Electronic Colloquium on Computational Complexity, Report No. 327 (2026)

What Can Be Computed with Indefinite Causal Structure?

Omri Shmueli*

Abstract

Can quantum computation become more powerful when operations do not need to follow a definite causal order? We study this question in the process matrix formalism of Oreshkov et al. (Nature Communications, 2012), which allows quantum operations to be connected without a fixed global order. In the computational model, a quantum circuit succinctly describes a process matrix, and the algorithm is given access to its generated channel, with computational cost measured by the size of the generating circuit.

We give polynomial-time algorithms in two settings. First, general process matrix computation suffices to solve every problem in **NP**. The valid processes used by our algorithm satisfy the purification postulate of Araújo et al. (Quantum, 2017), and their generated channels are unitary, although their generating circuits are not. Second, requiring the process matrices themselves to be pure, we give a polynomial-time algorithm for unique-sink orientation, a general problem capturing several longstanding computational problems in optimization. Our unitary generators are constructed explicitly and efficiently. Assuming that unique-sink orientation has no quantum polynomial-time algorithm, our result shows that indefinite causal order can provide computational power beyond **BQP** within the process matrix formalism, even while preserving unitarity.

*omri.shmueli1@gmail.com

ISSN 1433-8092

Contents

1 Introduction **1**
- 1.1 Results . . . 3
- 1.2 Related Work . . . 5

2 Technical Overview **6**
- 2.1 General process matrix computation and NP . . . 7
- 2.2 Purifiable versus pure processes . . . 9
- 2.3 Pure processes and unique-sink orientation . . . 10

3 Preliminaries **13**
- 3.1 Computational Complexity Theory . . . 14
- 3.2 Quantum Information Theory . . . 15

4 Process Matrix Formalism **17**
- 4.1 A Computational Model based on Process Matrices . . . 19
- 4.2 Existing and New Complexity Classes . . . 23

5 Quantum Computation with Efficient General Process Matrices **26**
- 5.1 Generating a unitary channel from a generator of its Choi state . . . 26
- 5.2 Finding a unique satisfying input . . . 31
- 5.3 Solving **NP** . . . 32
- 5.4 Inverting an efficiently computable permutation . . . 33

6 Quantum Computation with Efficient and Pure Process Matrices **34**
- 6.1 A unitary generator for the USO Oracle . . . 34
- 6.2 Connections to physical implementation . . . 39

A Supplementary Proofs **43**
- A.1 The USO characterization and validity of our construction . . . 43

1 Introduction

One of the basic questions in our understanding of nature is whether physical events must have a definite causal structure, namely, a relation specifying which events can influence which others. In general relativity, these relations depend on the geometry of spacetime, which is itself dynamical. If spacetime geometry is subject to quantum uncertainty, as may occur in quantum gravity, it raises the possibility that its causal structure can also be indefinite [Har05, Har07]. This possibility, referred to as *indefinite causal structure* (ICS), motivates studying information processing without a predetermined causal order. In particular, it raises the question of whether ICS can itself be a computational resource [Har09, CDPV13].

The quantum circuit model, based on standard quantum mechanics, allows superpositions of physical states, but the circuit still specifies a causal order (formally, a partial order) between the execution of its gates. It is therefore natural to ask whether relaxing the latter requirement can change the computational power of the model. To make this question meaningful, however, one needs to specify which connections between quantum operations are allowed, why they define consistent computations, and how to measure their computational cost.

Process matrices: local quantum mechanics without a fixed global order. Oreshkov, Costa and Brukner [OCB12] introduced the *process matrix formalism* (PMF), which has since become a central framework for studying indefinite causal order. The formalism separates two assumptions. The first is that ordinary quantum mechanics governs information processing within each local spacetime region, and the second is that globally, these regions are connected according to a definite causal order. In the PMF, these regions are modeled as laboratories, each receiving a quantum system, performing an operation, and producing an output system. A *process matrix* is an algebraic object describing the correlations and possible information flow between these laboratories, without imposing a definite causal order on one hand, but also not violating local quantum mechanics.

Let us explain what are process matrices in the language of computer science. A joint probability distribution describes correlations between classical registers, while a joint density matrix describes quantum correlations between quantum registers, including entanglement. These correlations, classical or quantum, are between physical *systems*. Process matrices extend this to correlations between quantum *operations*. To elaborate, an ordinary quantum circuit implements a completely positive trace-preserving (CPTP) map from input registers to output registers. A joint density matrix, in contrast, can be thought of as a snapshot of registers occupying different spatial regions: it specifies their joint state, but not their evolution. This motivates viewing density matrices as describing quantum correlations between systems across space at a given time, and process matrices as generalized states describing correlations between operations across space and time.

Both density matrices and quantum channels arise naturally as special cases of process matrices. When the laboratories receive quantum systems but send none out, the process matrix reduces to a joint density matrix describing their input systems. When one laboratory's output is sent to another laboratory's input, an ordinary quantum channel describes this connection. The PMF includes both situations, while also allowing correlations between local operations that cannot be explained by a definite global causal order [OCB12].

A bit more formally, consider local laboratories applying quantum channels $\mathcal{A}_1, \ldots, \mathcal{A}_k$. A process matrix W describes a transformation from the local operations to a global operation $\mathcal{C}_{\mathcal{A}_1,\ldots,\mathcal{A}_k}$. This transformation depends linearly on each local operation. We require $\mathcal{C}_{\mathcal{A}_1,\ldots,\mathcal{A}_k}$ to be a quantum channel for *every* choice of the local channels, including operations involving auxiliary registers whose inputs may be entangled across laboratories or with an external input [CDPV13, AFNB17]. This requirement allows us to describe how local quantum operations are *combined*, without either thinking of specific channels or prescribing a definite causal order between them.

Purification and physicality. The evolution of a closed quantum system is unitary, and hence preserves information. This reversibility is a fundamental feature of quantum mechanics. General quantum channels need not be reversible, but that is only since they may describe a part of a larger system. Namely, any quantum channel can be implemented by initializing auxiliary registers, applying a unitary transformation on the larger system, and discarding some output registers – this process is called purification. Thus, information that appears to be lost may remain in registers that are no longer observed. The PMF as was initially defined by [OCB12] does not guarantee an analogous reversibility for transformations corresponding to process matrices.

Araújo, Feix, Navascués and Brukner [AFNB17] later proposed a *purification postulate* to impose such a requirement. Specifically, we call a process *pure* if every choice of local unitary operations $\mathcal{A}_1, \ldots, \mathcal{A}_k$, including operations involving auxiliary registers, induces a global unitary transformation $U_{\mathcal{A}_1,\ldots,\mathcal{A}_k}$. A process is called *purifiable* if it can be obtained from a larger pure process by fixing additional inputs to pure states and discarding additional outputs. The postulate requires physically realizable processes to be purifiable. This is indeed a nontrivial restriction: the original process of [OCB12], whose correlations cannot arise under any definite causal order, even allowing classical uncertainty about that order, is not purifiable [AFNB17].

From process matrices to computation. Indefinite causal order already offers advantages in information processing tasks. The quantum switch [CDPV13], which coherently controls the order of two operations, already provides advantages using a pure process [AFNB17]. For example, it can determine with certainty whether two unknown unitary gates commute or anticommute using each only once, which is impossible for fixed-order circuits under the same query restriction [Chi12]. It also enables protocols that use exponentially less communication than one-way quantum protocols [GFAB16], and allows two channels that individually erase all input information to transmit classical information when applied in a superposition of orders [ESC18]. These results do not settle whether pure, or even general process matrices, enable efficient computation beyond **BQP**.

Araújo, Guérin and Baumeler [AGB17] formalize process matrix computation using uniformly generated, polynomial-size ordinary quantum circuits C with designated output registers connected back to input registers. In this work we call such a circuit, together with its designated connections, a *process matrix generator* when the connected transformation is a quantum channel for every choice of local channels on the individual connections, including auxiliary registers as above. The computational model grants access to the resulting channel, with cost measured by the size of C.

This computational model was shown [AGB17] to admit a description in terms of a subclass of *postselection closed timelike curves* (CTCs), represented using postselected teleportation [LMGP$^+$11a]. To elaborate, general postselection can introduce an input-dependent normalization, making the induced evolution nonlinear. In contrast, process matrix validity imposes a strong restriction: the postselection probability must be independent of both the input state and the local operations, preserving linearity. Thus, [AGB17] define the class corresponding to polynomial-time process matrix computation as $\mathbf{BQP}_{\ell\mathrm{CTC}}$, standing for **BQP** with access to linear postselection CTCS. Importantly, $\mathbf{BQP}_{\ell\mathrm{CTC}}$ corresponds to process matrix computation *without* asking for the purification restriction. Pure PMGs (Definition 4.13) further restrict this to *unitary* CTCs: the generator is unitary, and so is the connected evolution for every choice of local unitary operations.

An efficient process matrix generator circuit gives a short description of a process, but the computation obtained by its process matrix may be much harder to implement. Postselected teleportation illustrates this difficulty: its success probability can be exponentially small, even when the connected channel is unitary. Araújo, Guérin and Baumeler [AGB17] showed that $\mathbf{BQP}_{\ell\mathrm{CTC}}$ lies between **BQP** and **PP**, leaving open whether the first inclusion is strict. We explore this question, including under the further requirement that the process itself be pure: can indefinite causal structure enable computation beyond **BQP** while preserving the reversibility of quantum mechanics?

1.1 Results

We investigate quantum computation with indefinite causal structure, considering both purifiable processes that are not pure, and the further restriction to pure processes. Our work has two main parts.

1. First, we show that polynomial-time process computation can solve every problem in **NP**. Our constructed process is furthermore purifiable (Definition 4.17), but not pure (Definition 4.13).

2. Second, we give a polynomial-time algorithm for the unique-sink orientation problem, using pure processes whose unitary generators are constructed explicitly and efficiently.

Hardy [Har09, Section 4.3] asked whether indefinite causal structure can enable efficient computation beyond **BQP**. Within the process matrix framework, our results answer this question affirmatively, even when restricted to pure processes, assuming that USO has no quantum polynomial-time algorithm. These results also clarify the role of linearity in postselection closed timelike curves, and distinguish between purification as a proposed condition for physicality, and a potential condition for efficient realization.

As part of our work we also extend the computational framework of [AGB17]. For a promise problem, the generating circuit constructed from the input may be guaranteed to define a valid process only when the promise holds. Our definition also covers inputs that do not satisfy the promise: if the circuit does not define a valid process, the algorithm is given access to an arbitrary quantum channel, and must remain correct for any such channel. On promised inputs, the computation is unchanged. We give the formal definitions in Section 4.

Part I: Linear CTCs, purification, and NP. The computational power of postselection CTCs has been closely associated with their nonlinear evolution [LMGP+11a, Section V], and quantum computation with unrestricted postselection CTCs has exactly the power of **PP** [Aar05]. The restriction to linear CTCs was motivated, in part, by the possibility that removing nonlinearity would exclude such computational power [AGB17, Section III]. We show that linearity alone does not provide this restriction.

Theorem 1.1. *Let* $\boldsymbol{BQP}_{\ell CTC}$ *as in Definition 4.10. Then,*

$$\boldsymbol{NP} \subseteq \boldsymbol{BQP}_{\ell CTC} \ .$$

The containment continues to hold even given two more conditions: the valid generation queries are required to describe purifiable processes (Definition 4.17) whose generated channels (Definition 4.8) are unitary.

In our construction, even though the circuit generator is purifiable and the generated channel is unitary, the circuit generating the process is *not* unitary, which makes the process not pure (Definition 4.13). We discuss how this can be interpreted in our Technical Overview (see Section 2.2).

The main construction solves unique-witness search. Given a Boolean circuit $v : \{0,1\}^n \to \{0,1\}$ with exactly one satisfying input $z \in \{0,1\}^n$, we construct in classical polynomial time a PMG whose generated channel swaps between $|0\rangle$ and $|z\rangle$ and fixes the other computational basis states. Thus, one use of the generated channel obtains z. To solve all of **NP**, where the input circuit $v' : \{0,1\}^n \to \{0,1\}$ does not necessarily have exactly one witness, we use the Valiant-Vazirani reduction [VV85] and verify every candidate witness before accepting.

We also show how to use general process computation for coherent permutation inversion. Given a classical circuit computing a permutation $\pi : \{0,1\}^n \to \{0,1\}^n$, we efficiently construct a PMG whose generated channel implements $|y\rangle \mapsto |\pi^{-1}(y)\rangle$, including on superpositions and inputs entangled with other registers. The unique-witness and permutation-inversion constructions are given in Section 5.

Part II: Pure processes solve unique-sink orientation. We next require the generating circuits themselves to be unitary and the resulting processes to be pure and polynomial-time. We show an algorithm

in this framework solving the unique-sink orientation problem (USO), a well-studied problem arising in optimization [SW01]. An instance specifies the directions of the edges of an n-dimensional Boolean cube, with the promise that every face has exactly one sink[1]. The orientation is given succinctly by a Boolean circuit that, on input a vertex, outputs the directions of its outgoing edges. The task is to find the sink of the entire cube. The most efficient known quantum algorithms for USO take $2^{\Omega(n)}$ time [Bac17]. We prove the following.

Theorem 1.2 (Informal; See Theorem 6.2)**.** *The problem* USO *(Definition 6.1) can be solved in polynomial time using pure process computation (Definition 4.15). The algorithm uses the generated channel once and succeeds with probability* 1.

More specifically, our construction generate access to a coherent inversion of the USO function. Write $s(x)$ for the string whose i-th bit indicates whether the edge in direction i leaves x. The USO property implies that s is a permutation (Lemma 6.1), and our generated unitary implements

$$T_s \cdot |y\rangle = |s^{-1}(y)\rangle \ .$$

The sink is obtained by setting $y = 0^n$.

Unique-sink orientation and related problems. The study of USO connects decades of work in optimization, game theory, and formal verification [SW01, JS08]. Classical polynomial-time reductions to USO cover rational P-matrix linear complementarity, simple stochastic games, discounted and mean-payoff games, parity games, one-dimensional energy games, and full modal μ-calculus model checking [GR05, JS08, FJS09, BCD+11, Wil01]. These connections give the apparently simple task of finding a sink several different meanings. For example, stochastic games have been used to design controllers for autonomous transport and energy management [KPW18]. In cybersecurity, this includes finding optimal defensive strategies to preserve service availability against an adversary seeking to disrupt it [DKSS14]. Model checking makes it possible to verify properties of software through a formal model, even when that software's behavior is too complex to understand by inspecting its code [CGK+18]. For these problems, general polynomial-time solvability remains open in both classical and quantum computation.

Technical contributions: Constructing valid process matrices. Beyond the generating circuits themselves, our two main results introduce techniques in quantum information theory for proving process validity. For general processes, we show how an ordinary quantum circuit preparing the Choi state of an isometry can be turned into a process matrix generator implementing that isometry (Lemma 5.1). This gives the construction underlying our **NP** result. For pure processes, our statement is slightly more general than the USO algorithm. We construct explicit polynomial-size pure process matrix generators for a family of quantum changes of basis. Each basis vector is obtained from a computational-basis state $|x\rangle$ by applying *single-qubit* unitaries, each chosen between two options. The binary choices depend efficiently on x, and furthermore satisfy a combinatorial condition that guarantees process validity (Theorem 6.1). This combinatorial condition is exactly a characterization of unique-sink orientations, which is how the connection to USO comes up in the first place.

Directions for physical implementation. We also relate the structure of our pure generators to existing circuit simulations and quantum-switch implementation proposals (Section 6.2). In particular, our construction includes a product basis for which a quantum-switch measurement protocol and a photonic implementation proposal are known. Efficient physical realization for arbitrary USO instances remains open.

[1] Note that the overall number of faces is $\sum_{k\in\{0,1,\ldots,n\}}(\#\text{ of } k \text{ dimensional faces}) = \sum_{k\in\{0,1,\ldots,n\}} \binom{n}{k} 2^{n-k} = 3^n$.

1.2 Related Work

Process matrices and information processing. The process matrix formalism was introduced by Oreshkov, Costa and Brukner [OCB12] to describe quantum correlations without assuming a definite global causal order. A sequence of works explored the information-processing advantages available in this setting. Chiribella [Chi12], Araújo, Costa and Brukner [ACB14], and Renner and Brukner [RB22] demonstrated query advantages for tasks involving unknown quantum operations. In communication, Guérin et al. [GFAB16] gave a protocol using exponentially less communication than any one-way quantum protocol for the same task. Other communication advantages include transmitting classical information through completely depolarizing channels and perfect quantum communication using certain channels with zero quantum capacity [ESC18, CBB+21]. Applications also include quantum metrology and refrigeration [ZYC20, FV20]. More recently, Abbott, Mhalla and Pocreau [AMP24, AMP25] studied Boolean-function query complexity using general processes, obtaining advantages in both the error achievable with a fixed number of queries and exact query complexity. Bavaresco et al. [BKM+25] proved an exponential query lower bound for simulating the quantum switch: for arbitrary unknown n-qubit channels, a deterministic exact simulation that calls one channel only once must call the other exponentially many times. These query results concern access to unknown operations. For computation with explicitly described circuits, Araújo, Guérin and Baumeler [AGB17] introduced the computational framework on which we build. They also constructed processes that violate causal inequalities but admit ordinary quantum-circuit simulations with constant overhead. Finally, Araújo, Feix, Navascués and Brukner [AFNB17] defined pure and purifiable processes and proposed purifiability as a necessary condition for physical realizability.

Computational models of closed timelike curves. Deutsch [Deu91] introduced a model of CTCs in which the state travelling around a loop must be a fixed point of its evolution.[2] Aaronson and Watrous [AW09] characterized the power of polynomial-time classical and quantum computation in this model as **PSPACE**. Another model uses postselected teleportation to connect a system's output back to its input [LMGP+11b, LMGP+11a].[3] Results on quantum computation with postselection and its implementation using CTCs establish exactly the power of **PP** for unrestricted postselection CTCs [Aar05, BW12]. Both models generally induce nonlinear evolution.

Logical consistency and reversibility. Baumeler and Wolf [BW17, BW18] studied classical circuits with feedback under the requirement that, for each input, there is exactly one assignment of values to the wires consistent with the gates. They characterized the decision problems solvable in their polynomial-time non-causal circuit model as $\mathbf{UP} \cap \mathbf{coUP}$. They also obtained this class as an upper bound for classical deterministic processes, where consistency must hold under every choice of local operations. Baumeler et al. [BCR+19] further showed that every valid deterministic classical process admits a reversible extension. They distinguished classical reversibility from validity under arbitrary local quantum operations, which requires a separate argument for a pure quantum process.

