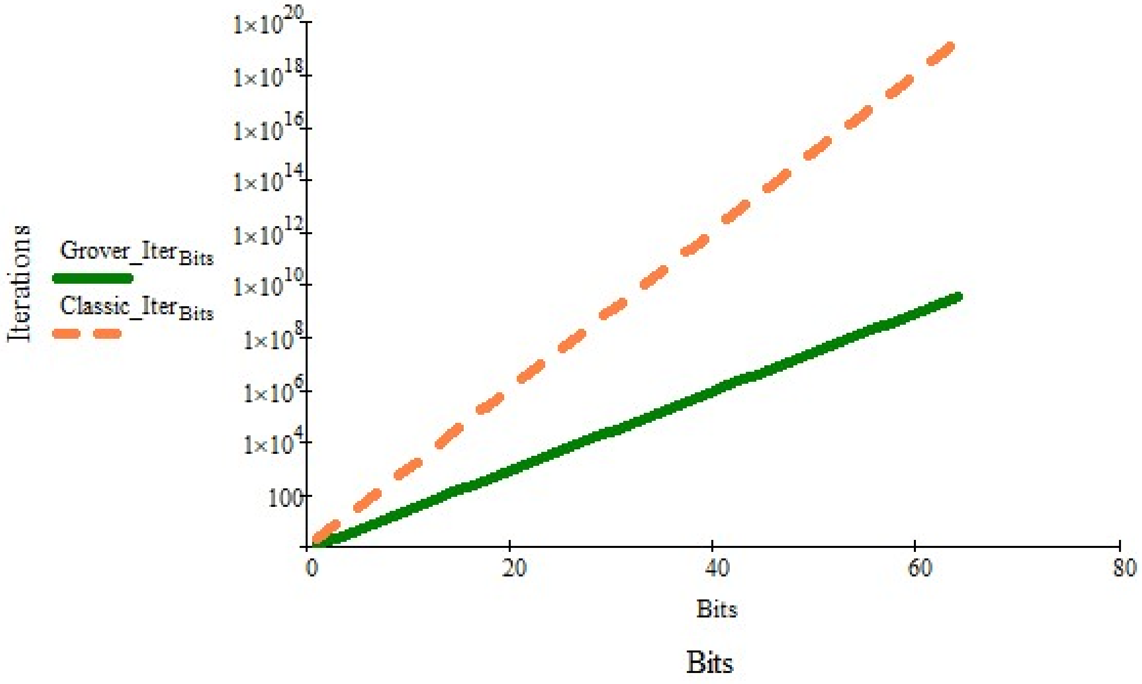

Fig. 6 Logarithmic plot of the difference between required iterations of Grover's algorithm (solid line) and linear search (dotted line) depending on the size of the analyzed register

The comparison shows that using search algorithms based on the Grover's algorithm, such as our proposed "quantum parallel for loop", can greatly reduce the number of required iterations.

In addition, increasing the number of solutions l can further improve performance of the quantum algorithm. For our example on fig. 4-5, this means that with the rising complexity of the graph, the number of iterations required for its analysis decreases.

Conclusion

As the result of this research, a "Quantum Parallel For loop" algorithm was proposed as a method for performing iterated parallel calculations (fig. 2, 3, 5). The proposed pattern was used for performing a parallel search for minimal numbers in their respective pairs and for finding the shortest route in a graph. The proposed algorithm is based on the Grover's algorithm, so the number of required iterations was compared to linear search (classic "for" loop). The comparison showed that the quantum algorithm uses a lot less iterations than its classic counterpart and benefits even more from task complexity. More precise benchmarking of the “Quantum Parallel For loop” algorithm has to be done on a real quantum computer.

Given more qubits, this proposed method for performing iterated parallel calculations could be used for solving more complex and specialized tasks from both the auto-road complex and other spheres of life and economy.

References

1. Pronin C. B., Ostroukh A.V., Pronin B.V., Vasiliev Y.-, Kotliarskiy E.: Development of a Quantum Algorithm Based on Quantum Parallelism for Finding the Shortest Path in a Graph // APRN Journal of Engineering and Applied Sciences, Vol. 14, No. 4, Feb. 2019, ISSN 1819-6608 – Scopus.
2. Cesar B. Pronin, Andrey V. Ostroukh. Researching the Possibilities of Creating Mathematical Oracle Functions for Grover’s Quantum Search Algorithm // viXra.org – open electronic preprint library / Quantum Physics., published on 30.04.2020 URL: https://vixra.org/abs/2004.0704
3. Cesar B. Pronin, Andrey V. Ostroukh. Researching Possible Interpretations and Implementations of Classical Logic Elements and Algorithms in Quantum Circuits // viXra.org – open electronic preprint library / Quantum Physics., published on 31.05.2020, URL: https://vixra.org/abs/2005.0291
4. Grover L.K.: A fast quantum mechanical algorithm for database search // Cornell University Library. URL: https://arxiv.org/abs/quant-ph/9605043
5. Nielsen, Michael A.; Chuang, Isaac. Quantum Computation and Quantum Information: 10th Anniversary Edition. // Cambridge University Press (9 December 2010): p. 97. - 2.3 Application: superdense coding, ISBN 978-1-139-49548-6.
6. Craig Gidney.: Grover’s Quantum Search Algorithm. URL: http://twistedoakstudios.com/blog/Post2644_grovers-quantum-search-algorithm

7. Ostroukh A.V., Pronin C.B., Researching the possibilities of creating mathematical oracle functions for the Grover's quantum search algorithm // Industrial ACS and controllers. – 2018. – № 9. – P. 3-10. (in Russ.)
8. Pronin B.V. Atomic physics and the basics of Quantum physics., Moscow, RSAU-MTAA, 2015, 178 p. ISBN 978-5-9675-1207-0 (in Russ.)
9. Quirk – online quantum computer simulator by Craig Gidney. URL: http://algassert.com/quirk

Author details

Andrey Vladimirovich Ostroukh

Russian Federation, full member RAE, Doctor of Technical Sciences, Professor, Department «Automated Control Systems».

State Technical University – MADI, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru

ostroukh@mail.ru

Cesar Borisovich Pronin

Russian Federation, Master of Engineering sciences, Department «Automated Control Systems».

State Technical University – MADI, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru

caesarpr12@gmail.com