Decrease in query complexity for quantum computers with superposition of circuits

Mateus Araújo,1, 2 Fabio Costa,1, 2 and Časlav Brukner1, 2

1Faculty of Physics, University of Vienna, Boltzmannasse 5, 1090 Vienna, Austria
2Institute of Quantum Optics and Quantum Information (IQOQI), Austrian Academy of Sciences, Boltzmannasse 3, 1090 Vienna, Austria

(Dated: February 3, 2014)

It is usually assumed that a quantum computation is performed by applying gates in a specific order. One can relax this assumption by allowing a control quantum system to switch the order in which the gates are applied. This provides a more general kind of quantum computing, that allows transformations on blackbox quantum gates that are impossible in a circuit with fixed order. Here we show that this model of quantum computing is physically realizable, by proposing an interferometric (in particular, linear optical) setup that can implement such a quantum control of the order between the gates. We show that this new resource provides a reduction in computational complexity: we propose a problem that can be solved using \( O(n) \) blackbox queries, whereas the best known quantum algorithm with fixed order between the gates requires \( O(n^2) \) queries.

I. INTRODUCTION

A useful tool to calculate the complexity of a quantum algorithm is the blackbox model of quantum computation. In this model, the input to the computation is encoded in a unitary gate – treated as a blackbox – and the complexity of the algorithm is the number of times this gate has to be queried to solve the problem.

Typically, black-box computation is studied within the quantum circuit formalism [1]. A quantum circuit consists of a collection of wires, representing quantum systems, that connect boxes, representing unitary transformations. In this framework, wires are assumed to connect the various gates in a fixed structure, thus the order in which the gates are applied is determined in advance and independently of the input states. It was first proposed in [2] that such a constraint can be relaxed: one can consider situations where the wires, and thus the order between gates, can be controlled by some extra variable. This is natural if one thinks of the circuit’s structure as a quantum system that can be in superposition.

Such superpositions of circuits allow performing information-theoretical tasks that are impossible in the quantum circuit model: it was shown in [3] that it is possible to decide whether a pair of blackbox unitaries commute or anticommute with a single use of each unitary, whereas in a circuit with a fixed structure at least one of the unitaries must be used twice. (The same task was considered in a quantum optics context in [4], where a less efficient protocol was found.)

It was not known, however, whether this advantage can be translated into more efficient algorithms for quantum computing, i.e., if a quantum computer that can have its circuit in a superposition can solve a computational problem with asymptotically less resources than a quantum computer with fixed circuit structure.

Here we present such a problem: given a set of \( n \) unitary matrices and the promise that they satisfy one out of \( n! \) specific properties, find which property is satisfied. The essential resource to solve this problem is the quantum control over the order of \( n \) blackboxes, first introduced in Ref. [5]. We show that, by using this resource, the problem can be solved with \( O(n) \) queries to the blackboxes, while the best algorithm with fixed order requires \( O(n^2) \) queries. Furthermore, while both quantum methods of solving the problem run in polynomial time, the best classical algorithm to solve it runs in exponential time, which may be of independent interest.

We further discuss a possible interferometric implementation of the protocol. For the superposition of the order of just two gates, a realization with current quantum optics techniques is possible. For a higher number of gates practical implementations become more challenging, but they are still possible in principle.

II. THE \( n \)-SWITCH GATE

The quantum control of the order between \( n \) unitary gates can be formalized by introducing the \( n \)-switch gate. As in Ref. [5], we consider a \( d \)-dimensional target system, initialized in some state \( |\psi\rangle \), and an \( n! \)-dimensional control system. Let \( \{U_i\}_{0}^{n-1} \) be a set of unitaries and

\[
\Pi_x = U_{\sigma_x(n-1)} \ldots U_{\sigma_x(1)} U_{\sigma_x(0)}
\]

for some permutation \( \sigma_x \), where \( x = 0, \ldots n! - 1 \) is a chosen labeling of permutations\(^1\). Then the \( n \)-switch \( S_n \) is a controlled quantum gate: its effect is to apply

---

\(^1\) Conventionally, \( \sigma_0 \) represents the identity permutation: \( \sigma_0(j) = j \).
the product of unitaries $\Pi_x$ to the state $|\psi\rangle$ conditioned on the value of the control register $|x\rangle$. In symbols,

$$S_x|x\rangle|\psi\rangle = |x\rangle\Pi_x|\psi\rangle. \quad (2)$$

### III. ALGORITHM

We present here an algorithm that can exploit the quantum control of orders provided by the $n$-switch gate to achieve a reduction in query complexity for the solution of a specific problem. The algorithm is based on the standard Hadamard test. The idea is to initialize the control system in a uniform superposition of all permutations, apply $S_n$, and then measure the control system in the Fourier basis. With a suitable choice of the unitaries, we can make the result of this measurement deterministic and, since there are $n!$ different results, this means that we can differentiate between $n!$ different properties of $n$ unitaries.