Experimental evidence and physical realizability. The physical realizability of general process matrix computation is unknown. Even for the standard quantum circuit model, constructing a scalable quantum computer remains an outstanding challenge. Nevertheless, several experiments support indefinite causal order between quantum operations. Photonic implementations of the quantum switch have been certified against classical mixtures of definite orders [RRF+17, GGK+18], and its full process matrix has been reconstructed [AQWR24]. More recent experiments reported violations of Bell-like inequalities for causal order [RAC+26, GTW+26, QLX+26]. These conclusions rely on assumptions about the devices and their spacetime arrangement, and experimental loopholes may remain. The experiments concern the quantum switch and do not establish the physical realizability of general or even pure process matrices.

[2] These are usually referred to as Deutschian CTCs in the literature.

[3] These are usually referred to as postselection CTCs in the literature.

2 Technical Overview

In this section we describe the main ideas behind our results. Before technical details, we recall how to formalize computation with process matrices.

How to compute with process matrices. In the introduction we described a process matrix as specifying quantum correlations between local quantum operations. To use such a process for efficient computation, we need a *succinct* description of it. In our model, this description is an ordinary quantum circuit. For ease of presentation, we first consider unitary circuits, although the same approach applies to general quantum circuits.

Consider a unitary circuit C acting on $(n+r)$ qubits. We call its first n single-qubit input and output registers the *feedback registers*, denoted $R_1, \ldots, R_n$, and denote the remaining registers by $Q_1, \ldots, Q_r$. For the moment, think of C as a static diagram, rather than a computation being executed. Its gates and wires describe connections between the input and output registers. We will use this entire diagram to specify how local operations at the feedback registers are connected to one another and to the remaining registers.

How to prove validity. At this point we can consider the interpretation in terms of n laboratories. The circuit C (or more precisely the pair $D := (n, C)$) defines the correlations between the laboratories. The PMF says that if D is a PMG, then whatever operations performed in the laboratories, without causal order, the rest of the system will continue to behave like a quantum channel.

Let us explain it more carefully. Each laboratory performs an operation $\mathcal{A}_i$ at one of the feedback registers. $\mathcal{A}_i$ receives the *output* of R_i and returns a system to its *input*. Furthermore allow each $\mathcal{A}_i$ to act also on an auxiliary input E_i and produce an auxiliary output E_i'. Connect the output of each feedback register back to its input through $\mathcal{A}_i$, leaving $Q := Q_1, \ldots, Q_r$ and all auxiliary registers external. The resulting transformation, denoted $\mathcal{C}_{\mathcal{A}_1,\ldots,\mathcal{A}_n}$, takes an input on $Q, E_1, \ldots, E_n$ and produces an output on $Q, E_1', \ldots, E_n'$. We require it to be a quantum channel for *every* choice of local quantum channels and auxiliary registers. In particular, the inputs may be entangled across laboratories, with Q, or with an external reference. As said, with $D = (n, C)$ satisfies this requirement we call it *process matrix generator* (PMG, Definition 4.8).

The generated channel. For the computation itself, we set all the local operations to the identity and use no auxiliary registers. Each feedback output is then connected directly to its corresponding input, leaving a transformation on Q. We call this the *generated channel* and denote it by $\mathbf{PMO}_D$.

For a unitary circuit C, there is a simple expression for this connection (see Lemma 4.1 for formal statement). Writing U_C for its unitary matrix and $R := R_1, \ldots, R_n$, we have

$$T_D := Tr_R(U_C)\ , \qquad \mathbf{PMO}_D(\rho) = T_D \rho T_D^\dagger\ .$$

The partial trace here acts on the matrix U_C itself: it connects the feedback input and output indices. It does *not* mean executing C and then discarding the feedback registers from the resulting state. For a unitary C, if D is a valid PMG, then T_D is unitary. In fact, a valid PMG with a unitary generating circuit is pure (Definition 4.13): every choice of local operations that are guaranteed to be unitary, including operations on auxiliary registers, gives a global unitary transformation.

For a valid PMG D, the model gives us access to $\mathbf{PMO}_D$, with cost measured by the size of C. This is where the opportunity for a jump in computational power: a small generating circuit can describe a channel that may be difficult to implement by an ordinary quantum circuit. Our constructions therefore have two parts. We design a circuit whose feedback connections give the transformation we want, and then prove that the circuit remains valid for every choice of local operations.

2.1 General process matrix computation and NP

The main quantum-information result behind our **NP** algorithm is a construction that turns a circuit preparing a Choi state into a process matrix generator for the corresponding channel. We first explain this construction, and then show how to use it to solve **NP**.

For an n-qubit unitary V, its Choi state is

$$|J_V\rangle := \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle \otimes V|x\rangle \ .$$

One can think of this state as a superposition of inputs together with their corresponding outputs. The intuition is that a circuit P_V that generates $|J_V\rangle$ gives a succinct description of the channel V, however, this description is "not accessible" – it is unclear how to execute the action of V, using P_V. Here, we show that general process matrices allow this seemingly inaccessible information to be extracted. Our construction gives the following statement, presented here for unitaries.

Theorem 2.1 (Informal; see Lemma 5.1)**.** *Given a quantum circuit preparing $|J_V\rangle$, there is a classical polynomial-time algorithm that constructs a PMG whose generated channel applies V.*

From a Choi state to a channel. Let $|\psi\rangle$ an n-qubit state and recall what happens when we use $|J_V\rangle$ for teleportation. For each $i \in [n]$, we perform a Bell-basis measurement on the i-th input qubit and the i-th qubit of the first half of $|J_V\rangle$. Together, these measurements give a uniformly random outcome $j \in \{0,1\}^{2n}$. The remaining register (what used to be the second part of the register containing $|J_V\rangle$) contains $V \cdot P_j|\psi\rangle$, where P_j is the Pauli operator associated with j and $P_{0^{2n}} = I$. Thus, the outcome $j = 0^{2n}$ gives exactly what we want, but its probability is only 2^{-2n}.

We construct a generating circuit C with $2n$ single-qubit feedback input registers and n external input qubits containing the state to which we want to apply V. Its outputs likewise consist of $2n$ feedback qubits and n external qubits. The idea is to arrange the feedback connections so that only the teleportation outcome $j = 0^{2n}$ exists in the generated channel.

The circuit first measures its feedback input registers in the computational basis, obtaining $r \in \{0,1\}^{2n}$. It then prepares $|J_V\rangle$ in fresh auxiliary registers and performs the teleportation procedure on the external input and the first half of $|J_V\rangle$, obtaining $j \in \{0,1\}^{2n}$. The circuit then swaps the second half of $|J_V\rangle$ into the external registers, which now contain $V \cdot P_j|\psi\rangle$, and sets the feedback output to the computational-basis state $|r \oplus j\rangle$. The remaining registers are discarded. When each feedback output is connected directly to its input, the two strings must agree. This implies

$$r = r \oplus j \quad \Longleftrightarrow \quad j = 0^{2n} \ .$$

If j is nonzero, at least one feedback bit is flipped, so that branch cannot contribute to the connected transformation. If $j = 0^{2n}$, every value of r is allowed, and the operation on the external input is V.

The normalization also works out exactly. Each value of r contributes the channel $\rho \mapsto V\rho V^\dagger$ with weight 2^{-2n}, and there are 2^{2n} possible values of r. Adding these weights gives the desired channel, without any input-dependent normalization.

Why this is a valid process. So far, we have connected each feedback output directly to its input, in other words, we know that *if* this is a valid process matrix, it will give us the channel we want. The main technical part is proving that C correctly describes a quantum correlation in the PMF, namely, that its derived operator is a valid process matrix. Formally, we need to argue that the transformation must remain a quantum channel when arbitrary local operations are inserted into the feedback slots. Our construction actually satisfies a stronger condition. We can allow a *single* quantum channel to act jointly on all the feedback qubits, together with arbitrary auxiliary registers. As we explain later, this stronger

validity property also guarantees that our process is purifiable. Let $\mathcal{A}$ be an arbitrary quantum channel acting on the $2n$ feedback qubits, possibly together with auxiliary registers.

We do not go into detail about the full proof, but in a nutshell, explicitly measuring the feedback registers inside the generating circuit C helps is the key to our validity proof.[4] The measurement is used in the following way: different measurement outcomes contribute as separate branches, rather than interfering with one another. In a branch with teleportation outcome j, the external input undergoes the unitary $V \cdot P_j$, while the inserted channel $\mathcal{A}$ determines the operation on the auxiliary registers.[5] When checking trace preservation, the unitary $V \cdot P_j$ cancels with its adjoint. We are therefore left with showing that the operations induced by $\mathcal{A}$ on the auxiliary registers, summed over all measurement branches with their corresponding weights, preserve the trace. We show this using the fact that $\mathcal{A}$ is a valid quantum channel and thus trace-preserving.

To see why the sum over branches preserves the trace, write the feedback output as $a = r \oplus j$. Summing over (r, j) is the same as summing over all pairs (r, a). For each fixed input a to the inserted channel $\mathcal{A}$, we are therefore summing over all its possible returned values r and all its branches. Since $\mathcal{A}$ is trace-preserving, the corresponding Kraus operators satisfy the completeness relation: the sum of each operator's adjoint times itself is the identity on the auxiliary input. Finally, summing over the 2^{2n} choices of a cancels the teleportation factor 2^{-2n}. This proves validity, including on entangled inputs. Lemma 5.1 gives the full calculation.

Using the construction to solve NP. We first recall that the Valiant–Vazirani reduction [VV85] transforms a satisfiable Boolean circuit into a different formula with exactly one satisfying input, with inverse-polynomial probability. It also preserves non-satisfiability of formulas (see Theorem 3.1). This suggests our approach to solving **NP**, by solving SAT: assume momentarily we could take care of unique-witness finding. Then, given a SAT formula, repeat the VV reduction polynomially many times independently, use our process matrix algorithm to search for a witness in each resulting instance, and verify every returned candidate witness against the VV-reduced formula. If our unique-witness finder works, then whenever *any* reduced instance has a unique witness, our construction will find it. On the remaining instances, whatever channel is returned, an incorrect candidate cannot pass the final classical check.

We therefore focus on unique-witness search. We are given a Boolean circuit $v : \{0,1\}^n \to \{0,1\}$ with exactly one satisfying input z, and would like to find z. For simplicity, assume that $z \neq 0^n$; we can ensure this by adding one input bit and requiring it to be 1 for verification to pass. Next, consider the unitary S_z that exchanges $|0^n\rangle$ and $|z\rangle$, and fixes every other computational-basis state. If we had access to S_z, finding the witness would be immediate by applying it to $|0^n\rangle$. However, the direct way to construct a circuit for S_z uses the very string z we are trying to find. The theorem above gives us another approach: prepare its Choi state using only the verifier v, and turn this preparation into a PMG whose generated channel applies S_z.

Generating the Choi state of S_z. For S_z, the Choi state is

$$|J_{S_z}\rangle := \frac{1}{\sqrt{2^n}} \left(\sum_{x \notin \{0^n, z\}} |x, x\rangle + |0^n, z\rangle + |z, 0^n\rangle \right) .$$

Almost all terms have the simple form $|x, x\rangle$. Only two terms are different, and we can recognize when to produce them by checking $v(x)$.

Start with a uniform superposition over pairs $(x, b) \in \{0,1\}^n \times \{0,1\}$. If $x \neq 0^n$, $v(x) = 0$, and $b = 0$, write (x, x) into an output register. If $v(x) = 1$, write $(0^n, x)$ when $b = 0$, and $(x, 0^n)$ when

[4] We later discuss whether the central role of measurement in this proof points to an artificiality of the model, when generating circuits are allowed to be non-unitary.

[5] Recall that $\mathcal{A}$ acts on both the feedback registers and its auxiliary registers. Once the feedback connections are closed, its auxiliary registers remain part of the external input and output.

$b = 1$. Conceptually, we mark these branches as successful. There are overall 2^{n+1} branches and 2^n successful branches: one for each $x \notin \{0^n, z\}$, and two for $x = z$. They produce exactly the terms of $|J_{S_z}\rangle$, with equal amplitudes.

Condition on x	b	First output	Second output
$x \neq 0^n$ and $v(x) = 0$	0	x	x
$v(x) = 1$	0	0^n	x
$v(x) = 1$	1	x	0^n

Observe that, on the successful branches, we have the following normalized state:

$$\frac{1}{\sqrt{2^n}} \left(\sum_{x \notin \{0^n, z\}} |x, 0\rangle \otimes |x, x\rangle + |z, 0\rangle \otimes |0^n, z\rangle + |z, 1\rangle \otimes |z, 0^n\rangle \right) .$$

In each term, the first ket contains the original registers (x, b), and the second contains the two output registers. The outputs contain the pairs we want, but they are still entangled with (x, b). We therefore need to uncompute these original registers. The point is that on successful branches this is easy: the output tells us what (x, b) was. If its first component is zero, then $b = 0$ and its second component is x; if its second component is zero, then $b = 1$ and its first component is x; otherwise, $b = 0$ and both components equal x. We can therefore uncompute (x, b) and the work registers, leaving the outputs in the state $|J_{S_z}\rangle$ on the successful branches. We showed how to uncompute information only once we are at the state of successful branches. However, the successful branches take exactly $1/2$, independently of z, and as we show in the body of the paper a constant number of amplitude-amplification operations makes the preparation exact iwth probability 1; see Section 5.2.

Notice that preparing this state does not by itself give an efficient way to find z. Measuring it in the computational basis reveals z only in the two exceptional terms, whose total probability is $2/2^n$. However, our general construction turns this state preparation into a PMG implementing S_z. This gives the unique-witness solver needed for the Valiant–Vazirani reduction, completing the algorithm for **NP**.

Inverting a permutation. The same Choi state transformation from Lemma 5.1 also gives coherent permutation inversion. Suppose we are given a classical circuit computing a permutation $\pi : \{0,1\}^n \to \{0,1\}^n$, then we can efficiently prepare

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |\pi(x), x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} |y, \pi^{-1}(y)\rangle = |J_{\pi^{-1}}\rangle$$

simply by evaluating π in superposition followed by swapping the input and output registers. The construction above turns this state preparation into a PMG whose generated channel applies π^{-1}, including on superpositions and inputs entangled with other registers (see Section 5.4).

2.2 Purifiable versus pure processes

Every process obtained by our Choi-state construction from Section 2.1 is actually purifiable. Note that it is purifiable (Definition 4.17) but not pure (Definition 4.13), because its generating circuit is not unitary. To establish purifiability, we need to show that the process has a pure extension, that is, a larger unitary transformation that remains a valid process under every choice of local operations, and that recovers the original process when we initialize its additional inputs and discard its additional outputs. As we show in Proposition 5.1, the reason for its purifiability is that validity holds even when a single channel $\mathcal{A}$ acts *jointly* on all $2n$ feedback qubits – this is a substantially stronger guarantee than validity holding against disjoint spacetime regions. We can therefore regard the entire process as a transformation that takes one

quantum channel $\mathcal{A}$ and produces another, $\mathcal{C}_{\mathcal{A}}$. We combine this fact with Theorem 1 of [CDP08] to establish the existence of a pure extension.

Is there an efficiency gap between purifiable and pure processes? There are two details to observe in what we've seen in this section so far. The first is that in the Choi-state transformation (Lemma 5.1) and its derived constructions for **NP** and permutation inversion, measurements are instrumental to our proof. The measurements make the proof work, but in our construction they also make the process non-pure. The second thing to observe is that the purification argument above does not tell us how to find the purification *efficiently*. This suggests a possible gap that does not exist in ordinary quantum computation. Specifically, finding a purification of a given quantum circuit is always easy, and can be done by a classical deterministic polynomial-time algorithm. Here, we do not know whether that is the case. Concretely: given a purifiable PMG that admits a polynomial-size pure PMG extension, is there a polynomial-time procedure, classical or quantum, that finds such an extension? Note, for example, that a straightforward purification procedure applied to the generating circuit from Lemma 5.1 gives a unitary quantum circuit, but the resulting circuit is no a PMG anymore: on some inputs to the additional registers introduced by purification, the feedback connections induce a transformation that is not trace-preserving.

A different question concerns physical realizability, which we are less equipped to answer. Here we are asking about efficient physical realization, rather than existence alone: in a theory that assumes pure processes are physical and have computational cost proportional to the generating circuit's size, is the same true for purifiable processes that are not pure? Concretely, in such a theory, assume we have a purifiable but non-pure PMG. Can one efficiently obtain access to its generated channel without first finding its purification? This question is currently difficult to answer, as we do not even know whether all pure processes are physical.

Remark 2.1 (An example of a pure extension that uses the witness). Consider our unique-witness construction. We only need the Boolean verifier v for it and the witness z is of course unknown. From v, we efficiently construct a circuit P_V that prepares $|J_{S_z}\rangle$, and use P_V inside our PMG C. This circuit prepares the Choi state, performs the teleportation measurements, and sets the feedback outputs according to their outcomes. None of these steps requires knowing z, yet the generated channel (after tracing out the feedback registers) applies S_z and therefore computes z on input $|0^n\rangle$.

Proposition 5.1 guarantees that this PMG has a pure extension. So, what happens if we apply this argument to our concrete PMG arising solving unique witness search? For this particular construction, the extension can be chosen to have a polynomial-size generating circuit that uses S_z once, together with elementary unitary operations. We leave verification of this construction to the reader. However, to write down the circuit implementing S_z, our construction uses z explicitly: its gates implement the exchange of $|0^n\rangle$ and $|z\rangle$. Thus, given z, we can efficiently construct and execute the unitary generating circuit for the pure extension. But the whole point of the model is to find z given v. The original generator is efficiently constructed from v alone, whereas constructing this particular description of its pure extension first requires the very witness we are trying to find.

This proves that a polynomial-size pure extension exists for every unique-witness instance. It does not give an efficient procedure for finding that extension from v, and whether a different construction can do so remains open. The purification postulate requires existence; it does not impose this computational requirement. In our next result, we construct the pure generators themselves explicitly and efficiently from the input.

2.3 Pure processes and unique-sink orientation

We now construct the unitary generating circuit directly from the input, and prove that it describes a pure process. Our goal is to solve unique-sink orientation (USO), defined formally in 6.1. We will approach this through permutation inversion again, however the technical chronology here will be different. We'll

observe that from a relatively simple idea, the description of USO arises naturally. As it turns out, the condition that makes our validity proof work is exactly a characterization of unique-sink orientations.

Idea: try to invert a permutation unitarily. Suppose we are given a classical circuit computing a permutation $\pi : \{0,1\}^n \to \{0,1\}^n$, and would like to apply its inverse. Here's an idea for inverting it – first just thinking of the generated channel, without thinking whether circuit is a valid PMG. Define

$$h(x) := x \oplus \pi(x) \quad .$$

We construct a unitary circuit C on $2n$ qubits, with n single-qubit feedback registers $R_1, \ldots, R_n$, collectively denoted by R, and an n-qubit external register Q.

