From quantum-walk-assisted optimization to non-backtracking quantum walks

H. Reittu, F. L. Marquezino, J. Kilpi

***Abstract*— A report of first results on the use of quantum walks in real-life problems. First, an experiment on a quantum computer using quantum walk-assisted QAOA for a graph problem was successful. Next, we defined a non-backtracking quantum walk (NBQW) using a quantum walk on a directed graph. NBQW could have potential applications in network analysis analogous to those of classical non-backtracking random walks. Our first experiments on implementing NBQW on real machines show high noise levels and demonstrate need for more sophisticated methods.**

***Keywords*— quantum walks, non-backtracking walk, non-backtracking matrix, graph theory, implementation**

I. Introduction

Quantum walks (QW) have been primarily applied to the design and analysis of spatial search algorithms [6]. For example, Grover's algorithm can be viewed as a quantum walk on a complete graph, and the quantum algorithm for element distinctness corresponds to a search for a marked vertex in a Johnson graph [1]. In this work, we investigate emerging applications of QW beyond search, including problems in optimization and spectral graph clustering [3].

In the context of optimization, we investigate the Quantum Walk Optimization Algorithm (QWOA) — a quantum-walk-assisted variant of the Quantum Approximate Optimization Algorithm (QAOA) [5]. In the standard formulation of QAOA, the mixer Hamiltonian corresponds to the adjacency matrix of a hypercube defined over the solution space, making its action equivalent to a continuous-time QW on the hypercube. However, in many practical problems, the feasible solution space constitutes only a subgraph of the hypercube. To address this, alternative quantum walks restricted to the feasible subgraph have been proposed [5]. In our approach, we employ a quantum walk on a circulant graph as well as a modified version of the staggered quantum walk; see Fig. 1.

In spectral graph clustering, the concept of non-backtracking random walks has led to novel and effective techniques. In a non-backtracking walk, the walker is prohibited from immediately reversing its previous step. This process can be naturally modeled as a standard walk on a larger directed graph, where each node represents a directed edge of the original graph, and edges encode allowed non-backtracking transitions. Consequently, a quantum non-backtracking walk can be formulated as a quantum walk on this expanded directed graph. To this end, we propose leveraging Montanaro's framework [2] for QW on directed graphs, and we also explore more specialized constructions tailored to this setting.

We demonstrated the capability of QWOA on a quantum computer using a small-scale hard problem. We implemented on a real machine Montanaro's definition of QW on directed graphs and applied it to define a new non-backtracking quantum walk (NBQW). The corresponding experiments indicate the need for better implementations using error correction and high-fidelity qubits and gates.

This work is organized as follows. In Section II, we discuss a version of the Quantum Approximate Optimization Algorithm based on quantum walks. In Section III, we discuss the model and implementation of directed quantum walks. In Section IV, we discuss the non-backtracking random walk, and potential ways of defining its quantum counterpart. In Section V, we present our conclusions and final discussions.

II. Quantum-walk-assisted QAOA

Building on the definition of QWOA by Marsh and Wang [5], we implemented a test case on real quantum hardware to explore its practical feasibility. QAOA — or its quantum-walk-assisted variant QWOA — alternates between two unitaries: one driven by the cost function of the optimization problem, and the other defined by a mixer Hamiltonian. In QWOA, the mixer is realized as a continuous-time quantum walk on a graph defined over the solution space. Two real number parameters, the times of evolutions, governing the application of both unitaries are optimized to increase the probability of sampling a desired solution with minimal cost. In both variants, the graph's nodes correspond to bit strings encoding candidate solutions to the problem.

One natural choice for QWOA is to define the quantum walk on a circulant graph constructed over the valid solution space, as such graphs are convenient for efficient circuit implementation because of their known spectral properties. Such a graph is characterized by an adjacency matrix whose columns are circular permutations of each other. For example, the complete graph — with all off-diagonal entries equal to one and zeros on the diagonal — is a circulant graph.