To be more precise, let $\omega = e^{2\pi i/n}$. We say that the set of unitaries $\{U_i\}_{i=0}^{n-1}$ has property $P_y$ if it is true that

$$\forall x \Pi_x = \omega^{xy} \Pi_0, \quad (3)$$

for the given $y$. For example, property $P_0$ is the property that $\Pi_x = \Pi_0$ for all $x$, i.e., that all the matrices commute with each other.

Note that it is not possible to satisfy property $P_1$ if the dimension of the unitaries $d$ is less than $n!$. To see that, consider $x = y = 1$, and take the determinant on both sides of equation (3):

$$\det \Pi_1 = \omega^d \det \Pi_0. \quad (4)$$

Since $\det \Pi_1 = \det \Pi_0$, it follows that $\omega^d = 1$, and therefore $d$ must be at least $n!$.

The computational problem is defined as follows: given a set $\{U_i\}_{i=0}^{n-1}$ of unitary matrices of dimension $d \geq n!$, decide which of the properties $P_y$ is satisfied by this set, given the promise that one of these $n!$ properties is satisfied.

The protocol for solving this problem is the following: initialize the target system in any state $|\psi\rangle$, and the control system in the state $|C\rangle$ which corresponds to an equal superposition of all permutations:

$$|C\rangle|\psi\rangle = \frac{1}{\sqrt{n!}} \sum_{x=0}^{n-1} |x\rangle|\psi\rangle. \quad (5)$$

Then, we apply the $n$-switch:

$$S_n|C\rangle|\psi\rangle = \frac{1}{\sqrt{n!}} \sum_{x=0}^{n-1} |x\rangle\Pi_x|\psi\rangle. \quad (6)$$

Now we apply the Fourier transform over $Z_n$ to our control qudit:

$$\mathcal{F}_n S_n |C\rangle|\psi\rangle = \frac{1}{\sqrt{n!}} \sum_{x,y=0}^{n-1} |y\rangle \omega^{-xy} \Pi_x|\psi\rangle. \quad (7)$$

and measure the control qudit in the computational basis, with outcome probabilities

$$p_y = \frac{1}{n!} \left| \sum_{x=0}^{n-1} \omega^{-xy} \Pi_x|\psi\rangle \right|^2. \quad (8)$$

Note that if the set of unitaries has property $P_0$ then $p_0 = 1$. In general it is easy to check that if they have property $P_y$ then $p_y = 1$, so we can find out which property they have with probability one in a single run of the protocol.

We should notice that the problem is not trivial, i.e. there exist, for every $n$, infinitely many sets of matrices that satisfy each of the $n!$ properties $P_y$ (see Appendix A.) The problem, and the corresponding protocol, can be also modified to tolerate possible experimental error. This modification is shown in Appendix B.

### IV. QUERY COMPLEXITY

We are interested in determining the number of times that the unitaries $U_i$ must be used to run the algorithm. Clearly this depends only on the implementation of the $n$-switch gate, since the unitaries are not used anywhere else. As proposed in [2], the switch can in principle be implemented by adding quantum control to the connections between the unitaries. In such an implementation it is sufficient to use a single copy of each unitary, while the control system determines the order in which the target system passes through the unitaries. Remarkably, it was proven in [2] that no quantum circuit can implement classical control over the order between two unitaries. This also implies the impossibility of quantum control of the order between $n$ unitaries in a circuit with one copy of each unitary.