Let us first describe the ordinary computation of C. On input $|x\rangle_R \otimes |y\rangle_Q$, compute $h(x)$ reversibly into fresh work registers, and add it bitwise modulo 2 to Q. The external register now contains $y \oplus h(x)$. Since R still contains x, we can reverse the computation of h and return the work registers to zero. Finally, swap R and Q. The resulting unitary acts as

$$U_C\left(|x\rangle_R \otimes |y\rangle_Q\right) = |y \oplus h(x)\rangle_R \otimes |x\rangle_Q \quad .$$

This circuit is efficiently constructed from the circuit for π. In particular, all we used was the ability to evaluate π in the forward direction.

Now connect each feedback output directly to its input. Recall that this connection takes the partial trace of the matrix U_C (see our simplification Lemma 4.1). A term contributes only when the feedback input x equals the feedback output $y \oplus h(x)$. By our choice of h,

$$x = y \oplus h(x) \quad \iff \quad y = \pi(x) \quad .$$

For each external input y, exactly one string x satisfies this equation, namely $x = \pi^{-1}(y)$. This is also the string placed in the external output. Therefore,

$$Tr_R\left(U_C\right)|y\rangle = |\pi^{-1}(y)\rangle \quad .$$

By linearity, the same transformation inverts π coherently on superpositions.

So far, this looks like a way to invert any efficiently computable permutation using a pure process. But we have not yet proved that our circuit describes a valid quantum correlation between operations, namely, whether it is a PMG. We only know that the identity feedback connections give a unitary transformation. We still need to show that arbitrary local operations can be inserted into these connections without making the resulting transformation invalid. This is where we will need a restriction on π.

What makes the circuit describe a valid quantum correlation? We begin with local unitaries that have no auxiliary registers. Let V_i be an arbitrary single-qubit unitary inserted into feedback pair i, and denote the feedback-connected transformation by $T_{\mathbf{V}}$, that is:

$$T_{\mathbf{V}} := Tr_R\left[\left(\left(\bigotimes_{i=1}^{n} V_i\right) \otimes I_{\mathcal{H}_Q}\right) U_C\right] \quad .$$

Our goal is to show that $T_{\mathbf{V}}$ is unitary for every choice of the V_i.

Fix a feedback input string x. Before the swap, the circuit applies $X^{h_i(x)}$ to qubit i of Q: it either leaves that qubit alone or flips it. After the swap, that qubit goes through V_i and must return to feedback input x_i. Its contribution is therefore the following row in the 2×2 matrix $V_i \cdot X^{h_i(x)}$,

$$\langle x_i | V_i \cdot X^{h_i(x)} \quad .$$

Since the circuit places x in the external output, the entire row of $T_{\mathbf{V}}$ corresponding to output x is

$$\langle x | T_{\mathbf{V}} = \bigotimes_{i=1}^{n} \langle x_i | V_i \cdot X^{h_i(x)} \quad .$$

Each factor is a row of a unitary matrix, so every full row has norm one. What remains is to show that different rows are orthogonal.

Take two strings $x \neq x'$. Because their rows are tensor products, it suffices to find one coordinate where the corresponding factors are orthogonal. Suppose that at some coordinate i,

$$x_i \neq x'_i \quad \text{and} \quad h_i(x) = h_i\left(x'\right) \quad .$$

At this coordinate, the circuit applies the same bit flip for both strings. We are therefore looking at different rows of the *same* unitary matrix $V_i \cdot X^{h_i(x)}$. These small rows are thus orthogonal, regardless of the choice of V_i. Consequently, the two full rows, which are in turn obtained by tensoring the small rows, are orthogonal as well.

This gives us the following sufficient condition:

$$x \neq x' \quad \Longrightarrow \quad \exists i \in [n]: \quad x_i \neq x'_i \text{ and } h_i(x) = h_i\left(x'\right) \quad .$$

In words, for every distinct strings, look at the coordinates where they differ – on one of these coordinates the circuit $h(\cdot)$ has to output the same bit on both inputs. That coordinate preserves orthogonality even after an arbitrary local unitary is inserted.

Where unique-sink orientation enters. Guided by the above condition on permutations, we look to the literature, and find that it has a combinatorial meaning. Recall that a unique-sink orientation [SW01] directs the edges of the Boolean cube so that every face, including the entire cube, has exactly one sink. Its outmap $s : \{0,1\}^n \to \{0,1\}^n$ records the outgoing edges: $s_i(x) = 1$ if the edge in direction i is going from x to the outside[6], and $s_i(x) = 0$ otherwise. The input to our algorithm is a Boolean circuit computing s, and the task is to find the vertex z with $s(z) = 0^n$.

A characterization of USOs states that an outmap s describes a unique-sink orientation if and only if

$$x \neq x' \quad \Longrightarrow \quad \exists i \in [n]: \quad x_i \neq x'_i \text{ and } s_i(x) \neq s_i\left(x'\right) \quad ;$$

see [SW01] and Lemma our 6.1. In particular, this implies that s is a permutation: distinct inputs always have distinct outputs.

We can therefore set $\pi = s$ in our construction. To see why this gives the condition needed for validity, recall that $h(x) = x \oplus s(x)$. At the coordinate supplied by the USO characterization, both x_i and $s_i(x)$ change when we move from x to x', so their XOR stays the same. Thus,

$$h_i(x) = h_i\left(x'\right) \quad .$$

Conversely, when $x_i \neq x'_i$, equality of these XORs implies $s_i(x) \neq s_i(x')$. The USO characterization is therefore exactly the condition we obtained from the unitarity calculation.

Including auxiliary registers. To finish proving purity, we must allow each V_i to act jointly on its feedback qubit *and* an arbitrary auxiliary register E_i. The inputs to these registers may also be entangled. We explain why the same orthogonality argument continues to work, by generalizing vectors to operators.

For a bit a, write

$$B_{i,a} := V_i\left(X^a \otimes I_{\mathcal{H}_{E_i}}\right) \quad .$$

Previously, fixing the input and output bits of the feedback qubit gave a scalar. Now it gives an operator on E_i. Denote this operator by $B_{i,a}^{u,v}$, where v is the feedback input bit and u is the feedback output bit. Unitarity of $B_{i,a}$ gives

$$\sum_{v \in \{0,1\}} B_{i,a}^{u,v}\left(B_{i,a}^{u',v}\right)^\dagger = \begin{cases} I_{\mathcal{H}_{E_i}}, & u = u' \\ 0, & u \neq u' \end{cases} .$$

[6]Formally, from x to $x \oplus e_i$, where $e_i \in \{0,1\}^n$ is the vector with all zeros except a single 1 in coordinate i.

This is the same statement about rows as before: each row has unit norm and different rows are orthogonal, except that the entries are now operators.

We use this identity to check that $T_{\mathbf{V}} \cdot T_{\mathbf{V}}^{\dagger} = I_{\mathcal{H}_{Q,E}}$ by a block-wise analysis. Fix two external output strings x, x' for the Q register, and consider the corresponding block of $T_{\mathbf{V}} \cdot T_{\mathbf{V}}^{\dagger}$, which is an operator on E. If $x = x'$, every coordinate contributes the identity on E_i. Otherwise, the USO condition gives a coordinate i with $x_i \neq x'_i$ and $h_i(x) = h_i(x')$. At that coordinate we multiply different block rows of the same unitary $B_{i,h_i(x)}$, obtaining zero. The entire off-diagonal block is therefore zero. This proves the unitarity condition from Definition 4.13, and hence our circuit is a pure PMG.

This concludes the pure construction for USO: given the circuit for a USO outmap s, construct C as above and use its generated channel, whose unitary acts as

$$T_s|y\rangle = |s^{-1}(y)\rangle \ .$$

On input $|0^n\rangle$, this gives the unique sink with probability 1. The generating circuit is constructed in classical polynomial time from the circuit for s, and the algorithm uses its generated channel once; see Theorem 6.2.

Beyond permutation inversion. Looking back at the proof, the bit flips were not essential. At each coordinate, what mattered was that equal values of h_i selected the same unitary. We can therefore replace the choices I, X by any two single-qubit unitaries $U_{i,0}, U_{i,1}$, given by unitary circuits. The generating circuit computes h, applies the selected gates to Q (that is, we execute $U_{i,h_i(x)}$ on register Q_i), uncomputes h, and swaps the registers, just as before. The same validity argument applies.

More explicitly, define

$$U(x) := \bigotimes_{i=1}^{n} U_{i,h_i(x)} \qquad \text{and} \qquad |b_x\rangle := U(x)^{\dagger}|x\rangle \ .$$

The condition on h guarantees that the vectors $|b_x\rangle$ form an orthonormal basis, and the generated unitary satisfies

$$T|b_x\rangle = |x\rangle \ .$$

This describes our overall general result, given in Theorem 6.1. The permutation construction is the case $U_{i,0} = I, U_{i,1} = X$, where $|b_x\rangle = |s(x)\rangle$.

3 Preliminaries

We use the following standard notions and notations from classical and quantum computation in this work.

- For $n \in \mathbb{N}$, define $[n] := \{1, 2, \cdots, n\}$. The set of all finite binary strings is $\{0,1\}^* := \bigcup_{n \geq 0} \{0,1\}^n$.
- Unless stated otherwise, $\log(\cdot)$ is the base-2 logarithm, $\|\cdot\|$ is the Euclidean norm of a vector, and $\oplus$ is bitwise addition modulo 2 (aka, bitwise XOR).
- For $n \geq 0$, the n-qubit state space is $\mathcal{S}(n) := \mathbb{C}^{2^n}$. In particular, it is known that $\mathcal{S}(n)$ is a 2^n-dimensional Hilbert space. A register is a copy of such a space with its specified computational basis. Registers of the same dimension may represent different systems.
- A pure quantum state is a unit vector $|\psi\rangle$ in $\mathcal{S}(n)$. A mixed quantum state is a matrix $\rho \in \mathbb{C}^{2^n \times 2^n}$ which is positive semidefinite and satisfies $Tr(\rho) = 1$. Positive semidefinite means that $\langle v|\rho|v\rangle \geq 0$ for every vector $|v\rangle$.

- For $n, m \in \mathbb{N}$, an n to m (general) quantum circuit is a quantum circuit C with input an n-qubit mixed state, can perform any quantum gate from the fixed set of single-qubit gates $\{H, T, T^\dagger\}$, the two-qubit quantum gates $\{CNOT, SWAP\}$ and three-qubit Toffoli gate Tof, execute measurement gates and add ancillary qubit registers containing $|0\rangle$, where each of the above operations costs 1 time unit. At the end of the execution, C outputs an m-qubit mixed state.
- For $n \in \mathbb{N}$, an n-qubit unitary circuit is an n to n quantum circuit such that there exists a unitary matrix $U_C \in \mathbb{C}^{2^n \times 2^n}$ such that for every n-qubit pure state $|\psi\rangle$ the action of C corresponds to the action of the unitary, that is, $C(|\psi\rangle\langle\psi|) = U_C \cdot |\psi\rangle\langle\psi| \cdot U_C^\dagger$.

Circuit conventions. The single-qubit gates in the fixed set are

$$H := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \qquad T := \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix},$$

and $T^\dagger$ is the adjoint of T. Measurements are in the computational basis. General quantum circuits may also discard registers and use measurement outcomes for classical control. A unitary circuit may use additional work registers, provided these registers are returned to $|0\rangle$ on every input to the stated unitary. The circuit description includes its register sizes and gate count, type and position, which sums up to $|C|$. Input and output registers with zero qubits are allowed, with $\mathcal{S}(0) = \mathbb{C}$ and empty tensor products equal to the one-dimensional identity. The definitions below use the same conventions for zero-qubit registers.

3.1 Computational Complexity Theory

Definition 3.1 (Promise Problem and Language)**.** *A promise problem $\prod := (\prod_{YES}, \prod_{NO})$ is a pair of disjoint sets $\prod_{YES}, \prod_{NO} \subseteq \{0,1\}^*$. It is a decision problem when $\prod_{YES} \cup \prod_{NO} = \{0,1\}^*$, and in this case its language is $\mathcal{L} := \prod_{YES}$.*

Definition 3.2 (Search Problem)**.** *A search problem is specified by a relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ and a promised input set $\mathcal{P}$. An algorithm solves the problem with probability p if, on every $x \in \mathcal{P}$, it outputs a string z satisfying $(x, z) \in \mathcal{R}$ with probability at least p. All relations used below have polynomially bounded output length.*

Definition 3.3 (The Complexity Class **BQP** and Promise **BQP**)**.** *The class **BQP** consists of languages decidable by a uniform polynomial-time quantum algorithm with probability at least $1 - 2^{-|x|}$ on every input. The analogous promise class requires correctness only on $\prod_{YES} \cup \prod_{NO}$.*

Definition 3.4 (The Complexity Class **UP**)**.** *A promise problem $\prod := (\prod_{YES}, \prod_{NO})$ is in **UP** if there is a deterministic polynomial-time verifier $V(x, z)$ and a polynomial p such that, for every $x \in \prod_{YES}$ there is a single $z \in \{0,1\}^{p(|x|)}$ such that $V(x, z) = 1$ and for $x \in \prod_{NO}$ there is no $z \in \{0,1\}^*$ such that $V(x, z) = 1$.*

Definition 3.5 (The Complexity Class **coUP**)**.** *A promise problem $\prod := (\prod_{YES}, \prod_{NO})$ is in **coUP** if there is a deterministic polynomial-time verifier $V(x, z)$ and a polynomial p such that, for every $x \in \prod_{YES}$ it follows that for every $z \in \{0,1\}^{p(|x|)}$ we have $V(x, z) = 1$ and for $x \in \prod_{NO}$ there is exactly one $z \in \{0,1\}^{p(|x|)}$ such that $V(x, z) = 0$.*

Definition 3.6 (The Complexity Class **TFUP**)**.** ***TFUP** is the class of search problems where the relations are polynomial-time verifiable, has polynomially bounded witnesses, and has exactly one witness on every input.*

The Satisfiability Problem SAT**.** We define the satisfiability problem SAT, which is known to be **NP**-complete [Kar10]. Recall that SAT is a decision problem and in the below definition, the input is an arbitrary binary string in $\{0,1\}^*$, which is then interpreted as a CNF formula φ. In case the input is not in the correct format to be interpreted as a CNF formula (which can be checked by a classical efficient algorithm) the input is counted as a $\prod_{\text{NO}}$ instance.

Definition 3.7 (The Satisfiability Problem)**.** *The Satisfiability problem, denoted* SAT *is a decision problem, where the input is a CNF formula φ with n input Boolean variables. The problem is defined as follows.*

- $\prod_{YES}$ *is the set of all (binary encodings of) CNF formulas that are satisfiable.*
- $\prod_{NO}$ *is* $\{0,1\}^* \setminus \prod_{YES}$.

The Valiant-Vazirani Theorem. We use the Valiant-Vazirani Theorem [VV85], which can be stated as follows.

Theorem 3.1 (Valiant-Vazirani Theorem)**.** *There exists a classical probabilistic polynomial-time Turing machine T_{VV} that gets as input a CNF formula φ that has input n Boolean variables, and outputs a CNF formula φ^* that has polynomially many (in n) Boolean variables, with the following guarantees.*

- *If* $\varphi \notin \mathsf{SAT}$ *then* $\Pr_{\varphi^* \leftarrow T_{VV}(\varphi)}[\varphi^* \notin \mathsf{SAT}] = 1$.
- *If* $\varphi \in \mathsf{SAT}$ *then* $\Pr_{\varphi^* \leftarrow T_{VV}(\varphi)}[\varphi^*$ *has exactly one satisfying assignment* $] \geq \Omega\left(\frac{1}{n}\right)$.

3.2 Quantum Information Theory

We define linear operators over Hilbert spaces. In the scope of this work we will always define Hilbert spaces as complex vector spaces, for different amounts of qubits[7].

Definition 3.8 (Set of linear operators between two vector spaces)**.** *Let A, B two complex vector spaces. The set $\mathcal{L}(A,B)$ is defined as the set of all linear transformations from A to B.*

- *In case $A = B$, we write $\mathcal{L}(A)$ for short.*
- *The identity map on A is denoted by $I_{\mathcal{L}(A)}$.*
- *In case the sets are Hilbert spaces (and in our case, state spaces), then the set of linear operators is the set of appropriate matrices. Formally, if there are $n, m \in \mathbb{N}$ such that $\mathcal{H}_0 = \mathcal{S}(n)$, $\mathcal{H}_1 = \mathcal{S}(m)$, then $\mathcal{L}(\mathcal{H}_0, \mathcal{H}_1) = \mathbb{C}^{2^m \times 2^n}$, and the action of a linear transformation is vector-matrix multiplication, where the matrix is the left multiplicand. In that case we denote $I_{\mathcal{L}(A)} := I_{\mathcal{H}_0} := I_n$ and $I_{\mathcal{L}(B)} := I_{\mathcal{H}_1} := I_m$.*

Definition 3.9 (Unitary matrix and isometry)**.** *Let $\mathcal{H}_0 = \mathcal{S}(n)$ and $\mathcal{H}_1 = \mathcal{S}(m)$. A matrix $U \in \mathcal{L}(\mathcal{H}_0)$ is unitary if $U^\dagger U = UU^\dagger = I_{\mathcal{L}(\mathcal{H}_0)}$. A matrix $V \in \mathcal{L}(\mathcal{H}_0, \mathcal{H}_1)$ is an isometry if $V^\dagger V = I_{\mathcal{L}(\mathcal{H}_0)}$; here the input and output spaces need not have the same dimension. We write Ad_V for the linear map $\mathrm{Ad}_V(X) := VXV^\dagger$.*

An operator which we will use extensively in this work is taking the partial trace of a square matrix.

[7] More precisely, we think of Hilbert spaces (with the notation $\mathcal{H}_A$) as a variable that takes a value $\mathcal{S}(n)$ for some $n \in \mathbb{N}$. This in particular means that we can have two different Hilbert spaces that capture the same spaces, but describe the states of non-identical physical quantum systems.

Definition 3.10 (Partial trace of an operator)**.** *Let $n, m \in \mathbb{N}$ and a linear operator $M \in \mathcal{L}(\mathcal{H}_0 \otimes \mathcal{H}_1)$ for $\mathcal{H}_0 = \mathcal{S}(n)$, $\mathcal{H}_1 = \mathcal{S}(m)$, that is, $M \in \mathbb{C}^{2^{m+n} \times 2^{m+n}}$ is a square matrix. The partial trace of M with respect to $\mathcal{H}_0$ is defined as the following matrix in $\mathbb{C}^{2^m \times 2^m}$:*

$$Tr_{\mathcal{H}_0}(M) := \sum_{x \in \{0,1\}^n} \left(\langle x| \otimes I_{\mathcal{L}(\mathcal{H}_1)}\right) \cdot M \cdot \left(|x\rangle \otimes I_{\mathcal{L}(\mathcal{H}_1)}\right) \quad .$$

We can also write $Tr_n(M) := Tr_{\mathcal{H}_0}(M)$.