As a sample problem, we consider the minimum vertex cover on a triangle graph, i.e., the complete graph on three vertices K_3; see Fig. 1. A vertex cover is a subset of marked nodes such that every edge in the graph is incident to at least one marked node. A minimum vertex cover is a vertex cover of

H. Reittu, VTT Technical Research Centre of Finland Ltd., e-mail: hannu.reittu@vtt.fi (corresponding author); F. L. Marquezino, Federal University of Rio de Janeiro, Brazil, e-mail: franklin@cos.ufrj.br; J. Kilpi, VTT Technical Research Centre of Finland Ltd., e-mail: jorma.kilpi@vtt.fi. This work was partially supported by Business Finland projects COHQCA and MatterQ, and CNPq/Brazil grants 407296/2021-2 and 306049/2025-2, and HORIZON-CL4-2022-QUANTUM01-SGA Project No. 101113946 OpenSuperQ-Plus100 of the EU Flagship on Quantum Technologies.

smallest possible cardinality. In case of the triangle, a minimal cover is any of the three colorings with two marked nodes. Any coloring with just one vertex marked is not a vertex cover and thus is outside the solution space. There is only one suboptimal cover in which all three nodes are marked. That is why the search space has size 4.

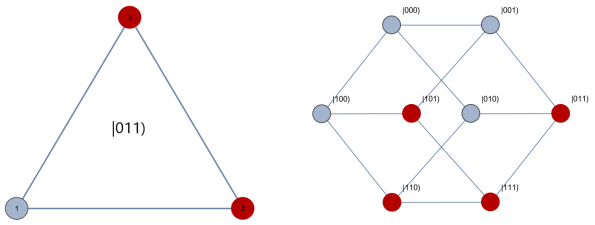

Fig. 1. Left: The graph K_3 with a minimum vertex cover highlighted in red. Right: The corresponding hypercube, or cube in this case, of all possible vertex colorings, where red nodes represent valid vertex covers and gray nodes correspond to non-covers.

We take K_4 as a circulant graph whose nodes correspond to the four vertex covers of the triangle graph. The adjacency matrix of K_4 has a well known spectral structure. Its eigenvectors take the form

$$v_j = \frac{1}{\sqrt{3}}(1, \omega^j, \omega^{2j}, \omega^{3j})^T, \tag{1}$$

in which $\omega = \exp(2\pi i/3)$ and $j = 0, 1, 2, 3$. For $j = 0$, the corresponding eigenvalue is 3, while for all other values of j, the eigenvalues are equal to -1. More generally, for any circulant graph, the eigenvectors form the columns of the discrete Fourier transform matrix, and the eigenvalues are known in a closed form.

In QAOA — or its quantum-walk-assisted variant QWOA — the mixer unitary has the form $U(\tau) = e^{-i\tau A}$, where A is the adjacency matrix of the underlying graph and the real τ is a variational parameter. The problem Hamiltonian is denoted by C, it is a diagonal matrix whose diagonal entries correspond to the cost of each vertex cover. The corresponding unitary is $V(t) = e^{itC}$, with real t as the second variational parameter. For circulant graphs, the spectral decomposition of A allows the mixer unitary to be expressed as

$$U(\tau) = Fe^{-i\tau\Lambda}F^\dagger, \tag{2}$$

where Λ is a diagonal matrix containing the eigenvalues of A, and F is the quantum Fourier transform matrix [5]. Since the eigenvalues of A are known analytically for any circulant graph, this formulation enables efficient construction of quantum circuits in a more general case.

A. Simulation and experimental results: QWOA

After presenting the general formalism of QWOA, we now turn to a concrete application in order to assess its behavior in both noiseless simulations and on real quantum hardware. This requires specifying the optimization task, constructing the corresponding quantum circuit, and comparing the expected performance with the outcomes obtained from actual devices.

The task of QAOA is to minimize the expected cost

$$\mathbb{E}(C) = \langle t, \tau | \, C \, | t, \tau \rangle \, , \tag{3}$$

where $|t, \tau\rangle = V(t)U(\tau)|\psi\rangle$ and $|\psi\rangle$ is the initial quantum state. For the minimum vertex cover problem on the triangle graph, this results in the circuit shown in Figure 2, implemented using standard Qiskit tools.