Since the implementation with quantum control of the connections between gates is explicitly outside the quantum circuit formalism, we cannot simply calculate the number of uses of the unitaries by counting the number of times they appear in a circuit. Nevertheless, we can formulate the notion of “gate uses” in a precise, operational, way. Imagine we append, to each gate, an additional “flag” quantum system that counts the number of times that gate is used. This can be done
in a reversible way: the \( j \)-th flag is initialized in the state \( |0\rangle \), and, whenever the unitary \( U_i \) is used, it is updated through the unitary transformation \( |f\rangle \rightarrow |f + 1\rangle \). It is easy to see that, after applying the \( n \)-switch, the state of the flags factorizes, with each flag in the state \( |1\rangle \). According to this definition, the total number of queries necessary to run the algorithm is \( n \).

In comparison, the best way we found\(^3\) of simulating the \( n \)-switch gate with a fixed circuit has query complexity \( O(n^2) \). The following circuit illustrates the idea for \( n = 2 \). (Note that, in the special case of the \( 2 \)-switch, it is sufficient to apply one of the unitaries twice, see Ref. [3].)

\[
\begin{align*}
|\psi\rangle & \\
|a_0\rangle & \\
|a_1\rangle & \\
\end{align*}
\]

where we adopt the standard graphical representation for controlled-swap gates [6] (the white dot denotes control on the state \( |0\rangle \), while the black dot denotes control on the state \( |1\rangle \)). The general implementation of the \( n \)-switch is discussed in appendix C.

Of course, it might not be necessary to use the \( n \)-switch gate in order to determine which property \( P_y \) the unitaries satisfy. For example, it is possible to solve the problem by directly measuring the phase obtained when applying the permutation \( \sigma_1 \). Since \( \Pi_1 = \omega^y \Pi_0 \), this is sufficient to determine \( y \). However, this protocol can work only if the relative phase is measured with an error smaller than \( \frac{\pi}{2^n} \), and for blackbox unitaries this can only be done with an exponential amount of queries.

This is the case, for example, for Kitaev’s phase estimation algorithm [7]. This algorithm is not usually applied to blackbox unitaries, but this can be done using the techniques in [8–16]. In this case, to calculate the phase with the required \( O(n \log n) \) bits of precision, one would need to implement the matrices control-\( U_i^2 \), with \( k = 1, \ldots, n \log n \), which would require an exponential amount of copies of the blackboxes \( U_i \). Event if one assumes that it is possible to apply control-\( U_i^2 \) efficiently – a necessary assumption to make Kitaev’s algorithm efficient – one would need \( O(n \log n) \) queries to each \( U_i^2 \) oracle. Since there are \( n \) unitaries \( U_i \), the query complexity would be \( O(n^2 \log n) \), which is still less efficient than simulating the \( n \)-switch with a fixed circuit.

V. RUNNING TIME

Instead of query complexity we may want to consider the running time of the algorithm. If we assume that this is dominated by applying the unitaries \( U_i \), then there is no difference between the implementation with superposition of circuits or the fixed quantum circuit: both run in time \( O(n) \).

It is interesting, nevertheless, to compare the time required to solve the problem between quantum and classical computers. If we assume that the unitaries \( U_i \) are decomposed in a polynomial amount of elementary gates, they can be given as an input of polynomial size to a classical algorithm, and it makes sense to compare the classical and quantum running times.

As argued above, the problem of determining \( y \) reduces to the problem of calculating the relative phase between \( \Pi_1 \) and \( \Pi_0 \), which may differ by the permutation of a single pair of unitaries. However, as discussed before, the dimension of the unitaries must be at least \( n! \) for this problem to be nontrivial, and it seems unlikely that one could extract the phase from these exponentially large unitary matrices on a classical computer in polynomial time. On the other hand, the running time of the quantum algorithm is clearly polynomial for unitaries decomposed in a polynomial amount of elementary gates. Therefore we conjecture that for the problem presented there is an exponential separation between classical and quantum complexity, which may be of independent interest.

VI. PHYSICAL IMPLEMENTATION

In Ref. [3] it was proposed to apply the superposition principle to the physical components of a quantum computer that determine the order between gates. Since this requires a quantum control over macroscopic systems, it seems outside of the reach of current technology and could be practically unfeasible. Here we propose an implementation of the \( n \)-switch that, although experimentally challenging, might be feasible.