Definition 3.11 (Partial trace with different external input and output spaces)**.** *Let $\mathcal{H}_R = \mathcal{S}(n)$, $\mathcal{H}_P = \mathcal{S}(r)$, $\mathcal{H}_F = \mathcal{S}(\ell)$, and let $M \in \mathcal{L}(\mathcal{H}_R \otimes \mathcal{H}_P, \mathcal{H}_R \otimes \mathcal{H}_F)$. The partial trace over the two copies of $\mathcal{H}_R$ is the matrix*

$$Tr_R(M) := \sum_{x \in \{0,1\}^n} \left(\langle x| \otimes I_{\mathcal{L}(\mathcal{H}_F)}\right) M \left(|x\rangle \otimes I_{\mathcal{L}(\mathcal{H}_P)}\right) \quad .$$

This extends Definition 3.10; when $\mathcal{H}_P = \mathcal{H}_F$, the two definitions agree.

Definition 3.12 (Completely positive and trace-preserving maps)**.** *Let $\mathcal{C} \in \mathcal{L}(\mathcal{L}(\mathcal{H}_0), \mathcal{L}(\mathcal{H}_1))$. We say that $\mathcal{C}$ is completely positive if, for every additional finite-dimensional register Z and every positive semidefinite matrix $X \in \mathcal{L}(\mathcal{H}_0 \otimes \mathcal{H}_Z)$, the matrix $(\mathcal{C} \otimes \mathrm{id}_Z)(X)$ is positive semidefinite. Here id_Z is the identity map on $\mathcal{L}(\mathcal{H}_Z)$. We say that $\mathcal{C}$ is trace-preserving if $Tr(\mathcal{C}(X)) = Tr(X)$ for every $X \in \mathcal{L}(\mathcal{H}_0)$. A map satisfying both conditions is called CPTP.*

Definition 3.13 (Quantum channel)**.** *Let $\mathcal{H}_0$, $\mathcal{H}_1$ Hilbert spaces and let $\mathcal{C} \in \mathcal{L}(\mathcal{L}(\mathcal{H}_0), \mathcal{L}(\mathcal{H}_1))$. We say that $\mathcal{C}$ is a quantum channel from $\mathcal{H}_0$ to $\mathcal{H}_1$ if it is a completely positive, trace-preserving (linear) map from $\mathcal{L}(\mathcal{H}_0)$ to $\mathcal{L}(\mathcal{H}_1)$. In case $\mathcal{H}_0 = \mathcal{S}(n)$, $\mathcal{H}_1 = \mathcal{S}(m)$ for some $n, m \in \mathbb{N}$, we say that $\mathcal{C}$ is a channel from n to m qubits, or an n-to-m quantum channel.*

Throughout this work we use the Choi-Jamiolkowski isomorphism (or CJ isomorphism for short), which for Hilbert spaces $\mathcal{H}_0$, $\mathcal{H}_1$, induces a bijection between

$$\mathcal{L}(\mathcal{L}(\mathcal{H}_0), \mathcal{L}(\mathcal{H}_1)) \quad ,$$

which is the set of linear transformations from $\mathcal{L}(\mathcal{H}_0)$ to $\mathcal{L}(\mathcal{H}_1)$, and

$$\mathcal{L}(\mathcal{H}_0 \otimes \mathcal{H}_1) \quad ,$$

which is the set of linear transformations from $\mathcal{H}_0 \otimes \mathcal{H}_1$ to itself. The CJ isomorphism in particular induces an isomorphism between (1) n-to-m quantum channels and (2) a special set of matrices in $\mathbb{C}^{2^{n+m} \times 2^{n+m}}$. The CJ isomorphism is defined as follows.

Definition 3.14 (Choi-Jamiolkowski operator of a linear operator)**.** *Let $\mathcal{C} \in \mathcal{L}(\mathcal{L}(\mathcal{H}_0), \mathcal{L}(\mathcal{H}_1))$ a linear operator for $\mathcal{H}_0 = \mathcal{S}(n)$, $\mathcal{H}_1 = \mathcal{S}(m)$ for some $n, m \in \mathbb{N}$. The Choi-Jamiolkowski (or CJ in short) operator of $\mathcal{C}$ is given by the following matrix $\mathbf{CJ}(\mathcal{C}) \in \mathcal{L}(\mathcal{H}_0 \otimes \mathcal{H}_1)$:*

$$\mathbf{CJ}(\mathcal{C}) := \sum_{x,y \in \{0,1\}^n} |x\rangle\langle y| \otimes \mathcal{C}(|x\rangle\langle y|) \quad .$$

We will also use the inverse CJ isomorphism, which is defined as follows.

Definition 3.15 (Inverse of Choi-Jamiolkowski transformation)**.** *Let $M \in \mathcal{L}(\mathcal{H}_0 \otimes \mathcal{H}_1)$ a linear operator. The inverse CJ transformation of the tuple $(M, \mathcal{H}_0, \mathcal{H}_1)$ is defined as the linear operator $\mathbf{CJ}^{-1}(M, \mathcal{H}_0, \mathcal{H}_1) \in \mathcal{L}(\mathcal{L}(\mathcal{H}_0), \mathcal{L}(\mathcal{H}_1))$ such that for every $X \in \mathcal{L}(\mathcal{H}_0)$,*

$$\left[\mathbf{CJ}^{-1}(M, \mathcal{H}_0, \mathcal{H}_1)\right](X) = Tr_{\mathcal{H}_0}\left(\left(X^T \otimes I_{\mathcal{L}(\mathcal{H}_1)}\right) \cdot M\right) \quad .$$

Here X^T is the transpose in the specified computational basis of $\mathcal{H}_0$.

Remark 3.1 (3 input parameters in inverse CJ isomorphism). In the above definition 3.15 of the reverse CJ transformation $\mathbf{CJ}^{-1}$, note that unlike the forward transformation $\mathbf{CJ}$, we need to specify not only the operator M but also the spaces $\mathcal{H}_0$, $\mathcal{H}_1$. The reason is that the space $\mathcal{H}_0 \otimes \mathcal{H}_1$ can be decomposed in many different ways (i.e. for Hilbert spaces of different dimensions), and imply different operators by the inverse CJ transformation. This is different from the case of the forward transformation $\mathbf{CJ}$, where the input and output linear spaces are implicitly defined as part of the operator $\mathcal{C}$.

Definition 3.16 (Matrix representation of a quantum channel)**.** *Let $\mathcal{C}$ a quantum channel from $\mathcal{H}_0$ to $\mathcal{H}_1$. A matrix representation of $\mathcal{C}$ is a finite collection of matrices $C_1, \ldots, C_t \in \mathcal{L}(\mathcal{H}_0, \mathcal{H}_1)$ such that, for every $X \in \mathcal{L}(\mathcal{H}_0)$,*

$$\mathcal{C}(X) = \sum_{\alpha=1}^{t} C_\alpha X C_\alpha^\dagger \quad , \qquad \sum_{\alpha=1}^{t} C_\alpha^\dagger C_\alpha = I_{\mathcal{L}(\mathcal{H}_0)} \quad . \tag{3.1}$$

Every finite-dimensional quantum channel has such a representation. The matrices C_α are also called Kraus operators. They describe the channel mathematically; they are not additional input matrices that an algorithm is required to write down. Omitting the second equality gives a representation of a completely positive map that does not necessarily preserve trace.

Definition 3.17 (Unitary implementation of a quantum channel)**.** *Let $\mathcal{C}$ a quantum channel from $\mathcal{H}_0$ to $\mathcal{H}_1$. A unitary implementation of $\mathcal{C}$ consists of additional registers E, E' and a unitary $U : \mathcal{H}_0 \otimes \mathcal{H}_E \to \mathcal{H}_1 \otimes \mathcal{H}_{E'}$ such that, for every input density matrix ρ,*

$$\mathcal{C}(\rho) = Tr_{E'}\left(U\left(\rho \otimes |0\rangle\langle 0|_E\right) U^\dagger\right) \quad .$$

Such an implementation is also called a unitary dilation. For a circuit-described channel, its measurement and discarded registers can be kept coherently to obtain a polynomial-size implementation of this form, which is called the purification of the quantum circuit.

Definition 3.18 (Trace distance and approximation of a state)**.** *For a matrix M, its trace norm is $\|M\|_1 := Tr\sqrt{M^\dagger M}$. For two density matrices ρ, σ of the same dimension, their trace distance is*

$$\Delta_{\mathrm{tr}}(\rho, \sigma) := \frac{1}{2}\|\rho - \sigma\|_1 \quad .$$

A state-generation procedure has error at most ϵ if its output has trace distance at most ϵ from its specified output state.

4 Process Matrix Formalism

Process matrices were first defined in [OCB12] and since then became a central object in the study of indefinite causal structure in quantum mechanics. In this section we give the required definitions for information processing based on process matrices, computational models, computationally-bounded variations and complexity classes. We follow the definitions from the literature, in some places using previous definitions as they are, and in some other places giving logically equivalent definitions which would be easier to operate in the scope of this paper. We also provide numerous new definitions.

Definition 4.1 (Double ket notation)**.** *Let $M \in \mathcal{L}(\mathcal{H}_0, \mathcal{H}_1)$ a matrix such that $\mathcal{H}_0 = \mathcal{S}(n)$, $\mathcal{H}_1 = \mathcal{S}(m)$ for some $n, m \in \mathbb{N}$. We define the following "double ket" vectors in $\mathbb{C}^{2^{n+m}}$:*

$$|M\rangle\rangle := \sum_{x \in \{0,1\}^n} \left(|x\rangle \otimes M \cdot |x\rangle\right) \quad ,$$

$$\langle\langle M| := |M\rangle\rangle^\dagger := \sum_{x \in \{0,1\}^n} \left(\langle x| \otimes \langle x| \cdot M^\dagger\right) \quad .$$

Definition 4.2 (Boundary pairs and local quantum channels)**.** *Let $k \geq 0$, let $\mathbf{m} = (m_1, \ldots, m_k)$ be a tuple of positive integers, and let $n := \sum_{i=1}^{k} m_i$. For every $i \in [k]$, let R_i, S_i be m_i-qubit registers, and set*

$$\mathcal{H}_{A_I} = \bigotimes_{i=1}^{k} \mathcal{H}_{R_i}, \quad \mathcal{H}_{A_O} = \bigotimes_{i=1}^{k} \mathcal{H}_{S_i} \ .$$

The tuple $\mathbf{m}$ is the ordered partition into boundary pairs. In our circuit notation, the R_i are inputs to the circuit and the S_i are outputs from it. A local quantum channel on pair i is any channel

$$\mathcal{A}_i : \mathcal{L}(\mathcal{H}_{S_i} \otimes \mathcal{H}_{E_i}) \longrightarrow \mathcal{L}(\mathcal{H}_{R_i} \otimes \mathcal{H}_{E_i'}) \ ,$$

where E_i, E_i' are arbitrary additional registers. These additional registers are external: they are not connected back to the circuit. Their inputs may be entangled with each other and with the external input to the circuit.

We next define the indefinite operator of a matrix W, which is used later to define what is a process matrix. In what follows, we think of W as a CJ representation of some quantum channel from $\mathcal{L}(\mathcal{H}_{A_I} \otimes \mathcal{H}_P)$ to $\mathcal{L}(\mathcal{H}_{A_O} \otimes \mathcal{H}_F)$.

Definition 4.3 (Indefinite Operator)**.** *Let $W \in \mathcal{L}(\mathcal{H}_{A_I} \otimes \mathcal{H}_P \otimes \mathcal{H}_{A_O} \otimes \mathcal{H}_F)$ for Hilbert spaces $\mathcal{H}_{A_I}$, $\mathcal{H}_P$, $\mathcal{H}_{A_O}$, $\mathcal{H}_F$ such that $\mathcal{H}_{A_I} = \mathcal{H}_{A_O} = \mathcal{S}(n)$. The indefinite operator of W with respect to the Hilbert spaces $\mathcal{H}_{A_I}$, $\mathcal{H}_{A_O}$ is the operator $G_{W,A_I,A_O} \in \mathcal{L}(\mathcal{H}_P \otimes \mathcal{H}_F)$ such that*

$$G_{W,A_I,A_O} := Tr_{A_I,A_O}\left(W \cdot |A_I, P, F\rangle\langle A_I, P, F|\right) \ ,$$

where

$$|A_I, P, F\rangle := \sum_{x \in \{0,1\}^n} |x\rangle \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes |x\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \ ,$$

$$\langle A_I, P, F| := \sum_{y \in \{0,1\}^n} \langle y| \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes \langle y| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \ .$$

Connecting local operations to the feedback registers. We next allow a quantum channel to be inserted into each feedback pair. Each local channel receives a feedback output and returns a feedback input; its auxiliary registers remain external. The following definition gives the CJ operator of the resulting transformation. It uses the contraction from [AGB17, Section III], with our input and output labels.

Definition 4.4 (Indefinite operator with local channels)**.** *Let*

$$W \in \mathcal{L}(\mathcal{H}_{A_I} \otimes \mathcal{H}_P \otimes \mathcal{H}_{A_O} \otimes \mathcal{H}_F) \, ,$$

and fix a partition into feedback pairs as in Definition 4.2, with

$$\mathcal{H}_{A_I} = \bigotimes_{i=1}^{k} \mathcal{H}_{R_i} \quad \text{and} \quad \mathcal{H}_{A_O} = \bigotimes_{i=1}^{k} \mathcal{H}_{S_i} \ .$$

For each $i \in [k]$, let E_i, E_i' be auxiliary registers and let

$$\mathcal{A}_i : \mathcal{L}(\mathcal{H}_{S_i} \otimes \mathcal{H}_{E_i}) \longrightarrow \mathcal{L}\left(\mathcal{H}_{R_i} \otimes \mathcal{H}_{E_i'}\right)$$

be a quantum channel. Write $\boldsymbol{\mathcal{A}} := (\mathcal{A}_1, \ldots, \mathcal{A}_k)$, $E := (E_1, \ldots, E_k)$, and $E' := (E_1', \ldots, E_k')$.

The indefinite operator of W with these local channels is

$$G_{W,A_I,A_O}[\mathcal{A}] := Tr_{A_I,A_O}\left[\left(W^{T_{A_I,A_O}} \otimes I_{\mathcal{H}_E \otimes \mathcal{H}_{E'}}\right)\left(I_{\mathcal{H}_P \otimes \mathcal{H}_F} \otimes \bigotimes_{i=1}^{k} \mathbf{CJ}(\mathcal{A}_i)\right)\right] .$$

Here T_{A_I,A_O} is the partial transpose on the feedback registers in their computational bases. Each factor acts on its named registers, with tensor factors reordered accordingly. After taking the partial trace, the remaining registers are ordered as P, E, F, E'.

The induced linear map from P, E to F, E' is

$$\mathbf{PMO}(W, \mathcal{H}_{A_I}, \mathcal{H}_{A_O})[\mathcal{A}] := \mathbf{CJ}^{-1}(G_{W,A_I,A_O}[\mathcal{A}], \mathcal{H}_P \otimes \mathcal{H}_E, \mathcal{H}_F \otimes \mathcal{H}_{E'}) . \tag{4.1}$$

When all local channels are identities and there are no auxiliary registers, this reduces to the indefinite operator in Definition 4.3. The induced map need not be a quantum channel for an arbitrary W; requiring this for every choice of local channels is part of the process matrix definition below.

We define process matrices as follows.

Definition 4.5 (Process Matrix)**.** *Let $W \in \mathcal{L}(\mathcal{H}_{A_I} \otimes \mathcal{H}_P \otimes \mathcal{H}_{A_O} \otimes \mathcal{H}_F)$ for Hilbert spaces $\mathcal{H}_{A_I}$, $\mathcal{H}_P$, $\mathcal{H}_{A_O}$, $\mathcal{H}_F$ such that $\mathcal{H}_{A_I} = \mathcal{H}_{A_O} = \mathcal{S}(n)$. Define the following linear operator $\mathbf{PMO}(W, \mathcal{H}_{A_I}, \mathcal{H}_{A_O}) \in \mathcal{L}(\mathcal{L}(\mathcal{H}_P), \mathcal{L}(\mathcal{H}_F))$ as*

$$\mathbf{PMO}(W, \mathcal{H}_{A_I}, \mathcal{H}_{A_O}) := \mathbf{CJ}^{-1}\left(G_{W,\mathcal{H}_{A_I},\mathcal{H}_{A_O}},\ \mathcal{H}_P, \mathcal{H}_F\right) ,$$

where $G_{W,\mathcal{H}_{A_I},\mathcal{H}_{A_O}}$ is the indefinite operator of W with respect to the spaces $\mathcal{H}_{A_I}$, $\mathcal{H}_{A_O}$, as in Definition 4.3, and $\mathbf{CJ}^{-1}$ is the inverse CJ transformation, as in Definition 3.15.

Fix a partition $\mathbf{m}$ of the two boundary registers as in Definition 4.2. We say that W is a process matrix with respect to $\mathcal{H}_{A_I}$, $\mathcal{H}_{A_O}$, and $\mathbf{m}$ if W is positive semidefinite and, for every collection of auxiliary registers and every choice of local quantum channels $\mathcal{A}_1, \ldots, \mathcal{A}_k$, the induced operator

$$\mathbf{PMO}(W, \mathcal{H}_{A_I}, \mathcal{H}_{A_O})[\mathcal{A}]$$

from Definition 4.4 is a quantum channel. From hereon unless noted otherwise the partition $\mathbf{m}$ is implicitly set to be $(m_1, \ldots, m_k)$, where $k = n$ and such that for all $i \in [k]$, $m_i := 1$, which is why we suppress the notation.

Remark 4.1 (Intuition behind the operator $\mathbf{PMO}$). The name $\mathbf{PMO}$ is meant for a Process Matrix Operator. The idea behind the operator $\mathbf{PMO}$ is that it tries to take a general linear map, and map it to not just any linear transformation, but a quantum channel. What defines a process matrix W is that it is positive semidefinite and that this mapping gives a quantum channel for every choice of local quantum channels on the specified boundary pairs, including additional registers. The identity connection is only one of these choices.

4.1 A Computational Model based on Process Matrices

In the above subsections, all definitions were information theoretical. The work [AGB17] was the first to consider the complexity theoretical aspect of process matrices. In this subsection we add our own notions of computation and computational complexity of process matrices, based on the previous work.

Definition 4.6 (Quantum channel of a quantum circuit)**.** *For $n, m \in \mathbb{N}$, let C a quantum circuit from n to m qubits (as in defined in Section 3).*

The quantum channel of C, denoted $\mathcal{C}_C \in \mathcal{L}(\mathcal{L}(\mathcal{S}(n)), \mathcal{L}(\mathcal{S}(m)))$ is the linear map that agrees with the execution of C on every n-qubit mixed quantum state.