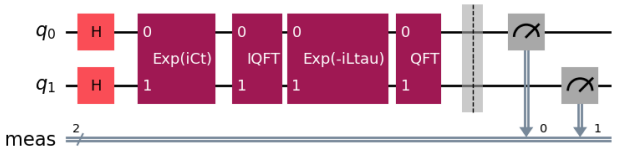

Fig. 2. Quantum circuit for running quantum-walk-assisted QAOA with fixed parameters to solve the minimum vertex cover problem on a triangle graph. The solution space has dimension 4, so two qubits are sufficient. The initial state $|\psi\rangle$ is an equal superposition over all computational basis states, prepared by applying Hadamard gates to each qubit.

In the ideal (noise-free) setting, the optimal parameters (τ, t) can be determined analytically. Under these conditions, the algorithm yields a uniform distribution over the three minimum vertex covers, each appearing with probability $1/3$ (see Fig. 3).

All noiseless simulations were performed using the Qiskit framework, which enables native support for constructing and executing the required quantum circuits under ideal, noise-free conditions. For the experimental implementation, the same circuits were deployed on the VTT Q50 quantum processor—a superconducting device co-developed by VTT and IQM featuring 53 flux-tunable qubits arranged in a lattice topology. Details on the hardware architecture, native gate set, and access mechanisms are available at `https://qx.vtt.fi/docs/devices/q50.html`. The results in Q50 were positive as seen in Fig. 3. We also got similar results using the so-called staggered QW on K_4; details are omitted due to lack of space.

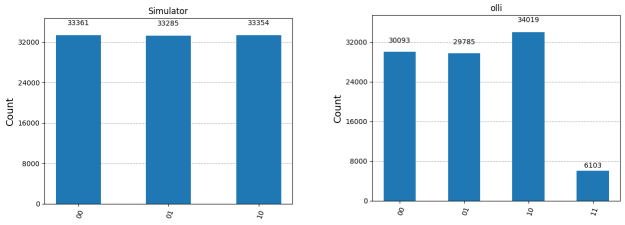

Fig. 3. Results of QWOA applied to the minimum vertex cover problem on the triangle graph K_3. Left: Simulation results in the ideal (noise-free) setting, showing nearly uniform sampling over the three optimal vertex covers. Right: Experimental results on VTT Q50 quantum processor. The distribution remains close to ideal, with the non-optimal solution appearing in approximately 4% of the shots. We obtained similar accuracy using a version of staggered QW on K_4.

III. Quantum walks on directed graphs

Montanaro [2] introduced a coined quantum walk on directed graphs in which simple directed cycles serve as the basis elements. In this construction, the walker's positions

correspond to the vertices of the graph. The shift operator is defined by selecting a collection of disjoint simple directed cycles covering a subset of vertices, while nodes not included in any cycle are treated as self-loops. Each such cycle induces a permutation of the vertices, which naturally defines a unitary operator U_i. To ensure that every directed edge of the graph participates in the dynamics, one considers a family of operators $\{U_i\}_i$ obtained from different cycle decompositions. Coin degrees of freedom are then introduced, assigning to each operator U_i a corresponding coin state that determines when it is applied. This framework, however, cannot be implemented on arbitrary directed graphs, since it may not always be possible to cover all directed edges with disjoint cycles.

For illustration, we implemented Montanaro's construction on the graph shown in Fig. 4. This graph has four vertices and contains two directed 3-cycles, which were chosen to define the dynamics of the walk. Each cycle corresponds to a unitary operator that performs a cyclic permutation of three vertices while leaving the remaining vertex unchanged. A single-qubit Hadamard coin is then used to select between these two shift operators, thereby completing one step of the quantum walk.

A. Simulation and experimental results: QW on a directed graph

To implement coin-controlled unitaries, we employed controlled versions of the two permutation matrices, resulting in a quantum circuit with three qubits, as shown in Fig. 4. In our experiment, the walker was initialized at node $|00\rangle$, which belongs to only one of the directed cycles. Consequently, for the coin state corresponding to that cycle, the walker moves to the neighboring node $|01\rangle$, while for the other coin state it remains at $|00\rangle$. Since the two coin states are prepared with equal amplitudes, the resulting positions are also expected to have equal amplitudes. Simulations confirm this behavior; however, experimental runs on the VTT Q50 quantum computer reveal a substantial effect of noise (see Fig. 4). This suggests that realizing quantum walks on directed graphs will require advanced error suppression and mitigation techniques. The transpiled circuit used here is far from optimal, and in future work we plan to apply tensor-network error mitigation (TEM) methods developed by Algoritmiq [4].