We first consider an optical implementation of a \( 2 \)-switch for \( 2 \times 2 \) unitaries, illustrated in Fig. 1. The control system is the polarization of a photon and the target system some internal degree of freedom of the same photon, such as space bins, time bins, or angular momentum modes. If the photon is prepared in a horizontally polarized state \( |\mathcal{H}\rangle \), it is transmitted by both polarizing beam splitters (PBSs), resulting in the application of the unitary \( U_0 \) first and of \( U_1 \) second. A

\(^{3}\) The quantum circuit implementation of the \( n \)-switch proposed in [5] only works for basis elements of the control, i.e., for classical control of the order, so it cannot be used for our protocol.
A photon in a vertically polarized state $|V\rangle$ is reflected by both PBSs, thus the two unitaries are applied in the reversed order. For an arbitrary polarization state $\alpha |H\rangle + \beta |V\rangle$, the photon exits the interferometer in the state $\alpha |H\rangle U_1 U_0 |\psi\rangle + \beta |V\rangle U_0 U_1 |\psi\rangle$, which corresponds to the output of the 2-switch.

The extension of this scheme to the general case of an $n$-switch can be obtained with a generalization of the PBS to an element, which we call $n$-router, with $n$ input modes and $n$ output modes (see Fig. 2). If the control system is in a state $|x\rangle$, the $n$-router sends the input mode $j$ to the output mode $c_x(j)$. The unitary $U_{c_x}(j)$ is applied to a system in the mode $c_x(j)$, which then enters a second router that performs the inverse permutation. The output mode $j$ of the second router is then directed to the input mode $j+1$ of the first one. It is straightforward to check that a system entering mode 0 of the first router in the state $|x\rangle|\psi\rangle$ exits mode $n-1$ of the second router in the state $|x\rangle \Pi_x |\psi\rangle$. (In appendix D we show how to construct an $n$-router with $O(n^2)$ binary routers.)

This higher-dimensional routing can be achieved, for example, with orbital angular momentum of light [11, 12]. However, the main limitation of an optical implementation of the $n$-router is that it is not scalable in an obvious way, since it requires encoding an exponential number of degrees of freedom in a single photon (n! for the control system and, as argued in section III, at least n! for the target system). A scalable implementation could be obtained by encoding the degrees of freedom in $O(n \log n)$ particles, each carrying a constant number of degrees of freedom (e.g., one qubit each). The main challenge is then to implement a router that, conditioned on the multiparticle state, coherently directs all the particles in a specific mode. This is in principle possible if the particles are bound together, e.g. as atoms in a molecule. Recent progress in matter-wave interferometry suggests that such a quantum control of composite systems could be achievable in the future [13, 14].

Other realizations of superposition of orders, based on different models of computation, could also be possible. For example, an implementation of the 2-switch within adiabatic quantum computing was proposed recently [15].

VII. CONCLUSIONS

We have shown that superpositions of circuits or, more specifically, quantum control of the order between gates, can provide a reduction in the number of queries needed to solve a specific computational problem. While the reduction is only polynomial, the result shows that extending the quantum circuit model is possible and can provide a computational advantage. Furthermore, we have proposed a physically realizable experimental implementation of the protocol.

Other extensions of the quantum circuit model of blackbox computation have been proposed [10]: it was shown recently [10, 16, 17] that a quantum circuit cannot apply blackbox gates conditioned on the state of a control qubit. However, such a control is physically realizable [8, 9] and therefore should be allowed by the formalism. It is an intriguing open question what further computational models could be conceived, once the restrictions imposed by the fixed causal structure of quantum mechanics are relaxed [18].

