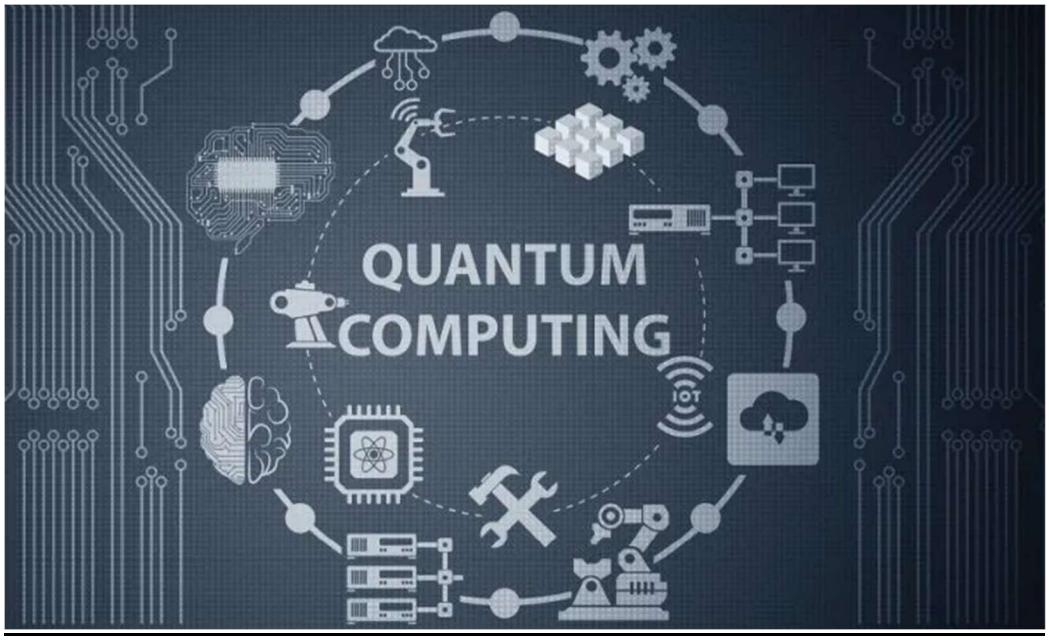
Full implementation of an IT security control on a Quantum Computer

Edition 1 (June 27, 2022)

Author: Christophe PARISEL

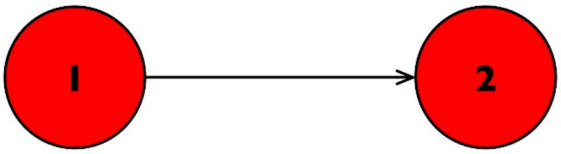


Permalink: https://github.com/labyrinthinesecurity/quantumComputing/blob/main/3_reachability.pdf

Quantum controls is a new and promising branch of **IT security**: it is expected to provide not only new kinds of controls that could complement the baseline found in compliance frameworks like CIS, but also provide a quadratical improvement in execution time.

In an article called *Introducing a new quantum control for IT security*[1], I provided a detailed design of one of the first quantum controls: by performing a random walk over all edges of a directed graph simultaneously, it may be used in a lot of situations where **asserting reachability** is important. So, it is not only useful for IT security (network access control, IAM, provable security), but for other branches of engineering especially networking.

In this short technical paper, we detail an end-to-end implementation of the quantum reachability control on the simplest network topology: the 2-vertices directed graph:



Its Laplacian matrix is simply:

$$\begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix}$$

Because the graph is directed, this matrix is not Hermitian: it needs to be transformed into a Hermitian operator using an eigenbasis transformation described in Quantum centrality testing on directed graphs via PT-symmetric quantum walks [3].

Pseudo-Hermitian to Hermitian transformation

We first calculate the eigenvectors phi0 and phi1 of the transposed Laplacian:

$$\varphi 0 = (0, 1) \quad \varphi 1 = \left(\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}} \right)$$

We calculate the outer products and sum them (equation 19 in the paper):

$$\varphi 0 \times \varphi 0 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \qquad \varphi 1 \times \varphi 1 = \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix} \qquad \sum_{i=0}^{1} \varphi i \times \varphi i = \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{3}{2} \end{pmatrix}$$

The (matrix) square root of this sum is the metric vector nu.