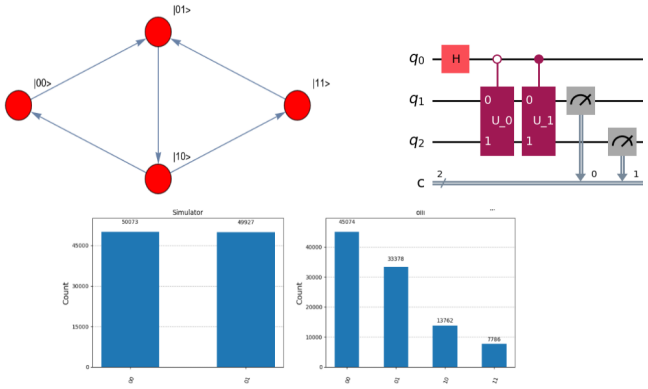

Fig. 4. Montanaro quantum walk on a directed graph with four vertices. Two triangle-shaped directed cycles define two permutations, which in turn specify two shift operators controlled by a single-qubit Hadamard coin. This construction yields the quantum circuit shown for one step of the walk. The corresponding results on a noiseless simulator (left histogram) and on the VTT Q50 quantum processor (right histogram) illustrate the significant impact of noise.

IV. Non-backtracking quantum walks via directed graphs

The classical non-backtracking walk on an undirected graph $G = (V, E)$, with vertex set V and edge set E, differs from the usual random walk by imposing the constraint that the walker cannot immediately return to its previous position. Specifically, if the walker moves from vertex a to vertex b, the following step cannot go back to a. An undirected graph G can be represented as a directed graph in which every edge $\{a, b\} \in E$ is replaced by two directed edges, (a, b) and (b, a). Under this representation, traversing (a, b) forbids (b, a) as the next move.

The classical non-backtracking walk can be conveniently represented using the *non-backtracking matrix* B, a binary square matrix of dimension $2|E|$. Its rows and columns are indexed by the directed edges of G. For a given row labeled by the directed edge (a, b), the nonzero entries specify the edges that may follow (a, b) in a non-backtracking walk. Explicitly,

$$B_{(a,b),(x,y)} = \begin{cases} 1, & \text{iff} \quad (a,b) \in E, (x,y) \in E, x = b, y \neq a \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The classical non-backtracking walk is equivalent to an ordinary walk on the directed graph whose adjacency matrix is B. This correspondence naturally leads to the definition of a non-backtracking quantum walk as a quantum walk on the same directed graph, with B serving as its adjacency matrix. Figure 5 illustrates this construction with a sample graph and its associated non-backtracking graph.

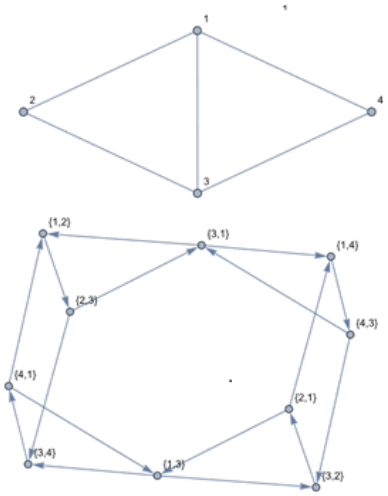

Fig. 5. Top: an undirected graph with four vertices and five edges. Bottom: the corresponding non-backtracking graph, whose adjacency matrix is B. The non-backtracking graph has ten vertices, each representing one of the ten directed edges of the original graph.

As a first example, we consider NBQW on the graph shown in Fig. 5. In this case, the construction requires choosing

between two permutations, illustrated in Fig. 6. Since only two alternatives are involved, a single qubit is sufficient to encode the coin space and select between them.