ACKNOWLEDGMENTS

This work was supported by the Austrian Science Fund (FWF) (Complex Quantum Systems (CoQuS),
[1] D. Deutsch, “Quantum Computational Networks,” Proc. R. Soc. Lond. A 425, 73–90 (1989).
[2] G. Chiribella, G. M. D’Ariano, P. Perinotti, and B. Valiron, “Quantum computations without definite causal structure,” Phys. Rev. A 88, 022318 (2013), arXiv:0912.0195 [quant-ph].
[3] G. Chiribella, “Perfect discrimination of no-signalling channels via quantum superposition of causal structures,” Phys. Rev. A 86, 040301 (2012), arXiv:1109.5154 [quant-ph].
[4] E. Andersson, J. Bergou, and I. Jex, “Comparison of unitary transforms using Franson interferometry,” J. Mod. Opt. 52, 1485–1494 (2005).
[5] T. Colnaghi, G. M. D’Ariano, S. Facchini, and P. Perinotti, “Quantum computation with programmable connections between gates,” Phys. Lett. A 376, 2940–2943 (2012), arXiv:1109.5987 [quant-ph].
[6] M. Nielsen and I. Chuang, Quantum Computation and Quantum Information. Cambridge University Press, 2000.
[7] A. Y. Kitaev, “Quantum measurements and the Abelian Stabilizer Problem,” quant-ph/9511026.
[8] X.-Q. Zhou, T. C. Ralph, P. Kalasuwan, M. Zhang, A. Peruzzo, B. P. Lanyon, and J. L. O’Brien, “Adding control to arbitrary unknown quantum operations,” Nat. Commun. 2, (2011), arXiv:1006.2670 [quant-ph].
[9] X.-Q. Zhou, P. Kalasuwan, T. C. Ralph, and J. L. O’Brien, “Calculating unknown eigenvalues with a quantum algorithm,” Nature Photonics 7, 223–228 (2013), arXiv:1110.4276 [quant-ph].
[10] M. Araújo, A. Feix, F. Costa, and Č. Brukner, “Quantum circuits cannot control unknown operations,” arXiv:1309.1794 [quant-ph].
[11] M. Mirhosseini, M. Malik, Z. Shi, and R. W. Boyd, “Efficient separation of the orbital angular momentum eigenstates of light,” Nat. Commun. 4, (2013), arXiv:1306.0849 [quant-ph].
[12] G. C. G. Berkhoult, M. P. J. Lavery, J. Courtial, M. W. Beijersbergen, and M. J. Padgett, “Efficient Sorting of Orbital Angular Momentum States of Light,” Phys. Rev. Lett. 105, 153601 (2010).
[13] M. Arndt, O. Nairz, J. Voss-Andreae, C. Keller, G. van der Zouw, and Z. A., “Wave-particle duality of C60 molecules,” Nature 401, 680–682 (1999).
[14] S. Eibenberger, S. Gerlich, M. Arndt, M. Mayor, and J. Tuxen, “Matter-wave interference with particles selected from a molecular library with masses exceeding 10 000 amu,” Phys. Chem. Chem. Phys 15, 14666–14700 (2013).
[15] K. Nakago, M. Hajdušek, S. Nakayama, and M. Murao, “Parallelized adiabatic gate teleportation,” arXiv:1310.4061 [quant-ph].
[16] A. Soeda, “Limitations on quantum subroutine designing due to the linear structure of quantum operators,” Talk at ICQIT2013.
[17] J. Thompson, M. Gu, K. Modi, and V. Vedral, “Quantum Computing with black-box Subroutines,” arXiv:1310.2927 [quant-ph].
[18] O. Oreshkov, F. Costa, and Č. Brukner, “Quantum correlations with no causal order,” Nat. Commun. 3, (2012), arXiv:1105.4464 [quant-ph].

Appendix A: Existence proof of the sets of unitaries

We need to show that for every $y$ there exists a set of unitaries that satisfies property $P_y$, otherwise the problem becomes trivial. More specifically, we will show that for every integer $n \geq 1$ and every $y \in \{0, \ldots, n! - 1\}$ it is possible to find a set of $n$ unitary matrices $\{U_i\}_{i=0}^{n-1}$ such that

$$\Pi_y = \omega^{xy} \Pi_0$$

is true for all $x$ for some choice of $y$, where

$$\Pi_x = U_{\omega_x(n-1)} \cdots U_{\omega_x(1)} U_{\omega_x(0)}$$

and $\omega = e^{i2\pi/n}$.

We first present the construction for $y = 1$. In this case, to each permutation of the matrices corresponds a different phase $\omega^x$. We start by proving that the pairwise relations

$$U_k U_j = \omega^{kj} U_k U_j \quad \text{for} \quad k, j \geq 0 \quad (A3)$$

generate all the $n!$ required phases. In order to prove this, it is convenient to introduce an explicit labeling of the permutations. A generic permutation of the $n$ matrices $\{U_0, \ldots, U_{n-1}\}$ is labeled by a sequence of integers $(a_{n-1}, \ldots, a_1)$, with $0 \leq a_k \leq k$, and is obtained by shifting the matrix $U_k$ to the right $a_k$ times, starting from $k = 1$. As an example, let us construct the four-element permutation labeled by $(3,1,1)$. First, we swap $U_0$ and $U_1$ (i.e., we shift $U_1$ one position to the right), obtaining $U_3 U_2 U_0 U_1$ then we shift $U_2$ one position to the right and obtain $U_3 U_2 U_0 U_1$. Finally, we shift $U_3$ three times to the right and get the desired permutation, $U_0 U_2 U_1 U_3$. In this procedure, each matrix $U_k$ is swapped $a_k$ times with matrices $U_j$ having $j < k$. Thus, if the relations $(A3)$ hold, the permutation labeled by $(a_{n-1}, \ldots, a_1)$ produces a phase $\omega^x$, where