We left-multiply the Laplacian by nu and right-multiply by the inverse of nu to obtain the Hamiltonian (equation 9 in the paper):

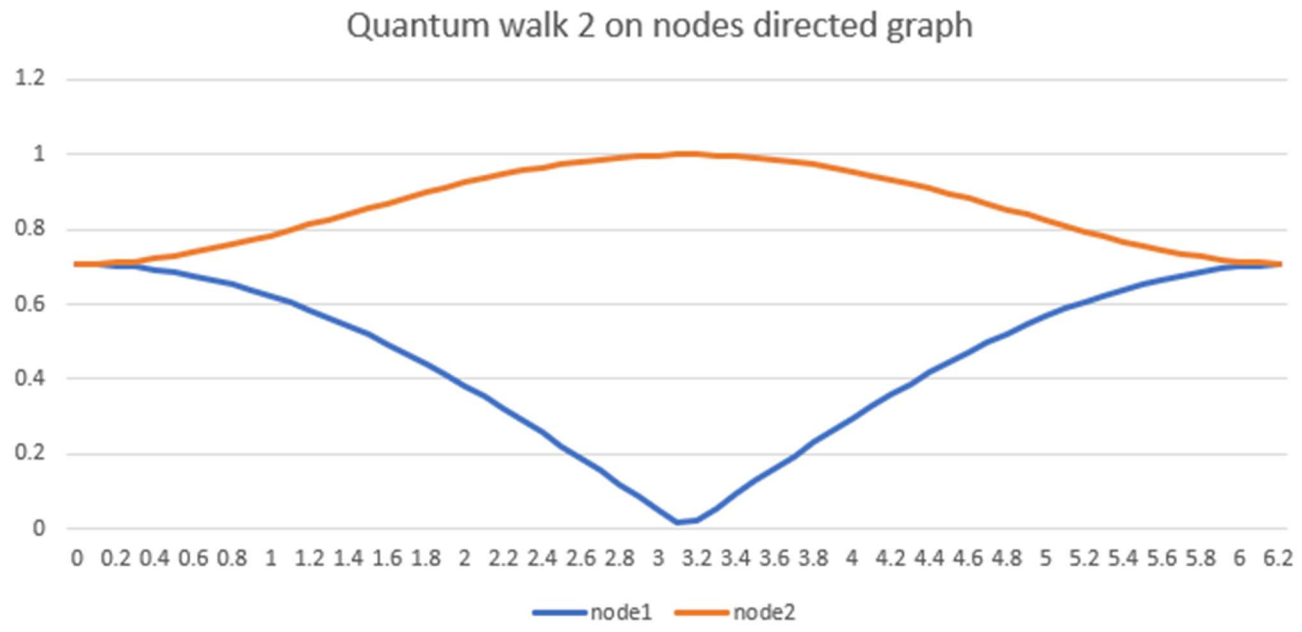
$$H = \eta L \eta^{-1}$$

The resulting Hamiltonian is:

$$\begin{pmatrix} \sqrt{2}+1 & -1 \\ -1 & \sqrt{2}-1 \end{pmatrix}$$

Quantum walks over the graph

If we set the initial state of a quantum walk to a single qubit in quantum superposition, running the Schrödinger equation with the Hamiltonian of the graph yields the following probability distribution:



We see that edge 1 is unreachable from edge 2 because of the null probability at time Pi modulo Pi.

Finding null probabilities is the key benefit of the quantum reachability control: it states that all edges with a null probability are unreachable from some part of the graph.

How to run this control on a quantum circuit?