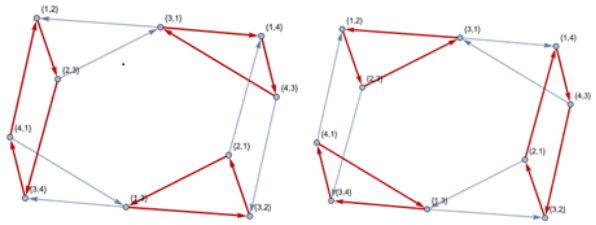

Fig. 6. Two systems of cycles, denoted by red links, cover all edges and can be used to define two permutations which define the shift operator in NBQW.

A. Simulation and experimental results: NBQW

Such a quantum walk can be simulated numerically, and the results are shown in Fig. 7. The figure displays the time-averaged probability of finding the walker at each node of the graph when the initial state is chosen with random complex amplitudes on all nodes. The time-averaged probability is defined as [6]

$$\overline{p}_v(T) = \frac{1}{T}\sum_{t=1}^{T} p_v(t),$$

where $p_v(t)$ denotes the probability of finding the walker at node v after t steps, and T is the total number of steps considered.

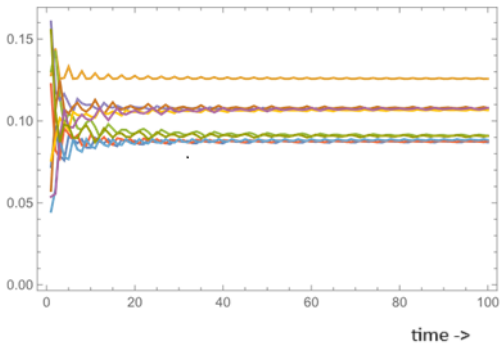

Fig. 7. Time-averaged probability of finding the walker at each node of the graph after 100 steps of the non-backtracking quantum walk.

We also implemented the quantum walk on the VTT Q50 quantum processor. The quantum circuit corresponding to a single step of the walk is shown in Fig. 8. In the absence of error correction, the results are strongly degraded by noise, and even the implementation of a single step becomes practically unfeasible, as illustrated in Fig. 9.

Next, we consider a larger example: a graph with ten vertices and sixteen edges, together with its corresponding non-backtracking graph shown in Fig. 10.

In this case, the edges of the non-backtracking graph can be covered by four Hamiltonian cycles (cycles that visit every vertex exactly once). These cycles serve as the basis for four permutations, which define the shift operators of the quantum walk. The coin space is implemented by a two-qubit Hadamard coin, and the resulting quantum circuit analogous to the one in Fig. 8 but with two coin qubits instead which control 4 unitaries acting on 5 qubits representing 32 positions on the graph.

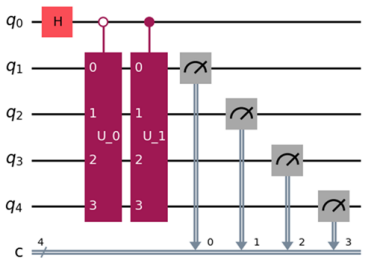

Fig. 8. Quantum circuit for one step of the non-backtracking quantum walk, with controlled unitaries U_1 and U_2 implementing the two permutations illustrated in Fig. 6.

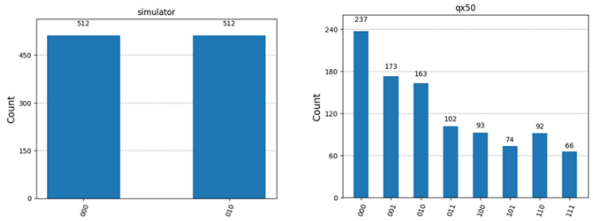

Fig. 9. One step of the non-backtracking quantum walk with a chosen initial condition. The expected outcome is shown in the left histogram, while the experimental results obtained on the VTT Q50 processor (right) exhibit significant noise.

This quantum walk can be simulated numerically. Figure 11 shows a sample result with the time-averaged probability distribution of finding the walker on the vertices of the non-backtracking graph, together with the corresponding distribution projected onto the vertices of the original graph. When implementing this quantum circuit on current hardware, the results are dominated by noise, to the extent that even a single step of the quantum walk cannot be reliably observed.