$$x = \sum_{k=1}^{n-1} a_k k! \quad (A4)$$

is the expansion of the integer $x$ in the factorial basis (or factoradic expansion). For the given example, $x = 3 \times 3! + 1 \times 2! + 1 \times 1! = 21$. 

Special Research Program Foundations and Applications of Quantum Science (FoQuS), Individual project 24621, the European Commission Project RAQUEL, FQXi, and by the John Templeton Foundation.
We construct now a set of \( n \) unitary matrices that satisfy the pairwise relations (A3). They are constructed from the generalized \( X \) and \( Z \) matrices
\[
X = \sum_{j=0}^{n-1} |j \oplus 1 \rangle \langle j|, \quad (A5a)
\]
\[
Z = \sum_{j=0}^{n-1} \omega^j |j \rangle \langle j|, \quad (A5b)
\]
where \( \oplus \) denotes the sum modulo \( n! \). Note that they satisfy the commutation relation
\[
ZX = \omega XZ. \quad (A6)
\]
Then we define
\[
U_k = (Z^k)^{\otimes n} \otimes X \otimes 1^{\otimes n-k-1} \quad \text{for} \quad k < n - 1,
\]
\[
U_{n-1} = (Z^{(n-1)!})^{\otimes n}, \quad (A7)
\]
where we are using the convention that \( A^{\otimes k} = 1 \) for \( k \leq 0 \). For \( n = 4 \), the set of unitaries defined by (A7) reads
\[
U_0 = X \otimes 1 \otimes 1,
U_1 = Z \otimes X \otimes 1,
U_2 = Z^2 \otimes Z^2 \otimes X,
U_3 = Z^6 \otimes Z^6 \otimes Z^6. \quad (A8)
\]
It is easy to check that the matrices generated according to these rules satisfy property (A3). A set of \( n \) unitary matrices that satisfies property (3) for an arbitrary \( y \) can be found by repeating this construction with \( \omega^y \) in place of \( \omega \).

Note that this construction is enough to show that there is an infinity of sets of unitaries satisfying property (3), since the choice of basis in the definition of the matrices \( X \) and \( Z \) in equation (A5) is arbitrary. This construction generates matrices with dimension \( d = n^{n-1} \). We can get constructions with the correct phases in lower dimension, which might be interesting for an experimental implementation. For example, for \( n = 3 \), the following 6-dimensional matrices also work:
\[
U_0 = X,
U_1 = ZX,
U_2 = Z^2. \quad (A9)
\]
This construction saturates the lower bound \( d \geq n! \) we presented in section III. It is an interesting puzzle to find out whether there exists a construction with \( d = n! \) for every \( n \).

Appendix B: Tolerating experimental error

In an experimental implementation, no property \( P_y \) can be satisfied exactly, so to run this algorithm one needs to be able to tolerate some deviations from it. To do this, notice that property \( P_y \) is satisfied if and only if
\[
\frac{1}{n!^2 d} \left\| \sum_{x=0}^{n-1} \omega^{xy} \Pi_x \right\|_{HS}^2 = 1, \quad (B1)
\]
where \( \| \cdot \|_{HS} \) is the Hilbert-Schmidt norm. This is a simple consequence of the fact that this norm is defined through an inner product. We then reformulate the problem such that a set of unitaries satisfies the modified property \( P'_y \) if
\[
\frac{2}{3} \leq \frac{1}{n!^2 d} \left\| \sum_{x=0}^{n-1} \omega^{xy} \Pi_x \right\|_{HS}^2 \leq 1 \quad (B2)
\]
for a given \( y \in \{0, \ldots, n! - 1\} \). The problem is still to decide which property \( P'_y \) the set of unitaries have, given the promise that they have one of them.