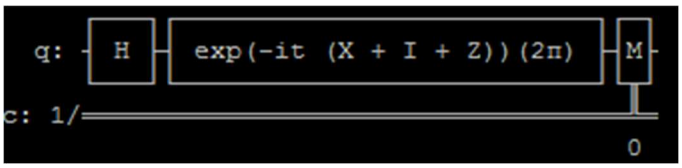
As explained in my article, we need to break down the Hamiltonian into simpler operators. Here is the most natural decomposition using the Pauli basis:

$$Z - X + I\sqrt{2}$$

Where I is the identity operator, Z and X correspond to Pauli-Z and Pauli-X.

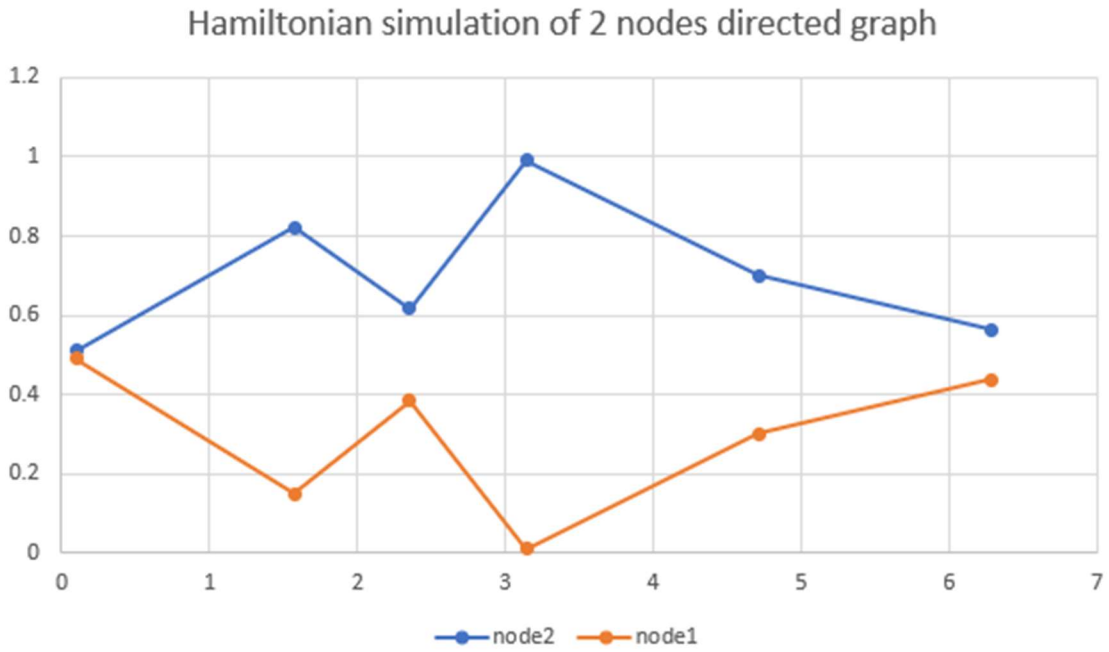
Building a quantum circuit for this linear combination is very straightforward with *QisKit* because it comes with the handy *PauliEvolutionGate*.

The circuit is available from my script hamiltonianSimulation2nodes.py [3]:



(Note that, as per *QisKit*'s documentation, the printed circuit doesn't show the coefficients applied to X, Z and I)

Here is one of the base Hamiltonian simulation samples I could get, you will see that the probability distribution of the state vector roughly follows the predicted behavior from Schrödinger equation:



One of the reasons why the sampling is sketchy is because the Hamiltonian decomposition generates errors (the Pauli matrices anti-commute).

But still, running the control seems to let us correctly identify the null probability at time t=Pi

Conclusion

Quantum Computing is no more theoretical. The number of useful applications is growing every day.

This proof of concept shows the practical interest of running Hamiltonian simulations on directed graphs for IT security compliance. I hope it will raise the awareness and the interest of the cybersecurity community in Quantum Computing.

References

[1] https://www.linkedin.com/pulse/introducing-new-quantum-control-security-christophe-parisel

[2] https://arxiv.org/abs/1607.02673

[3] https://github.com/labyrinthinesecurity/quantumComputing/blob/main/hamiltonianSimulation2nodes.py