Definition 4.7 (Circuit description for a process)**.** *Let n a non-negative integer, C an $(n+r)$-to-$(n+\ell)$ quantum circuit with the boundary registers ordered as in Definition 4.2. The pair $D := (n, C)$ is a circuit description for a process. We implicitly use n separate single-qubit boundary pairs (R_i, S_i), $i \in [n]$, where R_i is an input qubit of C and S_i is the corresponding output qubit.*

For any n arbitrary local channels $\mathcal{A}_1, \ldots, \mathcal{A}_n$, the channel $\mathcal{A}_i$ acts from $S_i E_i$ to $R_i E'_i$, where E_i, E'_i may have arbitrary sizes and remain external. Their inputs may be entangled with each other and with the external input to C. For $n = 0$, there are no boundary pairs. The size $|D|$ includes n and $|C|$.

Quantum Circuits as Process Matrix Generators. We next define a new notion, called Process Matrix Generators (PMG). Intuitively, PMGs are quantum circuits such that if we take their CJ representations, we get a process matrix. More precisely, for an $(n+r)$-to-$(n+\ell)$ quantum circuit C we take the n first qubits to be boundary registers and ask whether the CJ representation is a process matrix w.r.t. these registers. We care about polynomial-time PMGs, as they play a central role in the definition of polynomial-time process matrix computation. In particular, when we later define the complexity class $\mathbf{BQP}_{\ell\mathrm{CTC}}$ from previous work (and a new class $\mathbf{BQP}_{u\mathrm{CTC}}$), we will do so using the new definition, and ask that all process matrix transformations allowed in $\mathbf{BQP}_{\ell\mathrm{CTC}}$ are only by quantum polynomial-time PMGs.

Definition 4.8 (Process Matrix Generator)**.** *Let $n \in \mathbb{N} \cup \{0\}$, let $r, \ell \in (\mathbb{N} \cup \{0\})$ and let C an $(n+r)$-to-$(n+\ell)$ quantum circuit. Consider the quantum channel $\mathcal{C}_C \in \mathcal{L}\left(\mathcal{L}\left(\mathcal{H}_{A_I} \otimes \mathcal{H}_P\right), \mathcal{L}\left(\mathcal{H}_{A_O} \otimes \mathcal{H}_F\right)\right)$ of the circuit C, such that $\mathcal{H}_{A_I} = \mathcal{S}(n)$, $\mathcal{H}_P = \mathcal{S}(r)$, $\mathcal{H}_{A_O} = \mathcal{S}(n)$, $\mathcal{H}_F = \mathcal{S}(\ell)$.*

Define the following operator $\mathbf{PMO}_{(n,C)} \in \mathcal{L}\left(\mathcal{L}\left(\mathcal{H}_P\right), \mathcal{L}\left(\mathcal{H}_F\right)\right)$:

$$\mathbf{PMO}_{(n,C)} := \mathbf{PMO}\left(\mathbf{CJ}\left(\mathcal{C}_C\right), \mathcal{H}_{A_I}, \mathcal{H}_{A_O}\right) \quad ,$$

where $\mathbf{PMO}\left(\mathbf{CJ}\left(\mathcal{C}_C\right), \mathcal{H}_{A_I}, \mathcal{H}_{A_O}\right)$ is as per Definition 4.5. We use the implicit single-qubit boundary convention from Definition 4.7. If $\mathbf{CJ}\left(\mathcal{C}_C\right)$ is a process matrix for these n separate boundary pairs, as in Definition 4.5, we say that the tuple (n, C) is a Process Matrix Generator (or PMG), and denote by $\mathbf{PMO}_{(n,C)}$ the Process Matrix Channel of the tuple. For $D = (n, C)$, we also write $\mathbf{PMO}_D := \mathbf{PMO}_{(n,C)}$ and $\mathbf{PMO}_D[\mathcal{A}]$ for the connection with local channels.

Simplified formula for evaluating process matrix generators. The following Lemma gives a formula for the identity-connected operator of the tuple (n, C). It does not replace the validity condition for all local quantum channels. Specifically, assume that C is an isometry on quantum states, that given an $(n+r)$-qubit pure state $|\psi\rangle \in \mathcal{S}(n+r)$, concatenates k ancillary qubits and executes a unitary U on $n+r+k$ qubits. The below Lemma is a generalization of the simplifying Equation (21) from [AGB17].

Lemma 4.1 (Simplified PM Formula for Isometries)**.** *Let $n, r \in \mathbb{N}$, $k \in \mathbb{N} \cup \{0\}$, and let $U \in \mathbb{C}^{\left(2^{n+r+k}\right) \times \left(2^{n+r+k}\right)}$ a unitary matrix. Let $\mathcal{C}_{(U,k)}$ the quantum channel of the k-isometry of U, that is, for an input $(n+r)$-qubit mixed state ρ the channel adds an ancilla of k qubits and executes U, that is,*

$$\mathcal{C}_{(U,k)}(\rho) := U \cdot \left(\rho \otimes |0^k\rangle\langle 0^k|\right) \cdot U^\dagger \quad .$$

Denote the Hilbert spaces of the channel,

$$\mathcal{C}_{(U,k)} \in \mathcal{L}\left(\mathcal{L}\left(\mathcal{H}_{A_I} \otimes \mathcal{H}_P\right), \mathcal{L}\left(\mathcal{H}_{A_O} \otimes \mathcal{H}_F\right)\right) \quad ,$$

such that $\mathcal{H}_{A_I} = \mathcal{S}(n)$, $\mathcal{H}_P = \mathcal{S}(r)$, $\mathcal{H}_{A_O} = \mathcal{S}(n)$, $\mathcal{H}_F = \mathcal{S}(r+k)$.

Let $W := \mathbf{CJ}\left(\mathcal{C}_{(U,k)}\right)$ and let G_{W,A_I,A_O} the indefinite operator of W with respect to $\mathcal{H}_{A_I}$, $\mathcal{H}_{A_O}$ (as in Definition 4.3). Then we have the equality

$$G_{W,A_I,A_O} = \mathbf{CJ}\left(\mathcal{C}_{(U_G,k)}\right) \quad ,$$

for the matrix $U_G \in \mathbb{C}^{2^{r+k}\times 2^{r+k}}$ *such that*

$$U_G = Tr_{\mathcal{H}_{A_I}}(U) \quad .$$

That is, U_G *is the matrix derived by tracing out the Hilbert space of the leftmost* n *qubits from the unitary matrix* U *itself, and the linear operator* $\mathcal{C}_{(U_G,k)}$ *is such that for an input matrix* $\sigma \in \mathbb{C}^{2^r\times 2^r}$*, outputs:*

$$\mathcal{C}_{(U_G,k)}(\sigma) = U_G \cdot \left(\sigma \otimes |0^k\rangle\langle 0^k|\right) \cdot U_G^\dagger \quad .$$

In particular, if W *is a process matrix with respect to* $\mathcal{H}_{A_I}$ *and* $\mathcal{H}_{A_O}$ *under the implicit single-qubit boundary convention, then*

- $\mathcal{C}_{(U_G,k)}$ *is a valid quantum channel.*
- *For any quantum circuit* C *that implements the original channel* $\mathcal{C}_{(U,k)}$ *with the same ordered boundary registers, the tuple* (n, C) *is a PMG with generated channel* $\mathbf{PMO}_{(n,C)} = \mathcal{C}_{(U_G,k)}$ *(as in Definition 4.8).*

Note that the converse does not follow from the identity-connected formula: $\mathcal{C}_{(U_G,k)}$ *can be a quantum channel even when* W *is not a process matrix.*

Proof. Note that there is an isometry $V \in \mathbb{C}^{2^{n+r+k}\times 2^{n+r}}$ that for any $x \in \{0,1\}^{n+r}$, $V \cdot |x\rangle = U \cdot \left(|x\rangle \otimes |0^k\rangle\right)$. It follows that the action of the channel $\mathcal{C}_{(U,k)}$ for an input $(n+r)$-qubit mixed state ρ is $V \cdot \rho \cdot V^\dagger$. This implies that $W := \mathbf{CJ}\left(\mathcal{C}_{(U,k)}\right) = |V\rangle\!\rangle\langle\!\langle V|$.

By Definition 4.3, the indefinite operator G_{W,A_I,A_O} of $W = |V\rangle\!\rangle\langle\!\langle V|$ with respect to the Hilbert spaces $\mathcal{H}_{A_I}, \mathcal{H}_{A_O}$ is

$$G_{W,A_I,A_O} := Tr_{A_I,A_O}\left(|V\rangle\!\rangle\langle\!\langle V| \cdot |A_I,P,F\rangle\langle A_I,P,F|\right) \quad .$$

Let us break down the expression inside the partial trace. By Definition 4.1,

$$|V\rangle\!\rangle\langle\!\langle V| := \sum_{x\in\{0,1\}^{n+r}} (|x\rangle \otimes V \cdot |x\rangle) \sum_{y\in\{0,1\}^{n+r}} \left(\langle y| \otimes \langle y| \cdot V^\dagger\right) \quad .$$

From Definition 4.3 we also have

$$|A_I,P,F\rangle := \sum_{u\in\{0,1\}^n} |u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes |u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \quad ,$$

$$\langle A_I,P,F| := \sum_{v\in\{0,1\}^n} \langle v| \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes \langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \quad .$$

By the linearity of trace we get

$$\begin{aligned} G_{W,A_I,A_O} &:= Tr_{A_I,A_O}\left(|V\rangle\!\rangle\langle\!\langle V| \cdot |A_I,P,F\rangle\langle A_I,P,F|\right) \\ = \sum_{x,y\in\{0,1\}^{n+r},u,v\in\{0,1\}^n} & Tr_{A_I,A_O}\Big((|x\rangle \otimes V \cdot |x\rangle)\left(\langle y| \otimes \langle y| \cdot V^\dagger\right) \\ & \cdot \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes |u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)}\right)\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes \langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)}\right)\Big) \quad . \end{aligned}$$

Next, by the cyclic property of trace, the above equals

$$= \sum_{x,y\in\{0,1\}^{n+r},u,v\in\{0,1\}^n} Tr_{A_I,A_O}\Big(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes \langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)}\right)(|x\rangle \otimes V \cdot |x\rangle)$$

$$\cdot \Big(\langle y| \otimes \langle y| \cdot V^\dagger \Big) \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes |u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \Bigg)$$

$$= \sum_{a,b\in\{0,1\}^n, i,j\in\{0,1\}^r, u,v\in\{0,1\}^n} Tr_{A_I,A_O} \Bigg(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes \langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \left(|a,i\rangle \otimes V \cdot |a,i\rangle \right)$$

$$\cdot \Big(\langle b,j| \otimes \langle b,j| \cdot V^\dagger \Big) \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_P)} \otimes |u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \Bigg) \ .$$

By considering the result of inner products for the expression inside the trace, we get that the summand for i, j, u, v is

$$\Big(|i\rangle \otimes \left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v,i\rangle \Big) \cdot \Big(\langle j| \otimes \langle u,j| \cdot V^\dagger \cdot \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \Big) \ .$$

Observe that at this point of the calculation, for the expression inside the trace (for each set of indices i, j, u, v), the systems $\mathcal{H}_{A_I}, \mathcal{H}_{A_O}$ are not longer present. In particular, the trace can be removed from the previous sum, and we get that G_{W,A_I,A_O} equals:

$$= \sum_{i,j\in\{0,1\}^r, u,v\in\{0,1\}^n} \Big(|i\rangle \otimes \left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v,i\rangle \Big) \cdot \Big(\langle j| \otimes \langle u,j| \cdot V^\dagger \cdot \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \Big) \ .$$

We next consider, for every *fixed* pair $i, j \in \{0,1\}^r$, the sum over all *different* $u, v \in \{0,1\}^n$:

$$\sum_{v,u\in\{0,1\}^n} \Big(|i\rangle \otimes \left(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v,i\rangle \right) \Big) \cdot \Big(\langle j| \otimes \Big(\langle u,j| \cdot V^\dagger \cdot \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \Big) \Big) \ .$$

Observe that in the above we have a sum, over $u, v \in \{0,1\}^n$, where every summand is a product between two elements. However, the left element in each product only depends on v, and every element on the right only depends on u. It follows that the above can be written in the form:

$$= \left(\sum_{v\in\{0,1\}^n} |i\rangle \otimes \left(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v,i\rangle \right) \right) \cdot \left(\sum_{u\in\{0,1\}^n} \langle j| \otimes \Big(\langle u,j| \cdot V^\dagger \cdot \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \Big) \right) \ .$$

Let us analyze the left multiplicand, and the analysis of the right multiplicand will be symmetric:

$$\sum_{v\in\{0,1\}^n} |i\rangle \otimes \left(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v,i\rangle \right)$$

$$= |i\rangle \otimes \left(\sum_{v\in\{0,1\}^n} \left(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v,i\rangle \right) \right) \ .$$

Recall that $\forall (v,i) \in \{0,1\}^{n+r}$ we have $V \cdot |v,i\rangle = U \cdot \left(|v,i\rangle \otimes |0^k\rangle \right)$. Now, observe that for every $i \in \{0,1\}^r$ we have

$$\sum_{v\in\{0,1\}^n} \left(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v,i\rangle \right)$$

$$= \sum_{v\in\{0,1\}^n} \Big(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot U \cdot \Big(|v,i\rangle \otimes |0^k\rangle \Big) \Big)$$

$$= \left(Tr_{A_I}(U) \right) \cdot \Big(|i\rangle \otimes |0^k\rangle \Big) \ .$$

To conclude, by denoting $U_G := Tr_{A_I}(U) \in \mathbb{C}^{2^{r+k} \times 2^{r+k}}$, for every $i \in \{0,1\}^r$ we have the equality :

$$\sum_{v \in \{0,1\}^n} |i\rangle \otimes \left(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v, i\rangle \right) = |i\rangle \otimes U_G \cdot \left(|i\rangle \otimes |0^k\rangle \right) \ .$$

In a similar manner, by a symmetric calculation we have the equality for every $j \in \{0,1\}^r$:

$$\sum_{u \in \{0,1\}^n} \langle j| \otimes \left(\langle u, j| \cdot V^\dagger \cdot \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \right) = \langle j| \otimes \left(\langle j| \otimes \langle 0^k| \right) \cdot U_G^\dagger \ .$$

Let us summarize the results and conclude the proof. We saw that

$$G_{W,A_I,A_O} = \sum_{i,j, \in \{0,1\}^r} \left(\sum_{v \in \{0,1\}^n} |i\rangle \otimes \left(\left(\langle v| \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \cdot V \cdot |v, i\rangle \right) \right) \cdot \left(\sum_{u \in \{0,1\}^n} \langle j| \otimes \left(\langle u, j| \cdot V^\dagger \cdot \left(|u\rangle \otimes I_{\mathcal{L}(\mathcal{H}_F)} \right) \right) \right) \ ,$$

and by our analysis of the multiplicands in the sum, the above equals

$$= \sum_{i,j, \in \{0,1\}^r} \left(|i\rangle \otimes U_G \cdot \left(|i\rangle \otimes |0^k\rangle \right) \right) \left(\langle j| \otimes \left(\langle j| \otimes \langle 0^k| \right) \cdot U_G^\dagger \right)$$

$$= \sum_{i,j, \in \{0,1\}^r} |i\rangle\langle j| \otimes \left(U_G \cdot \left(|i\rangle \otimes |0^k\rangle \right) \right) \left(\left(\langle j| \otimes \langle 0^k| \right) \cdot U_G^\dagger \right)$$

$$= \sum_{i,j, \in \{0,1\}^r} |i\rangle\langle j| \otimes \left(U_G \cdot \left(|i\rangle\langle j| \otimes |0^k\rangle\langle 0^k| \right) \cdot U_G^\dagger \right) \ .$$

It is left to observe that by the Definition 3.14, the last expression is exactly the CJ transformation of the linear operator $\mathcal{C}_{(U_G,k)}$ defined in the statement of Lemma 4.1. □

4.2 Existing and New Complexity Classes

We now define two models of efficient process matrix computation. The first allows general generating circuits, provided they describe valid processes. The second requires the generating circuits to be unitary and the processes to be pure. We also distinguish this restriction from purifiability, which requires a pure extension to exist without imposing an efficiency requirement on its construction.

General process computation. Both our model and that of [AGB17, Section IV] assume access to the channel specified by a classical generating-circuit description. In the CTC interpretation, the model grants the ability to construct the corresponding feedback connections. The previous definition requires the supplied circuit to describe a valid process. Here we also specify what happens when it does not: a well-formed generation query returns an arbitrary quantum channel of the prescribed dimensions, and the algorithm must satisfy its correctness requirement for every such channel. This defines the generation operation on every well-formed description, without requiring the algorithm to determine process validity.

Definition 4.9 (Linear Closed Timelike Curve Generator)**.** *Let A a function with input set of pairs (m, C) where $m \in \mathbb{N} \cup \{0\}$ and C a quantum circuit from $m + r$ to $m + \ell$ qubits for some $r, \ell \in \mathbb{N} \cup \{0\}$, and output set of r-to-ℓ qubit quantum channels. A need not be efficiently computable. We say that A is a linear CTC generator if for every well-formed input (m, C), if the pair is a PMG (as in Definition 4.8) then A outputs the channel $\mathbf{PMO}_{(m,C)}$, and if the input is not a PMG, the output channel of A can be arbitrary.*

We can now define the complexity class $\mathbf{BQP}_{\ell\mathrm{CTC}}$.

Definition 4.10 (The Complexity Class $\mathbf{BQP}_{\ell\mathrm{CTC}}$)**.** *The complexity class $\boldsymbol{BQP}_{\ell CTC}$ is the set of promise problems $\prod := \left(\prod_{YES}, \prod_{NO}\right)$ that are solvable by a quantum polynomial-time algorithm with access to linear post-selection CTCs. Formally, such that there exists a quantum polynomial-time Turing Machine M such that for every linear CTC generator A (as per Definition 4.9), for every $x \in \prod_{YES}$, $M^A(x) = 1$ with probability at least $p(x)$, and for every $x \in \prod_{NO}$, $M^A(x) = 0$ with probability at least $p(x)$, such that*

- $$\forall x \in \left(\prod_{YES} \cup \prod_{NO}\right) : p(x) \geq 1 - 2^{-|x|} \ ,$$
- *M^A is the machine M with oracle access to A and the channels that it outputs.*
- *Every generation query (m, C) has a classical description. The query may depend on the input x itself and on the preceding computation, and may be adaptive.*
- *Every query (m, C) to A and every use of the channel it produces costs $\Theta(|C|)$ time.*

Pure processes. Araújo, Feix, Navascués and Brukner [AFNB17, Definition 1] introduced pure processes as those that induce a global unitary transformation whenever the local operations are unitary, including operations on auxiliary registers. We give the relevant definitions.