Note that, in this version of the problem, we cannot anymore use an arbitrary pure state \( |\psi\rangle \) to perform the protocol, since it is not true anymore that the probabilities \( p_y \) in Eq. (8) are independent of \( |\psi\rangle \). In fact, it is possible to have a set of matrices such that the lhs of Eq. (B1) is arbitrarily close to 1, while the probability \( p_y \) in Eq. (8) for some state \( |\psi\rangle \) is equal to 0.

Instead, we can use the maximally mixed state, since then the outcome probabilities \( p_y \) become directly related to the Hilbert-Schmidt norm, as
\[
p_y = \frac{1}{n!^2 d} \left\| \sum_{x=0}^{n-1} \omega^{xy} \Pi_x \right\|_{HS}^2 \quad (B3)
\]
As before, the algorithm is that if we obtain outcome \( y \), then we guess property \( P'_y \). Since this is now a probabilistic algorithm, we must check for completeness and soundness. Completeness means that the probability of answering \( P'_y \) given that \( P'_y \) is true must be larger than 2/3, while soundness means that the probability of answering \( P'_y \) when \( P'_y \) is false must be smaller than 1/3.

Completeness is implied by the definition of \( P'_y \): if it is true, then the probability \( p_y \) is larger than 2/3. Soundness follows from the promise that one of \( P'_y \) must be true, since this implies that the probability of all the other outcomes must be smaller than 1/3.

Appendix C: Implementing the n-switch gate in the quantum circuit model

Here we describe how to simulate the \( n \)-switch gate in the quantum circuit model. The simulation is based
on the circuit presented on Ref. [5], with the difference that our scheme can be used for quantum (and not only classical) control of the order.

In the main text we described the control system $|C\rangle$ as encoding the permutation to be applied simply as a state running from $|0\rangle$ to $|n!-1\rangle$. Here we shall use a more convenient representation, expressing $|C\rangle$ as a sequence of $n$ $n$-dimensional qudits $|C_k\rangle$, where $|C_k\rangle$ runs from $|0\rangle$ to $|n!-1\rangle$, and indicates the unitary to be applied in the position $k$. For example, to encode the permutation that first applies $U_1$, then $U_0$, and then $U_2$, we shall write $|C\rangle = |102\rangle$. This representation requires $n\lceil \log_2 n \rceil$ qudits, a small overhead over the $\lceil \log_2 n! \rceil$ qudits that are necessary to encode a permutation of $n$ elements. We also remark that the conversion between these two representations can be done on a classical computer in polynomial time, and therefore we shall note it no further.

The circuit consists of a composition of $n$ instances of the following element, where $k$ goes from 1 to $n$:

![Circuit Diagram]

The $|a_i\rangle$ are ancillae, and $S$ is a gate that swaps $|\psi\rangle$ with the ancilla $|a_i\rangle$ controlled on $|C_k\rangle = |i\rangle$, leaving the other ancillae invariant. This gate clearly can be implemented with a linear amount of elementary gates, so we shall not discuss its implementation. Note that each element contains $n$ unitaries; therefore, since we need $n$ of these elements to implement the $n$-switch, the total number of queries in this implementation is $n^2$.

For example, using this construction for $n = 3$ gives us the circuit

![Circuit Diagram for n = 3]

Note that when each unitary is applied only once, $i.e.$, when $|C\rangle$ encodes a permutation, then the ancillae disentangle from the control system at the end of the circuit. In general, this is the case when for each value of the control system, each unitary is applied the same number of times. For example, when $|C\rangle$ is a superposition of the sequences $|002\rangle, |020\rangle$, and $|200\rangle$ the ancillae still disentangle, but not when it is a superposition of $|002\rangle$ and $|021\rangle$.

Appendix D: Decomposition of the $n$-router

We show how an $n$-router can be constructed using a polynomial number of elementary resources, each equivalent to a polarizing beam splitter (PBS). Let $|x\rangle$ be a basis element of the control system, and $|j\rangle_{\text{in}}$ denote the $j$-th input mode to the router, with $j = 0, \ldots, n-1$ (for simplicity, we don’t write explicitly the target system). Then the $n$-router performs the transformation $|x\rangle|j\rangle_{\text{in}} \mapsto |x\rangle|\sigma_x(j)\rangle_{\text{out}}$, where $|\sigma_x(j)\rangle_{\text{out}}$ is the $\sigma_x(j)$-th output mode. In order to decompose this gate in elementary ones, we first express the control variable in terms of its factoradic representation, introduced in Appendix A: $x \leftrightarrow (a_{n-1}, \ldots, a_1)$, with $0 \leq a_k \leq k$. Then, we represent each coefficient $a_k$ using $k$ bits$^4$ $b_{k1}, \ldots, b_{kk}$, with $b_{kj} = 1$ for $a_k < b_{kj}$ and $b_{kj} = 0$ for $j \geq a_k$. For example, the values of $a_3$ are represented as $a_3 = 0123$.