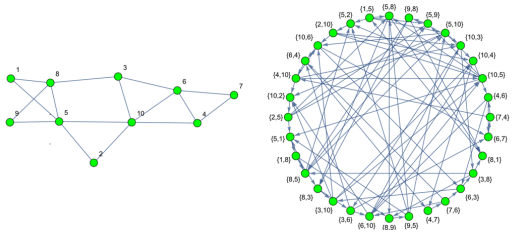

Fig. 10. Graph with ten vertices and sixteen edges (left) and its corresponding non-backtracking graph (right), where vertices represent directed edges of the original graph.

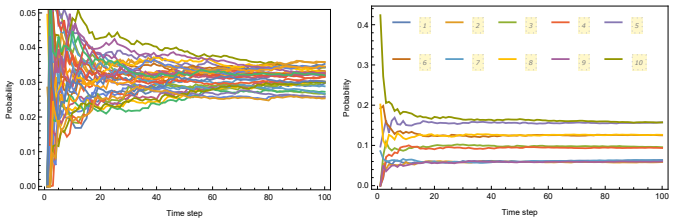

Fig. 11. Left: time-averaged probability of finding the quantum walker at each vertex of the non-backtracking graph with a uniformly random initial state. Right: the same distribution projected onto the original graph, where occupation of a vertex (i, j) in the non-backtracking graph is counted as the walker being at vertex j of the original graph.

B. A solvable case of non-backtracking quantum walk

So far, we have obtained an analytical solution only for the non-backtracking quantum walk on K_3. In this case, the non-backtracking classical walk admits only two possible trajectories: a clockwise cycle and a counterclockwise cycle. Once the direction is chosen, the walker continues deterministically along that cycle, since the non-backtracking rule forbids reversing the last step.

The triangle K_3 has six directed edges, which correspond to the six vertices of its non-backtracking graph. As shown in Fig. 12, this graph is the union of two disjoint directed 3-cycles. These two cycles define the permutations used to construct the shift operators of the quantum walk — one permutation acts as a cyclic shift on the upper triangle while leaving the lower triangle unchanged, and the other does the opposite. A Hadamard coin is then employed to select between the two permutations. The six position states are represented by the orthonormal basis vectors $|1\rangle, |2\rangle, \ldots, |6\rangle$.

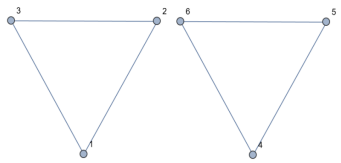

Fig. 12. Non-backtracking graph associated with the complete graph K_3.

Let $|0\rangle$ and $|1\rangle$ denote the basis states of the coin. The shift operator is defined as $S = |0\rangle\langle 0| S_1 + |1\rangle\langle 1| S_2$, where $S_1 = |2\rangle\langle 1| + |3\rangle\langle 2| + |1\rangle\langle 3|$ and $S_2 = |5\rangle\langle 4| + |6\rangle\langle 5| + |4\rangle\langle 6|$ implement the two cyclic shifts on position space. The unitary operator of the non-backtracking quantum walk is then $U = S \otimes H$, with H the Hadamard operator acting on coin space.

S_1 is diagonal in Fourier basis: $(|1\rangle, |2\rangle, |3\rangle) \rightarrow (|\alpha_1\rangle, |\alpha_2\rangle, |\alpha_3\rangle)$ with

- $|\alpha_1\rangle = \frac{1}{\sqrt{3}}(|1\rangle + |2\rangle + |3\rangle)$
- $|\alpha_2\rangle = \frac{1}{\sqrt{3}}(|1\rangle + \omega|2\rangle + \omega^2|3\rangle)$, $\omega := \exp(2\pi i/3)$
- $|\alpha_3\rangle = \frac{1}{\sqrt{3}}(|1\rangle + \omega^2|2\rangle + \omega|3\rangle)$, $(\omega^3 = 1)$

S_2 is diagonal in a similarly constructed Fourier basis:

- $(|4\rangle, |5\rangle, |6\rangle) \rightarrow (|\beta_1\rangle, |\beta_2\rangle, |\beta_3\rangle)$

We have the following:

- $S_1|\alpha_i\rangle = \lambda_i|\alpha_i\rangle$, $\lambda_1 = 1, \lambda_2 = \omega^{-1}, \lambda_3 = \omega^{-2}$
- $S_2|\beta_i\rangle = \lambda_i|\beta_i\rangle$