Definition 4.11 (Unitary Quantum Channel)**.** *Let $n \in \mathbb{N} \cup \{0\}$ and let $\mathcal{C}$ be an n-to-n qubit quantum channel, as in Definition 3.13. We say that $\mathcal{C}$ is a unitary quantum channel if there exists a unitary matrix $U_{\mathcal{C}} \in \mathbb{C}^{2^n \times 2^n}$ such that, for every n-qubit pure state $|\psi\rangle$,*

$$\mathcal{C}(|\psi\rangle\langle\psi|) = U_{\mathcal{C}} \cdot |\psi\rangle\langle\psi| \cdot U_{\mathcal{C}}^{\dagger} \ .$$

Definition 4.12 (Pure Process Matrix)**.** *Let W be a process matrix as in Definition 4.5, with feedback input R, feedback output S, external input P, and external output F, with a fixed partition into feedback pairs (R_i, S_i), $i \in [k]$.*

We say that W is pure *if, for every choice of auxiliary registers E_i, E_i' and local unitary channels $\mathcal{A}_i$ from S_i, E_i to R_i, E_i', the induced channel*

$$\mathbf{PMO}(W, \mathcal{H}_R, \mathcal{H}_S)[\mathcal{A}_1, \ldots, \mathcal{A}_k]$$

from $P, E_1, \ldots, E_k$ to $F, E_1', \ldots, E_k'$ is unitary.

By Theorem 2 of [AFNB17], a process matrix W is pure if and only if

$$W = |U\rangle\rangle\langle\langle U|$$

for a unitary $U : \mathcal{H}_R \otimes \mathcal{H}_P \longrightarrow \mathcal{H}_S \otimes \mathcal{H}_F$. We use this notion to define pure process matrix generators.

Definition 4.13 (Pure Process Matrix Generator)**.** *Let $D = (n, C)$ be a PMG as in Definition 4.8. We say that D is a* pure process matrix generator*, or pure PMG, if C is a unitary quantum circuit. Each of the following conditions is equivalent to D being pure:*

- **CJ** $(\mathcal{C}_C)$ *is a pure process matrix according to Definition 4.12, with the same n separate single-qubit feedback pairs.*

- *C is a unitary circuit with matrix U_C satisfying the following condition. Write $R := (R_1, \ldots, R_n)$ for the feedback registers. For every choice of auxiliary registers $E := (E_1, \ldots, E_n)$ and local unitaries V_i acting on R_i, E_i,*

$$Tr_R \left[\left(\bigotimes_{i=1}^{n} V_i \right) (U_C \otimes I_{\mathcal{H}_E}) \right] \quad \text{is unitary.} \tag{4.2}$$

Here we identify corresponding feedback input and output registers, and each operator acts as the identity on registers not mentioned in its definition. The auxiliary registers remain external and are not traced out.

In particular, the generated channel $\mathbf{PMO}_D$ is unitary.

Remark 4.2 (A PMG with a unitary generated channel is not necessarily pure). A unitary generated channel alone does not make a PMG pure. For example, our Choi-state construction in Lemma 5.1 generates a unitary channel when V is unitary, although the ordinary channel $\mathcal{C}_C$ implemented by its generating circuit is not unitary. To see which local operations reveal this distinction, introduce an auxiliary input qubit E_i at each feedback pair (R_i, S_i) and choose the local operation to be a SWAP. This sends E_i into R_i and returns S_i as an auxiliary output E_i'. The resulting global channel is exactly $\mathcal{C}_C$, up to register names. Thus, although every inserted operation is unitary, the global channel is not. Purity requires unitarity for every such choice, not only when the local operations are identities.

Pure process computation. We define the class $\mathbf{BQP}_{u\mathrm{CTC}}$, which restricts $\mathbf{BQP}_{\ell\mathrm{CTC}}$ to pure process computation.

Definition 4.14 (Unitary Closed Timelike Curve Generator)**.** *Let A a function with input set of pairs (m, C) where $m \in \mathbb{N} \cup \{0\}$ and C a quantum circuit from $m + r$ to $m + r$ qubits for some $r \in \mathbb{N} \cup \{0\}$, and output set of r-to-r qubit quantum channels. We say that A is a unitary CTC generator if for every well-formed input (m, C), if the pair is a pure PMG (as in Definition 4.13) then A outputs the channel $\mathbf{PMO}_{(m,C)}$ (and if the input is not a pure PMG, the output channel of A can be any r-to-r quantum channel).*

We can now define the complexity class $\mathbf{BQP}_{u\mathrm{CTC}}$.

Definition 4.15 (The Complexity Class $\mathbf{BQP}_{u\mathrm{CTC}}$)**.** *The complexity class $\boldsymbol{BQP}_{uCTC}$ is the set of promise problems $\prod := (\prod_{YES}, \prod_{NO})$ solvable by a quantum polynomial-time algorithm with access to unitary post-selection CTCs. Formally, there exists a quantum polynomial-time Turing machine M such that, for every unitary CTC generator A as in Definition 4.14,*

$$\forall x \in \prod_{YES} : \Pr\left[M^A(x) = 1\right] \geq 1 - 2^{-|x|} \ ,$$

and

$$\forall x \in \prod_{NO} : \Pr\left[M^A(x) = 0\right] \geq 1 - 2^{-|x|} \ .$$

The access and cost conventions are as follows:

- *M^A is the machine M with oracle access to A and the channels that it returns.*
- *Every generation query (m, C) has a classical description. Queries may depend on the input x and on the preceding computation, and may be adaptive.*
- *Every query (m, C) to A and every use of the channel it returns costs $\Theta(|C|)$ time.*

Purifiable processes. Araújo, Feix, Navascués and Brukner [AFNB17, Definition 3] introduced purifiable processes and proposed purifiability as a necessary condition for physical realizability. Informally, a process is purifiable if it can be recovered from a larger pure process by initializing additional inputs to pure states and discarding additional outputs.

Definition 4.16 (Purifiable Process Matrix)**.** *Let W be a process matrix as in Definition 4.5, with feedback input R, feedback output S, external input P, and external output F, with a fixed partition of R, S into feedback pairs. We say that W is* purifiable *if there are additional external registers P', F' and a unitary*

$$U : \mathcal{H}_R \otimes \mathcal{H}_P \otimes \mathcal{H}_{P'} \longrightarrow \mathcal{H}_S \otimes \mathcal{H}_F \otimes \mathcal{H}_{F'}$$

satisfying both of the following conditions:

1. *$\widetilde{W} := |U\rangle\!\rangle\langle\!\langle U|$ is a process matrix for the same feedback pairs, with external input P, P' and external output F, F'. Since U is unitary, $\widetilde{W}$ is a pure process.*
2. *Initializing P' to zero and discarding F' recovers W. Formally, writing*

$$V := U\left(I_{\mathcal{H}_R \otimes \mathcal{H}_P} \otimes |0\rangle_{P'}\right) \quad ,$$

we have

$$W = Tr_{F'}\left(|V\rangle\!\rangle\langle\!\langle V|\right) \quad .$$

Validity of $\widetilde{W}$ is required on every input to P', including inputs entangled with the other external registers, and under every choice of local quantum channels. No circuit-synthesis or efficiency requirement is imposed on U.

Definition 4.17 (Purifiable Process Matrix Generator)**.** *Let $D = (n, C)$ be a PMG as in Definition 4.8. We say that D is* purifiable *if* **CJ** $(\mathcal{C}_C)$ *is a purifiable process matrix according to Definition 4.16, with the same n separate single-qubit feedback pairs.*

Equivalently, there is a unitary U defining a pure extension as above such that, for every matrix X on the original circuit input,

$$\mathcal{C}_C(X) = Tr_{F'}\left[U\left(X \otimes |0\rangle\langle 0|_{P'}\right)U^\dagger\right] \quad .$$

If U is implemented by a circuit $\widetilde{C}$, then $\left(n, \widetilde{C}\right)$ is a pure PMG. The existence of such a circuit over our fixed gate set, its size, and the complexity of constructing it are additional requirements, not part of purifiability.

5 Quantum Computation with Efficient General Process Matrices

We show that general process matrix computation can solve every problem in **NP**. Our main tool turns an efficient preparation of an isometry's Choi state into a process matrix generator implementing that isometry. We first prove this construction and its purifiability, and then use it for unique-witness search, solving **NP**, and coherent permutation inversion.

5.1 Generating a unitary channel from a generator of its Choi state

Preparing the Choi state of an isometry does not by itself give an efficient circuit implementing that isometry. The following lemma shows that it does give an efficient process matrix generator. We also prove a stronger validity guarantee that will allow us to establish purifiability.

Lemma 5.1 (Generating an isometry from its Choi-state generator)**.** *Let $n, m \in \mathbb{N} \cup \{0\}$, and let $V \in \mathcal{L}(\mathcal{S}(n), \mathcal{S}(m))$ be an isometry. Denote $d := 2^n$, and let*

$$|J_V\rangle := \frac{1}{\sqrt{d}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes V|x\rangle \quad .$$

Assume there is an ordinary polynomial-size quantum circuit P_V that prepares the state $|J_V\rangle$ exactly. Then there is a classical deterministic polynomial-time algorithm that, given P_V, outputs a pair $(2n, C)$ which is a PMG (Definition 4.8) such that, for every input quantum state ρ,

$$\mathbf{PMO}_{(2n,C)}(\rho) = V \rho V^\dagger \quad . \tag{5.1}$$

In particular, when $n = m$, the generated channel is unitary.

The pair $(2n, C)$ actually satisfies a stronger property than being a PMG. Specifically, the induced Process Matrix Operator is a quantum channel even when the $2n$ single-qubit feedback registers are treated as one joint register. That is, the validity condition holds for every quantum channel $\mathcal{A}$ acting jointly on all $2n$ feedback qubits, with arbitrary auxiliary input and output registers, rather than only for tuples of local quantum channels $\mathcal{A}_1, \ldots, \mathcal{A}_{2n}$ as in Definition 4.5.

Proof. Let $d := 2^n$, and denote by P_j, for $j \in \{0,1\}^{2n}$, the n-qubit Pauli matrices, chosen with phases so that $P_j^\dagger = P_j$, $P_j^2 = I_n$, and $P_{0^{2n}} = I_n$. These matrices satisfy that the trace of their inner product with each other is always 0 unless it is with themselves, specifically, $Tr(P_j^\dagger P_{j'}) = d \cdot \mathbf{1}[j = j']$.

Write

$$|\Phi_d\rangle := \frac{1}{\sqrt{d}} \sum_{x \in \{0,1\}^n} |x, x\rangle \quad \text{and} \quad |\beta_j\rangle := (P_j \otimes I_n)|\Phi_d\rangle \quad .$$

The vectors $\{|\beta_j\rangle\}_{j \in \{0,1\}^{2n}}$ form an orthonormal basis for $\mathcal{S}(2n)$. Next, consider the $(2n+m)$-qubit state $|\psi\rangle \otimes |J_V\rangle$, and specifically, making a partial measurement to its leftmost $2n$ qubits in the basis $\{|\beta_j\rangle\}_{j \in \{0,1\}^{2n}}$. By definition, for every $j \in \{0,1\}^{2n}$, the measurement outputs j with probability

$$p_j := \left\| \left(\langle \beta_j | \otimes I_m \right) \left(|\psi\rangle \otimes |J_V\rangle \right) \right\|^2 \quad ,$$

and conditioned on this outcome, the remaining m qubits are in the state

$$\frac{1}{\sqrt{p_j}} \left(\langle \beta_j | \otimes I_m \right) \left(|\psi\rangle \otimes |J_V\rangle \right) \quad ,$$

whenever $p_j > 0$. A useful observation is that for every quantum state $|\psi\rangle \in \mathcal{S}(n)$ and $j \in \{0,1\}^{2n}$ we have

$$\left(\langle \beta_j | \otimes I_m \right) \left(|\psi\rangle \otimes |J_V\rangle \right) = \frac{1}{d} V P_j |\psi\rangle \quad . \tag{5.2}$$

To see the equality, expand the three vectors in the computational basis and use the definition of $|\beta_j\rangle$.

Now, since $V^\dagger V = I$ and P_j is unitary, the squared norm of the right-hand side is $1/d^2$, independently from $|\psi\rangle$. This altogether implies that measuring the first two registers in the basis $\{|\beta_j\rangle\}_j$ produces each j with probability $1/d^2$, and the corresponding external state is $V P_j |\psi\rangle$. The corresponding statement for mixed inputs and inputs entangled with additional registers follows by linearity, where the transformation acts as the identity on the additional registers.

The process matrix generator. We describe a polynomial-size ordinary quantum circuit C acting on $(2n+n)$-qubit inputs and outputting $2n+m$ qubits. Specifically, the feedback registers are the first $2n$ qubits, thus there are additional n external input qubits and m external output qubits. On an arbitrary input, C executes the following procedure:

1. Measure the feedback input in the computational basis, and denote the outcome by $r \in \{0,1\}^{2n}$.
2. Execute the circuit P_V to prepare $|J_V\rangle$ in an $(n+m)$-qubits register R_V, and perform the measurement from Equation (5.2) on the remaining n qubits in the external input and the first n qubits of R_V. Denote the measurement outcome by $j \in \{0,1\}^{2n}$.
3. Set the feedback output to $a := r \oplus j$ by adding (mod 2) the measurement outcome to the feedback registers. Output the remaining m qubits of R_V as the external output, and discard the remaining registers.

The circuit C uses only ordinary quantum operations and note that its size is polynomial in $|P_V|+n+m$.

Validity of the generator. We next show that the pair $(2n, C)$ constructed above is a PMG. Recall that in our definition a PMG needs to remain a valid quantum channel even when arbitrary quantum channels, with arbitrary auxiliary registers, are inserted in the feedback. In fact, we prove a stronger property, where all the $2n$ feedback qubits are treated as a single register and the inserted quantum channel can act jointly on all of them.

Let E, E' be arbitrary auxiliary registers, and let $\mathcal{A}$ be an arbitrary quantum channel from the $2n$ feedback output qubits and E to the $2n$ feedback input qubits and E'. Formally,

$$\mathcal{A} : \mathcal{L}\left(\mathcal{S}(2n) \otimes \mathcal{H}_E\right) \longrightarrow \mathcal{L}\left(\mathcal{S}(2n) \otimes \mathcal{H}_{E'}\right) \quad .$$

Recall the matrix representation of a quantum channel from Definition 3.16 through Kraus operators. Since $\mathcal{A}$ is a quantum channel, there exist matrices $A_1, \ldots, A_t \in \mathcal{L}\left(\mathcal{S}(2n) \otimes \mathcal{H}_E, \mathcal{S}(2n) \otimes \mathcal{H}_{E'}\right)$ such that for every input matrix X,

$$\mathcal{A}(X) = \sum_{\ell=1}^{t} A_\ell X A_\ell^\dagger \quad \text{and} \quad \sum_{\ell=1}^{t} A_\ell^\dagger A_\ell = I_{\mathcal{L}(\mathcal{S}(2n)\otimes\mathcal{H}_E)} \quad .$$

We use the Kraus operators $A_1, \ldots, A_t$ only for analyzing the quantum channel $\mathcal{A}$, and do not assume that the algorithm receives their descriptions or can compute them efficiently.

Recall that the feedback channel $\mathcal{A}$ has input which is the out a of C on its first $2n$ qubits and output r which is conceptually the input to C. For every $r, a \in \{0,1\}^{2n}$ and $\ell \in [t]$, denote by $A_\ell^{r,a} \in \mathcal{L}(\mathcal{H}_E, \mathcal{H}_{E'})$ the part of A_ℓ corresponding to the feedback input a and the feedback output r. Formally, we define

$$A_\ell^{r,a} := \left(\langle r| \otimes I_{\mathcal{L}(\mathcal{H}_{E'})}\right) A_\ell \left(|a\rangle \otimes I_{\mathcal{L}(\mathcal{H}_E)}\right) \quad .$$

Thus, on an auxiliary input state $|\eta\rangle \in \mathcal{H}_E$, the vector $A_\ell^{r,a}|\eta\rangle$ is the r-component of $A_\ell\left(|a\rangle \otimes |\eta\rangle\right)$ after removing the feedback output register. In particular, the second equality in the matrix representation of $\mathcal{A}$ gives, for every fixed $a \in \{0,1\}^{2n}$,

$$\begin{aligned} \sum_{r\in\{0,1\}^{2n}} \sum_{\ell=1}^{t} \left(A_\ell^{r,a}\right)^\dagger A_\ell^{r,a} &= \left(\langle a| \otimes I_{\mathcal{L}(\mathcal{H}_E)}\right) \left(\sum_{\ell=1}^{t} A_\ell^\dagger A_\ell\right) \left(|a\rangle \otimes I_{\mathcal{L}(\mathcal{H}_E)}\right) \\ &= I_{\mathcal{L}(\mathcal{H}_E)} \quad . \end{aligned} \tag{5.3}$$

We now consider the transformation obtained by connecting the circuit C through the channel $\mathcal{A}$. Recall the execution of C: it first measures the feedback input and obtains r, then makes the Bell-basis measurement and obtains j, and finally sets the feedback output to $a = r \oplus j$. By Equation (5.2), the operation applied to the external input in the branch with outcome j is VP_j/d. For this branch to contribute to the connected transformation, the channel $\mathcal{A}$ must take the feedback output $a = r \oplus j$ back to the feedback input r. According to the above notation, the corresponding operation on the auxiliary registers is $A_\ell^{r,r\oplus j}$, for some $\ell \in [t]$.

Consequently, the connected transformation has the following matrix representation. For $r, j \in \{0,1\}^{2n}$ and $\ell \in [t]$, define the matrices

$$K_{r,j,\ell} := \frac{1}{d} V P_j \otimes A_\ell^{r,r\oplus j} \ .$$

Since the feedback input and the Bell basis are measured during the execution of C, the different outcomes r, j contribute as separate branches. Therefore, denoting by $\mathcal{G}_\mathcal{A}$ the induced Process Matrix Operator, for every matrix X on the external input and the auxiliary input E we have

$$\mathcal{G}_\mathcal{A}(X) = \sum_{r,j\in\{0,1\}^{2n}} \sum_{\ell=1}^{t} K_{r,j,\ell} X K_{r,j,\ell}^\dagger \ .$$

In particular, this shows that $\mathcal{G}_\mathcal{A}$ is completely positive, also when its input is entangled with arbitrary additional registers. It remains to prove that the induced operator is trace-preserving.