In this way, we can encode the control quantum system in $\frac{n(n-1)}{2}$ qubits, $|x\rangle \rightarrow \bigotimes_{k=1}^{n} \bigotimes_{j=1}^{k} |b_{kj}\rangle$. Now we can construct the $n$-router using a controlled binary swap of modes for each control qubit $|b_{kj}\rangle$. Recall that an arbitrary permutation $|j\rangle_{\text{in}} \mapsto |\sigma_x(j)\rangle_{\text{out}}$ is obtained shifting each mode $k$ “to the right” by a number $a_k$ of positions, $i.e.$ applying the unitary $|k\rangle_{\text{in}} \mapsto |k-a_k\rangle_{\text{out}}$, $|j\rangle_{\text{in}} \mapsto |j+1\rangle_{\text{out}}$ for $a_k \leq j < k$. The idea is to decompose this shift in swaps between neighboring modes, with each swap controlled by one control qubit. Explicitly, the controlled mode-swaps are defined as

$$
|b_{kj}\rangle|j-k\rangle_{\text{in}} \mapsto |b_{kj}\rangle|k-j+b_{kj}\rangle_{\text{out}} \quad (D1)
$$

and identity for the other modes. The $n$-router is then obtained by first applying the mode-swap controlled by $|b_{k1}\rangle$, then the one controlled by $|b_{k2}\rangle$ followed by the one controlled by $|b_{k2}\rangle$ and so on, applying successively each mode-swap controlled by $|b_{kj}\rangle$ increasing $k$ and, for each $k$, increasing $j$.

$^4$ This encoding is clearly not optimal, since it requires $O(n^2)$ bits to encode the $n!$ permutations, instead of the required $O(n \log n)$. We will however not invest more time to optimize this aspect of the problem.
As an example, consider a permutation identified by the single non-vanishing factoradic coefficient $a_3 = 2$. The corresponding control qubits are then in the states $|b_{31} = 1\rangle|b_{32} = 1\rangle|b_{33} = 0\rangle$. First we apply the swap between modes 3 and 2 controlled on $|b_{31}\rangle$. Since $b_{31} = 1$ this results in the mode-swap $|3\rangle \leftrightarrow |2\rangle$. Then, since $b_{32} = 1$, the modes 2 and 1 are swapped. Composing the two swaps, we obtain the transformation $|3\rangle \rightarrow |1\rangle$, $|1\rangle \rightarrow |2\rangle$, $|2\rangle \rightarrow |3\rangle$. Since $b_{33} = 0$, the last mode-swap is not applied, the result of the two mode-swaps is therefore the shift of mode 3 by $a_3 = 2$ positions to the right. By composing this procedure for an arbitrary set of coefficients $(a_{n-1}, \ldots, a_1)$, the corresponding permutation of modes is realized.

The building block of this construction is the controlled swap of two modes, which is essentially equivalent to a PBS (for a PBS, the control qubit is the photon polarization). We stress that this element does not correspond to any traditional elementary gate of a quantum circuit, and it should thus be considered as a new elementary resource.

In Ref. [5], a different approach was proposed for the scalable realization of the $n$-switch: a construction was proposed that realizes the $n$-switch using $O(n^2)$ 2-switches. This construction, however, is not applicable to our case, since our definitions differ slightly\(^5\). Furthermore, the construction based on the $n$-router element is more directly related to the interferometric implementation proposed in the main text.

---

\(^5\) According to the definition from [5], the $n$-switch orders the $n$ unitaries according to the prescribed permutation, but it does not directly compose them to each other. It is thus possible to plug additional elements between any pairs of unitaries, making the construction of the $n$-switch from 2-switches possible. The 2-switch of [5] can be reproduced by a 3-switch as defined here, with the prescription that only the first and last position are swapped, while the position in the middle is fixed and can be used either as an identity or to plug input and output of another switch.