In order to diagonalize U, we need to choose appropriate basis for the coin. In coin basis $(|0\rangle, |1\rangle)$ we have $S \otimes H|0\rangle|\alpha_i\rangle = \frac{1}{\sqrt{2}}S(|0\rangle + |1\rangle)|\alpha_i\rangle = \frac{1}{\sqrt{2}}(\lambda_i|0\rangle + |1\rangle)|\alpha_i\rangle$ and $S \otimes H|1\rangle|\alpha_i\rangle = \frac{1}{\sqrt{2}}S(|0\rangle - |1\rangle)|\alpha_i\rangle = \frac{1}{\sqrt{2}}(\lambda_i|0\rangle - |1\rangle)|\alpha_i\rangle$, Or in other words, for any coin state $|\chi\rangle$: $U|\chi\rangle|\alpha_i\rangle = u_\alpha(i)|\chi\rangle|\alpha_i\rangle$ with $u_(\alpha_i) = \frac{1}{\sqrt{2}}\begin{pmatrix}\lambda_i & \lambda_i \\ 1 & -1\end{pmatrix}$ By diagonalizing 2×2 matrix $u(\alpha_i)$, we get a coin basis $(|a_i^1\rangle, |a_i^2\rangle)$ for a given position state $|\alpha_i\rangle$. Similarly for position states $|\beta_i\rangle$ we get matrices: $u(\beta_i) = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1 \\ \lambda_i & -\lambda_i\end{pmatrix}$ with a pair of eigenvectors which we denote as $(|b_i^1\rangle, |b_i^2\rangle)$.

As a result, U is diagonal in the basis $\{(|a_i^k\rangle|\alpha_i\rangle, |b_j^k\rangle|\beta_j\rangle)\}$, $k = 1, 2$ and $i, j = 1, 2, 3$. The eigenvectors and eigenvalues of U can be found analytically. Other eigenvalues are distinct, except for eigenvalues ± 1, which are doubly degenerate.

V. Conclusions and Final Discussions

We demonstrated the implementation of the Quantum Walk–based Optimization Algorithm (QWOA) on quantum hardware for a small hard instance, and realized Montanaro's construction of quantum walks on directed graphs to introduce a new model of non-backtracking quantum walk (NBQW). The experiments underline both the potential of these approaches and their current limitations, as noise dominates even in small circuits. These results provide a proof of principle while emphasizing the need for error mitigation and error correction to enable scalable applications.

Our near-term plans include optimizing the transpiled circuits and applying error mitigation techniques, in particular the tensor-network error mitigation (TEM) method developed by Algoritmiq [4]. Looking further ahead, we expect that longer coherence times in superconducting transmon qubits [7] and the eventual integration of quantum error correction will be essential for scaling these experiments. We will carry out theoretical studies of NBQW whicha are necessary to understand its potential for application.

References

[1] A. Ambainis. Quantum walk algorithm for element distinctness. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 22–31, 2004.

[2] A.Montanaro. Quantum walks on directed graphs. *Quantum Information & Computation*, 7(1):93–102, 2005.

[3] Marianna Bolla, Hannu Reittu, and Fatma Abdelkhalek. Clustering the Nodes of Sparse Edge-Weighted Graphs via Non-backtracking Spectra. In Kohei Arai, editor, *Advances in Information and Communication*, pages 81–99, Cham, 2025. Springer Nature Switzerland.

[4] Sergey N. Filippov, Sabrina Maniscalco, and Guillermo García-Pérez. Scalability of quantum error mitigation techniques: from utility to advantage, 2024.

[5] S. Marsh and J.B. Wang. A quantum walk-assisted approximate algorithm. *Quantum Inf. Process*, 18(3), 2019.

[6] R. Portugal. *Quantum Walks and Search Algorithms*, page 308. Springer, Cham, Switzerland, 2018.

[7] Mikko Tuokkola, Yoshiki Sunada, Heidi Kivijärvi, Jonatan Albanese, Leif Grönberg, Jukka-Pekka Kaikkonen, Visa Vesterinen, Joonas Govenius, and Mikko Möttönen. Methods to achieve near-millisecond energy relaxation and dephasing times for a superconducting transmon qubit. *Nature Communications*, 16(1):5421, 2025.