Recall that V is an isometry and the matrices P_j are unitary. Thus $V^\dagger V = P_j^\dagger P_j = I_n$, and we have

$$\begin{aligned}
\sum_{r,j,\ell} K_{r,j,\ell}^\dagger K_{r,j,\ell} &= \frac{1}{d^2} I_{\mathcal{L}(\mathcal{S}(n))} \otimes \sum_{r,j,\ell} \left(A_\ell^{r,r\oplus j}\right)^\dagger A_\ell^{r,r\oplus j} \\
&= \frac{1}{d^2} I_{\mathcal{L}(\mathcal{S}(n))} \otimes \sum_{a\in\{0,1\}^{2n}} \sum_{r,\ell} \left(A_\ell^{r,a}\right)^\dagger A_\ell^{r,a} \\
&= \frac{1}{d^2} I_{\mathcal{L}(\mathcal{S}(n))} \otimes \sum_{a\in\{0,1\}^{2n}} I_{\mathcal{L}(\mathcal{H}_E)} \\
&= I_{\mathcal{L}(\mathcal{S}(n)\otimes\mathcal{H}_E)} \ .
\end{aligned}$$

In the second equality we simply used the change of variables $a = r \oplus j$, and the third equality follows from Equation (5.3). The last equality holds because there are $2^{2n} = d^2$ possible strings a. By Definition 3.16, this proves that the induced operator is trace-preserving. Combining this with complete positivity, we get that the Process Matrix Operator induced by C is a quantum channel for every $\mathcal{A}$ and arbitrary auxiliary registers E, E'. Thus, $(2n, C)$ is a PMG, and the stronger validity property in the statement of the lemma also holds.

The generated channel. We are left to show the action of the generated channel $\mathbf{PMO}_{(2n,C)}$, which is obtained by inserting the identity channel on the feedback registers without auxiliary input. In this case the inserted channel has the single matrix $A_1 = I_{\mathcal{L}(\mathcal{S}(2n))}$, and therefore $A_1^{r,a} = \mathbf{1}[r = a]$. Recall that the feedback output of C is $a = r \oplus j$, so the condition $r = a$ holds if and only if $j = 0^{2n}$. Thus, in the matrix representation of the derived channel, the only nonzero operators are

$$K_{r,0^{2n},1} = \frac{1}{d} V \ ,$$

one for every $r \in \{0,1\}^{2n}$. In particular, recalling that there are d^2 different values of r, for every quantum state ρ we have

$$\mathbf{PMO}_{(2n,C)}(\rho) = \sum_{r\in\{0,1\}^{2n}} \frac{1}{d^2} V\rho V^\dagger = V\rho V^\dagger \ .$$

This completes the proof. □

Purifiability of the construction. The stronger validity property in Lemma 5.1 also implies purifiability. We state the implication separately, since it applies beyond our particular construction.

Proposition 5.1 (Joint validity implies purifiability)**.** *Let $D = (n, C)$ be a PMG. Suppose that its induced transformation remains a quantum channel when an arbitrary quantum channel $\mathcal{A}$ acts jointly on all n feedback qubits, including arbitrary auxiliary input and output registers. Then D is purifiable in the sense of Definition 4.17.*

Proof. Write R, S for the feedback input and output of C, and P, F for its external input and output. Treat all feedback pairs as one pair, so that $\mathcal{A}$ acts from S to R, possibly together with auxiliary registers. Denote the resulting transformation by $\mathcal{G}_{\mathcal{A}}$.

The mapping $\mathcal{A} \mapsto \mathcal{G}_{\mathcal{A}}$ is a deterministic quantum supermap in the terminology of [CDP08]. Indeed, it is linear and completely positive in the CJ representation, since its Choi operator is $\mathbf{CJ}(\mathcal{C}_C)$ up to a reordering of registers. By our assumption, it maps every quantum channel to a quantum channel, including when auxiliary registers are present.

Theorem 1 of [CDP08] therefore gives a memory register M, an additional output register G, and isometries

$$B_1 : \mathcal{H}_P \longrightarrow \mathcal{H}_S \otimes \mathcal{H}_M \quad \text{and} \quad B_2 : \mathcal{H}_R \otimes \mathcal{H}_M \longrightarrow \mathcal{H}_F \otimes \mathcal{H}_G$$

such that, for every $\mathcal{A}$ without auxiliary registers,

$$\mathcal{G}_{\mathcal{A}}(X) = Tr_G \left[B_2 \left(\mathcal{A} \otimes \mathrm{id}_M \right) \left(B_1 X B_1^\dagger \right) B_2^\dagger \right] \ .$$

The same identity holds with arbitrary auxiliary registers, on which the surrounding operations act as the identity. Crucially, B_1 and B_2 are fixed independently of $\mathcal{A}$.

Extend B_1 and B_2 to unitaries by adding input registers K_1, K_2, respectively. Denote these unitaries by

$$U_1 : \mathcal{H}_P \otimes \mathcal{H}_{K_1} \longrightarrow \mathcal{H}_S \otimes \mathcal{H}_M \quad \text{and} \quad U_2 : \mathcal{H}_R \otimes \mathcal{H}_M \otimes \mathcal{H}_{K_2} \longrightarrow \mathcal{H}_F \otimes \mathcal{H}_G \ .$$

Memory and environment spaces may be enlarged to qubit registers as needed. Initializing K_i to zero recovers B_i. Define a unitary U from RPK_1K_2 to SFG by applying U_1 to PK_1, and then applying U_2 to RMK_2, leaving S untouched.

To verify process validity, insert an arbitrary joint channel $\mathcal{A}$ from SE to RE'. The connected transformation is an ordinary circuit: apply U_1, pass S through $\mathcal{A}$, and apply U_2 to the returned register R and the kept registers M, K_2. Every operation is a quantum channel, so the connected transformation is a quantum channel on every joint input to P, K_1, K_2, E. If $\mathcal{A}$ is unitary, every operation is unitary. In particular, this holds for every tuple of local unitary operations on the original separate feedback pairs. Thus, $\mathbf{CJ}(\mathrm{Ad}_U)$ is a pure process.

Finally, initialize K_1, K_2 to zero and discard G. By the realization above, this recovers $\mathcal{G}_{\mathcal{A}}$ for every $\mathcal{A}$, including its auxiliary extensions. This also recovers the entire original generating channel $\mathcal{C}_C$: choose $\mathcal{A}$ to swap S with an external n-qubit input, returning that input to R and sending S to an external output. With this choice, the connected transformation is precisely $\mathcal{C}_C$, up to register ordering. Consequently, for every matrix X on RP,

$$\mathcal{C}_C(X) = Tr_G \left[U \left(X \otimes |0\rangle\langle 0|_{K_1K_2} \right) U^\dagger \right] \ .$$

Taking $P' = K_1K_2$ and $F' = G$ proves purifiability. □

Applying Proposition 5.1 to the stronger validity guarantee in Lemma 5.1, every process obtained by that lemma is purifiable. In particular, this applies to the unique-witness and permutation-inversion constructions below. The argument establishes existence of a pure extension; it does not provide a polynomial-time algorithm for constructing its description from the supplied Choi-state preparation circuit.

5.2 Finding a unique satisfying input

Our next construction applies the previous lemma to the problem of finding the unique satisfying input of an efficiently computable Boolean function.

Theorem 5.1 (Unique-witness search)**.** *There exists a classical deterministic polynomial-time algorithm M that, given a Boolean circuit $v : \{0,1\}^k \to \{0,1\}$ with exactly one satisfying input z, computes a pair (n, C) for $n = O(k)$ which is a PMG and whose generated channel is unitary. Moreover, using one query to this generated channel, M recovers z with probability* 1.

Proof. Let v be a Boolean circuit as in the statement, with unique satisfying input z. Introduce one additional input bit and require it to be 1 for verification to pass. We continue to write v, k, z for the padded verifier, its input length, and its unique witness, respectively. Thus, $z \neq 0^k$. We discard the added bit when reporting the witness for the original instance. Denote $d := 2^k$. Consider the following unitary matrix $S_z \in \mathbb{C}^{d \times d}$:

$$S_z|x\rangle := \begin{cases} |z\rangle, & x = 0^k, \\ |0^k\rangle, & x = z, \\ |x\rangle, & \text{otherwise.} \end{cases}$$

Observe that S_z is unitary, as it is a permutation of the computational-basis vectors, and that $S_z|0^k\rangle = |z\rangle$. We will show how to generate the channel corresponding to S_z without first computing z.

By Lemma 5.1, it suffices to give an ordinary polynomial-size quantum circuit that prepares the state

$$|J_z\rangle := \frac{1}{\sqrt{d}} \left(\sum_{x \in \{0,1\}^k \setminus \{0^k, z\}} |x, x\rangle + |0^k, z\rangle + |z, 0^k\rangle \right) . \tag{5.4}$$

We describe such a circuit next.

Preparing the state $|J_z\rangle$. Start with the uniform superposition on $k+1$ qubits,

$$\frac{1}{\sqrt{2d}} \sum_{x \in \{0,1\}^k} \sum_{b \in \{0,1\}} |x, b\rangle .$$

Add registers A, C, G on k, k and 1 qubits respectively, initially in the all-zero state $|0^{2k+1}\rangle$. By ordinary reversible computation, evaluate $v(x)$ and compute the following (additions are modulo 2):

1. If $x \neq 0^k$, $v(x) = 0$, and $b = 0$, add (x, x) to (A, C) and 1 to G.
2. If $v(x) = 1$ and $b = 0$, add $(0^k, x)$ to (A, C) and 1 to G.
3. If $v(x) = 1$ and $b = 1$, add $(x, 0^k)$ to (A, C) and 1 to G.
4. In all other cases, leave (A, C, G) unchanged.

Observe that, on all branches with $G = 1$, the pair (A, C) uniquely determines the original input (x, b): if $A = 0^k$ then $(x, b) = (C, 0)$; if $C = 0^k$ then $(x, b) = (A, 1)$; and otherwise $(x, b) = (A, 0)$. Therefore, conditioned on $G = 1$, the original registers x, b and all computational work registers can be reversibly returned to the all-zero state, without changing (A, C).

There are exactly d successful pairs (x, b): one for each $x \notin \{0^k, z\}$ and two more for $x = z$. All successful pairs give different values of (A, C), and these values are exactly the d pairs appearing in Equation (5.4). Therefore, the probability that $G = 1$ is exactly $1/2$, and conditioned on this event the output registers are in state $|J_z\rangle$, with no entangled information left behind.

Recall that in our Lemma 5.1 we use a circuit that makes $|J_z\rangle$ with probability 1, and the above approach achieves probability $1/2$, and can get arbitrarily close to 1 by repetition, but not exactly. We next turn this constant-probability preparation to be exact.

Append an independent Hadamard bit $|+\rangle_T$ in a single-qubit register T, and call a branch good if $G = T = 1$. The probability of this event is exactly $1/4$. Let N be the total number of qubits used by this preparation, including all work registers. Denote the unitary computation described so far by A, let $|\Psi\rangle := A|0^N\rangle$, and let Π be the projector onto the good subspace. Then we can write

$$|\Psi\rangle := \frac{1}{2}|S_0\rangle + \frac{\sqrt{3}}{2}|S_1\rangle \ ,$$

for normalized orthogonal states S_0, S_1, where $|S_0\rangle$ has the output registers (A, C) in state $|J_z\rangle$ and every work register in a fixed (untangled) state. Direct calculation gives

$$(2|\Psi\rangle\langle\Psi| - I)\,(I - 2\Pi)|\Psi\rangle = |S_0\rangle \ . \tag{5.5}$$

Indeed, the right reflection produces $-|S_0\rangle/2 + \sqrt{3}|S_1\rangle/2$, whose inner product with $|\Psi\rangle$ is $1/2$. Reflecting this state about $|\Psi\rangle$ (which is the second operation) therefore gives $|S_0\rangle$. The first (rightmost) operation in Equation (5.5) is an ordinary phase flip controlled on $G = T = 1$ (i.e., apply a control Z two-qubit gate with control register G and target register T) and the second reflection is implemented by applying $A^\dagger$, then $2|0^N\rangle\langle 0^N| - I$, and finally A. Both are implementable by polynomial-size quantum circuits. Thus we obtain, without measurement or postselection, an exact polynomial-size preparation of $|J_z\rangle$.

Process generation and correctness. Apply Lemma 5.1 to this preparation circuit, with $V := S_z$. We obtain a polynomial-size ordinary quantum circuit C for which $(n, C) := (2k, C)$ is a PMG and whose generated channel is

$$\mathbf{PMO}_{(n,C)}(\rho) = S_z \rho S_z^\dagger \ .$$

The generated channel is unitary, even though the original circuit C may use measurements and discarded registers. By applying this generated channel to $|0^k\rangle\langle 0^k|$ and measuring the output, the algorithm recovers the unique satisfying input z with probability 1. The construction and every oracle use cost polynomial resources, which concludes the proof. □

5.3 Solving NP

We now use Theorem 5.1 and the Valiant–Vazirani reduction from Theorem 3.1 to solve SAT, and consequently all of **NP**, in $\mathbf{BQP}_{\ell\mathrm{CTC}}$. We apply our unique-witness construction to each reduced formula and verify the returned candidate assignment before accepting.

Theorem 5.2.

$$\boldsymbol{NP} \subseteq \boldsymbol{BQP}_{\ell CTC} \ .$$

The algorithm makes polynomially many nonadaptive generation queries. Whenever the formula used in a query has exactly one satisfying assignment, the query is a PMG and its generated channel is unitary.

Proof. It suffices to solve SAT. Let A be any generation oracle satisfying Definition 4.9, and let φ an input CNF formula with m input variables. If the input does not encode a CNF formula, reject. Formulas with no variables are evaluated directly. We describe a quantum polynomial-time Turing Machine M that, with access to A, decides whether $\varphi \in \mathsf{SAT}$.

Let $T := \lceil K|\varphi|^2 \rceil$, for a sufficiently large absolute constant K. The machine M executes the following procedure T times, with independent randomness for each execution of the Valiant-Vazirani reduction:

1. Execute $T_{VV}(\varphi)$ from Theorem 3.1, obtaining a CNF formula φ^*.

2. Apply the deterministic circuit construction from Theorem 5.1 to the Boolean circuit that verifies a satisfying assignment of φ^*. Denote the resulting pair by (n, C), and query $A(n, C)$.

3. Apply the returned channel to the designated all-zero input, and measure its output in the computational basis. Remove the extra witness bit introduced in the proof of Theorem 5.1, obtaining a candidate assignment w. Accept if $\varphi^*(w) = 1$.

If none of the iterations accepted, reject.

The construction in Step 2 outputs a well-formed circuit description for every φ^*. If φ^* has exactly one satisfying assignment, Theorem 5.1 guarantees that this description is a PMG whose generated channel finds the assignment.

Correctness. Suppose first that $\varphi \notin \mathsf{SAT}$. By Theorem 3.1, every formula φ^* generated in Step 1 is unsatisfiable. Therefore, no candidate assignment can pass Step 3, regardless of which channels A returns. The machine rejects with probability 1.

Now suppose that $\varphi \in \mathsf{SAT}$. There is a constant $c > 0$ such that each execution of Step 1 produces a uniquely satisfiable formula with probability at least c/m. Whenever this event occurs, Theorem 5.1 guarantees that Step 2 supplies the channel that recovers the unique satisfying assignment. Step 3 therefore accepts with probability 1. Consequently, the probability that all T iterations reject is at most

$$\left(1 - \frac{c}{m}\right)^T \leq \exp\left(-\frac{cT}{m}\right) \quad .$$

Since $m \leq |\varphi|$, choosing K sufficiently large gives

$$\exp\left(-\frac{cT}{m}\right) \leq \exp\left(-cK|\varphi|\right) \leq 2^{-|\varphi|} \quad .$$

Running time and generation queries. The reduction, the circuit construction, and the final verification all take polynomial time. Every generation query and every use of its returned channel have polynomial cost under Definition 4.10. As a side note, one can also verify that the generation queries are nonadaptive. This proves $\mathsf{SAT} \in \mathbf{BQP}_{\ell\mathrm{CTC}}$. □

5.4 Inverting an efficiently computable permutation

We end with an application of Lemma 5.1 to computing a function, rather than finding a single witness. The construction uses the same computational model $\mathbf{BQP}_{\ell\mathrm{CTC}}$, and provides a unitary that inverts a permutation on arbitrary quantum inputs.

Theorem 5.3 (Permutation inversion)**.** *There exists a deterministic polynomial-time algorithm that, given an ordinary classical circuit for a permutation $\pi : \{0,1\}^n \to \{0,1\}^n$, outputs a PMG whose generated channel is the unitary transformation*

$$|y\rangle \longmapsto |\pi^{-1}(y)\rangle \quad . \tag{5.6}$$

In particular, this transformation acts coherently on arbitrary input superpositions and inputs entangled with additional registers.

Proof. Let π be a permutation with polynomial-size evaluation circuit. The strategy is to use Lemma 5.1 by a simply circuit that generates the Choi state of the inverse π^{-1}.

Prepare the uniform superposition over x, compute $\pi(x)$ into a fresh register and SWAP the two registers. Since π is a permutation, the same state equals

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |\pi(x)\rangle \otimes |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} |y\rangle \otimes |\pi^{-1}(y)\rangle = |J_{U_{\pi^{-1}}}\rangle \ ,$$

where $U_{\pi^{-1}}|y\rangle := |\pi^{-1}(y)\rangle$ is unitary. Applying Lemma 5.1 gives a polynomial-size PMG with generated channel $U_{\pi^{-1}}$. The computational model therefore gives access to this channel. This proves the theorem. □

6 Quantum Computation with Efficient and Pure Process Matrices

In this section we construct pure processes, with unitary generating circuits constructed efficiently from the input. We first prove a general result on quantum changes of basis and then use it to invert USO outmaps. This gives our algorithm for USO and its consequences through classical reductions. We end by discussing connections to physical implementation.

6.1 A unitary generator for the USO Oracle

We recall the unique-sink orientation problem, introduced in [SW01]. The input describes an exponentially large graph by a circuit which computes the directions of the edges incident to a given vertex.

Definition 6.1 (Unique-Sink Orientation (USO) Problem)**.** *Let $n \in \mathbb{N}$. The n-dimensional Boolean cube has vertex set $\{0,1\}^n$, where two vertices are adjacent if they differ in exactly one coordinate. For every $i \in [n]$, let $e_i \in \{0,1\}^n$ be the string whose i-th coordinate is 1 and all other coordinates are 0. An orientation of the cube assigns a direction to every edge, and is represented by a function $s : \{0,1\}^n \to \{0,1\}^n$, where $s_i(x) = 1$ if the edge between x and $x \oplus e_i$ is directed away from x, and $s_i(x) = 0$ if it is directed to x. In particular, $s_i(x) \neq s_i(x \oplus e_i)$ for every x and i. We call s the outmap of the orientation.*

A face of the cube is obtained by fixing an arbitrary subset $F \subseteq [n]$ of coordinates to specified values. A sink of a face is a vertex of that face with no outgoing edges along any of its free coordinates. An orientation is called a unique-sink orientation (USO) if every face has exactly one sink. This includes the entire cube, obtained by fixing no coordinates, and faces consisting of a single vertex, obtained by fixing all coordinates (which trivially satisfies the requirement, because there are no free coordinates in that case).

In the computational problem USO*, the input is a classical Boolean circuit computing an outmap s, promised that the corresponding orientation is a USO. The output is the unique vertex $z \in \{0,1\}^n$ satisfying $s(z) = 0^n$. Running times are measured in the description length of the input circuit.*

The following characterization of unique-sink orientations is given in [SW01, Lemma 2.3]. We also state an equivalent formulation that will be useful for our constructions. For completeness, we provide a proof in Appendix A.1.[8]

Lemma 6.1 (Pairwise characterization of USO)**.** *A function $s : \{0,1\}^n \to \{0,1\}^n$ is a USO outmap if and only if*

$$x \neq x' \quad \Longrightarrow \quad \exists i \in [n] : \ x_i \neq x'_i \ \text{ and } \ s_i(x) \neq s_i(x') \ . \tag{6.1}$$

In particular, every USO outmap is a permutation of $\{0,1\}^n$. For $h(x) := x \oplus s(x)$, Equation (6.1) is equivalent to

$$x \neq x' \quad \Longrightarrow \quad \exists i \in [n] : \ x_i \neq x'_i \ \text{ and } \ h_i(x) = h_i(x') \ . \tag{6.2}$$

[8] The original paper does not provide a proof, but this combinatorial fact is well-known and folklore.

Main technical theorem. We next construct a pure PMG for a family of quantum changes of basis. For each computational-basis label $|x\rangle$, consider a product state obtained by applying a single-qubit unitary at each coordinate, choosing between two gates according to $h(x)$. The generated transformation will map this state back to $|x\rangle$. The condition in Equation (6.2) guarantees both that these states form an orthonormal basis and that our generating circuit describes a pure process. Our algorithm for USO will follow as a special case.

Theorem 6.1 (Generating a USO-decomposable unitary by a pure PMG)**.** *Let $h : \{0,1\}^n \to \{0,1\}^n$ be given by a classical Boolean circuit satisfying Equation (6.2). Let $\{U_{i,a}\}_{i\in[n],a\in\{0,1\}}$ a set of single-qubit unitaries, each given by an ordinary unitary circuit. For every $x \in \{0,1\}^n$ define*

$$U(x) := \bigotimes_{i=1}^{n} U_{i,h_i(x)} \quad \text{and} \quad |b_x\rangle := U(x)^\dagger |x\rangle \ .$$

Then, the following holds.

- *The vectors $\{|b_x\rangle\}_{x\in\{0,1\}^n}$ form an orthonormal basis of $\mathcal{S}(n)$ and thus the matrix*

$$T := \sum_{x\in\{0,1\}^n} |x\rangle\langle b_x| = \sum_{x\in\{0,1\}^n} |x\rangle\langle x|U(x) \tag{6.3}$$

  is unitary.

- *Moreover, there is a classical deterministic polynomial-time algorithm that, given the circuits $\{U_{i,a}\}_{i\in[n],a\in\{0,1\}}$ and h, outputs a pair (n, C) which is a pure PMG and whose generated Process Matrix Operator unitary (as in Definition 4.13) is the matrix T.*

Proof. We first verify the claim about the basis. Every $|b_x\rangle$ has norm one because $U(x)$ is unitary. For different x, x', choose i as in Equation (6.2). In the inner product $\langle b_x|b_{x'}\rangle$, the factor at this coordinate is

$$\langle x_i|U_{i,h_i(x)}U^\dagger_{i,h_i(x')}|x'_i\rangle = \langle x_i|x'_i\rangle = 0 \ .$$

Thus the vectors are orthonormal, and there are 2^n of them in $\mathcal{S}(n)$.

The process matrix generator. We describe a polynomial-size unitary quantum circuit C acting on 2 registers R, Q, each having n qubits. Specifically, let $R = R_1 \cdots R_n$ be the n-qubit feedback input register with one qubit in each R_i, and let Q be the n-qubit external input register. On a basis input $|x\rangle_R$ and an arbitrary quantum state $|\psi\rangle_Q$, the circuit C performs the following operations:

1. Compute $h(x)$ in an initially all-zero, n-qubit work register W.
2. For every $i \in [n]$, apply $U_{i,h_i(x)}$ to qubit i of Q, controlled on the corresponding bit of W.
3. Reverse the computation of $h(x)$, returning W to its all-zero state. The register R has not been changed, so this uncomputation does not require knowing the contents of Q.
4. Swap R and Q.

Consequently, the channel of C is the unitary U_C given by

$$U_C\left(|x\rangle_R \otimes |\psi\rangle_Q\right) = \left(U(x)|\psi\rangle\right)_R \otimes |x\rangle_Q \ . \tag{6.4}$$

Validity of the generator. We prove the equivalent condition in Definition 4.13. Let $E_1, \ldots, E_n$ be arbitrary additional registers, and let V_i be an arbitrary unitary on R_iE_i. Write $E := E_1, \cdots, E_n$. Recall

again that these registers and unitaries are used only in the proof of validity, the algorithm is not given their descriptions and need not implement them. Our goal is to show that the matrix

$$T_{\mathbf{V}} := Tr_R \left[\left(\bigotimes_{i=1}^{n} V_i \right) \left(U_C \otimes I_{\mathcal{L}(\mathcal{H}_E)} \right) \right]$$

is a unitary transformation on the registers Q, E.

Our first step will be to look on the entire transformation before applying the partial trace. Let $r \in \{0,1\}^n$, $y \in \{0,1\}^n$ be inputs to the R and Q registers respectively and for $i \in [n]$ let $w_i \in \{0,1\}^{|E_i|}$ be an input to the E register. We have the equalities

$$\begin{aligned}
&\left(\bigotimes_{i=1}^{n} V_i \right) \left(U_C \otimes I_{\mathcal{L}(\mathcal{H}_E)} \right) \left(|r\rangle_R \otimes |y\rangle_Q \otimes |w\rangle_E \right) \\
&\quad \underset{\text{Equation (6.4)}}{=} \left(\bigotimes_{i=1}^{n} V_i \right) \left(\left(U(r)|y\rangle \right)_R \otimes |r\rangle_Q \otimes |w\rangle_E \right) \\
&\quad \underset{\text{Def. of } U(r)}{=} \left(\bigotimes_{i=1}^{n} V_i \right) \left(\left(\bigotimes_{i=1}^{n} U_{i,h_i(r)} |y_i\rangle \right)_R \otimes |r\rangle_Q \otimes |w\rangle_E \right) .
\end{aligned} \tag{6.5}$$

We next want to trace out the R register from the above transformation.

Writing the state as a standard-basis superposition over the feedback register. To compute the partial trace explicitly, we first expand the feedback output in the computational basis. This will let us identify the component which stays in each summand of the partial trace. Let us focus on registers R and E in Equation 6.5 and stop thinking about this as the output of a computation, and just as a state – we seek to write this state in a way that the R register contains a superposition of computational-basis vectors. Even more specifically, the state is enumerated by $r \in \{0,1\}^n$ which we can fix, so what remains in registers R, E is

$$\bigotimes_{i=1}^{n} V_i \cdot \left(U_{i,h_i(r)} |y_i\rangle_{R_i} \otimes |w_i\rangle_{E_i} \right) ,$$

where for each $i \in [n]$, we define $w_i \in \{0,1\}^{|E_i|}$ as the disjoint i-th part of the string $w \in \{0,1\}^{|E|}$.

Next, for each value $i \in [n]$, $a \in \{0,1\}$, define the unitary

$$B_{i,a} := V_i \left(U_{i,a} \otimes I_{\mathcal{L}(\mathcal{H}_{E_i})} \right) .$$

For bits $u, v \in \{0,1\}$, let $B_{i,a}^{u,v}$ be the matrix on E_i obtained by fixing the input feedback bit of $B_{i,a}$ to v and its output feedback bit to u. Formally,

$$B_{i,a}^{u,v} := \left(\langle u| \otimes I_{\mathcal{L}(\mathcal{H}_{E_i})} \right) B_{i,a} \left(|v\rangle \otimes I_{\mathcal{L}(\mathcal{H}_{E_i})} \right) .$$

Equivalently, $B_{i,a} = \sum_{u,v} |u\rangle\langle v| \otimes B_{i,a}^{u,v}$. We deduce the equality:

$$\bigotimes_{i=1}^{n} V_i \cdot \left(U_{i,h_i(r)} |y_i\rangle_{R_i} \otimes |w_i\rangle_{E_i} \right) = \sum_{z \in \{0,1\}^n} |z\rangle_R \otimes \left(\bigotimes_{i=1}^{n} B_{i,h_i(r)}^{z_i,y_i} \right) |w\rangle_E ,$$

which overall, given Equation 6.5 implies

$$\begin{aligned}
&\left(\bigotimes_{i=1}^{n} V_i \right) \left(U_C \otimes I_{\mathcal{L}(\mathcal{H}_E)} \right) \left(|r\rangle_R \otimes |y\rangle_Q \otimes |\eta\rangle_E \right) \\
&\qquad = \sum_{z \in \{0,1\}^n} |z\rangle_R \otimes |r\rangle_Q \otimes \left(\bigotimes_{i=1}^{n} B_{i,h_i(r)}^{z_i,y_i} \right) |\eta\rangle_E .
\end{aligned}$$

Therefore, when we take the partial trace, only the term $z = r$ survives. The external output $|r\rangle_Q$ remains, together with the matrix on E multiplying that term. Summing over all feedback input labels r, we obtain

$$T_{\mathbf{V}}\left(|y\rangle_Q \otimes |\eta\rangle_E\right) = \sum_{r\in\{0,1\}^n} |r\rangle_Q \otimes \left(\bigotimes_{i=1}^{n} B_{i,h_i(r)}^{r_i,y_i}\right)|\eta\rangle_E \ . \tag{6.6}$$

Proving unitarity by block-wise analysis. We would next like to prove that $T_{\mathbf{V}} \cdot T_{\mathbf{V}}^\dagger = I_{\mathcal{L}(\mathcal{H}_Q\otimes\mathcal{H}_E)}$. One view that Equation 6.6 allows is that the input and output of the Q register completely determines the transformation applied to the remaining system E. Accordingly we will analyze $T_{\mathbf{V}} \cdot T_{\mathbf{V}}^\dagger$ block-wise, partitioned into blocks fixing the input-output of Q.

For $x, y \in \{0,1\}^n$, define the matrix $(T_{\mathbf{V}})_{x,y} \in \mathcal{L}(\mathcal{H}_E)$ by

$$(T_{\mathbf{V}})_{x,y} := \left(\langle x|_Q \otimes I_{\mathcal{L}(\mathcal{H}_E)}\right) T_{\mathbf{V}} \left(|y\rangle_Q \otimes I_{\mathcal{L}(\mathcal{H}_E)}\right) \ .$$

Thus, x and y refer only to the external output and input registers, respectively. It follows that

$$(T_{\mathbf{V}})_{x,y} = \sum_{r\in\{0,1\}^n} \langle x|r\rangle \bigotimes_{i=1}^{n} B_{i,h_i(r)}^{r_i,y_i} = \bigotimes_{i=1}^{n} B_{i,h_i(x)}^{x_i,y_i} \ .$$

We next compute the Q-blocks of $T_{\mathbf{V}}T_{\mathbf{V}}^\dagger$. For every $x, x' \in \{0,1\}^n$,

$$\begin{aligned}
\left(T_{\mathbf{V}}T_{\mathbf{V}}^\dagger\right)_{x,x'} &= \sum_{y\in\{0,1\}^n} (T_{\mathbf{V}})_{x,y} \left((T_{\mathbf{V}})_{x',y}\right)^\dagger \\
&= \sum_{y\in\{0,1\}^n} \bigotimes_{i=1}^{n} \left[B_{i,h_i(x)}^{x_i,y_i} \left(B_{i,h_i(x')}^{x'_i,y_i}\right)^\dagger\right] \\
&= \bigotimes_{i=1}^{n} \left[\sum_{v\in\{0,1\}} B_{i,h_i(x)}^{x_i,v} \left(B_{i,h_i(x')}^{x'_i,v}\right)^\dagger\right] \ .
\end{aligned} \tag{6.7}$$

In the last equality, we used that, for fixed x, x', the i-th factor depends on y only through its i-th bit. Recall that for all i, a the matrix $B_{i,a}$ is unitary and thus $B_{i,a}B_{i,a}^\dagger = I_{1+|E_i|}$. For $u, u' \in \{0,1\}$, taking the (u,u') part of the equality $B_{i,a}B_{i,a}^\dagger = I$ gives

$$\sum_{v\in\{0,1\}} B_{i,a}^{u,v} \left(B_{i,a}^{u',v}\right)^\dagger = \mathbf{1}[u = u'] I_{\mathcal{L}(\mathcal{H}_{E_i})} \ . \tag{6.8}$$

Back to analyzing $\left(T_{\mathbf{V}}T_{\mathbf{V}}^\dagger\right)_{x,x'}$. If $x = x'$, Equation (6.8) implies that every factor in Equation (6.7) is $I_{\mathcal{L}(\mathcal{H}_{E_i})}$, so the resulting matrix is $I_{\mathcal{L}(\mathcal{H}_E)}$. Otherwise, Equation (6.2) gives an index i such that $x_i \neq x'_i$ and $h_i(x) = h_i(x')$. Write $a := h_i(x) = h_i(x')$. At this coordinate, the factor is

$$\sum_{v\in\{0,1\}} B_{i,a}^{x_i,v} \left(B_{i,a}^{x'_i,v}\right)^\dagger = 0 \ ,$$

where the equality follows from Equation (6.8), because $x_i \neq x'_i$. Hence the entire tensor product is zero. We have proved that the diagonal matrices of $T_{\mathbf{V}}T_{\mathbf{V}}^\dagger$ are identities and all its off-diagonal matrices are zero. Consequently,

$$T_{\mathbf{V}}T_{\mathbf{V}}^\dagger = I_{\mathcal{L}(\mathcal{H}_Q\otimes\mathcal{H}_E)} \ .$$

Since the additional registers E_i and the unitaries V_i were arbitrary, this proves the equivalent condition in Definition 4.13. Together with the unitarity of C, we conclude that (n, C) is a pure PMG.

The generated channel. For the identity insertion, Lemma 4.1 identifies the generated unitary with $Tr_R(U_C)$. For every $y \in \{0,1\}^n$, its action is

$$\begin{aligned}(Tr_R(U_C))|y\rangle &= \sum_{x\in\{0,1\}^n} (\langle x| \otimes I_Q) U_C (|x\rangle \otimes |y\rangle) \\ &= \sum_{x\in\{0,1\}^n} \langle x|U(x)|y\rangle |x\rangle = T|y\rangle \ ,\end{aligned}$$

for the matrix T from the statement of Theorem 6.1. This proves Equation (6.3). In particular, the generated operation sends $\sum_x \alpha_x |b_x\rangle$ to $\sum_x \alpha_x |x\rangle$, and the same equality holds with an identity operation on any untouched reference register. This concludes the proof. □

The special case of USO. Taking $U_{i,0} = I_1$ and $U_{i,1} = X$ in the preceding theorem gives the following result.

Theorem 6.2 (Coherent inversion of a USO outmap)**.** *There is a classical deterministic polynomial-time algorithm which, given a circuit for a USO outmap $s : \{0,1\}^n \to \{0,1\}^n$, outputs a pure PMG (n, C_s) (w.r.t. Definition 4.13) whose generated unitary is*

$$T_s := \sum_{x\in\{0,1\}^n} |x\rangle\langle s(x)| \quad \textit{and hence} \quad T_s|y\rangle = |s^{-1}(y)\rangle \ . \tag{6.9}$$

In particular, one query to the generated channel on $|0^n\rangle$ finds the unique sink with probability 1.

Proof. Compute $h(x) = x \oplus s(x)$ by an ordinary Boolean circuit. By Lemma 6.1, h satisfies Equation (6.2). For the gates $U_{i,a} = X^a$, the circuit in Theorem 6.1 is

$$U_{C_s}(|x\rangle_R |y\rangle_Q) = |y \oplus x \oplus s(x)\rangle_R |x\rangle_Q \ . \tag{6.10}$$

It is a pure PMG by that theorem. Moreover, $|b_x\rangle = |x \oplus h(x)\rangle = |s(x)\rangle$, which proves Equation (6.9). The computational model gives access to the generated channel T_s. Applying it to $|0^n\rangle$ gives $|s^{-1}(0^n)\rangle$, the unique sink. □

For general $U_{i,a}$, Theorem 6.1 instead gives a quantum basis transformation. Thus the same validity argument supports both the permutation inverse and the quantum constructions below. The relation between process functions and orthogonal product bases has been studied in [KB23]; here we specify a succinct circuit representation and prove the corresponding algorithm in $\mathbf{BQP}_{u\mathrm{CTC}}$.

Corollary 6.1 (Consequences of classical reductions)**.** *Every problem with a classical polynomial-time promise-preserving reduction to circuit-described Cube-USO has a polynomial-time algorithm using pure process computation.*

Proof. Compute the outmap circuit supplied by the reduction, use Theorem 6.2 to find its sink, and decode the solution. For a polynomial-time Turing reduction, apply the same procedure to each USO instance it produces. On a promised source input, each such instance satisfies the USO promise, so our construction supplies a pure PMG for every call. □

Known reductions therefore give polynomial-time algorithms using pure process computation for rational P-matrix linear complementarity [SW01], its generalized and P–Lin-Bellman formulations, and Grid-USO with total-side-length encoding [BFG+24]. They also cover simple stochastic, discounted,

mean-payoff, and parity games [GR05, JS08, FJS09], quantitative stochastic parity games [BKZ25], and finite-state modal μ-calculus model checking [Wil01]. The game problems here have explicitly represented finite arenas and are turn-based, fully observed, two-player, and zero-sum, with binary rational numerical data where applicable. Model checking takes an explicitly represented finite transition system and a formula as input. We use these classical reductions directly; the process construction needed for each resulting USO instance is the one in Theorem 6.2.

6.2 Connections to physical implementation

Theorem 6.2 gives an efficient generating circuit for a pure process that solves USO. Whether pure processes can be implemented physically with resources polynomial in the size of their generating circuits remains unknown. The structure of our construction gives concrete connections to existing circuit simulations and proposals for physical implementation, which we discuss below.

For our algorithm, a physical implementation would need to provide the generated channel, obtained when all local operations are identities. Reproducing the entire process under arbitrary local operations is a stronger requirement. This distinction already appears in [AGB17, Section VI]: they construct a family of pure processes that violate causal inequalities, yet whose connected channels admit efficient ordinary-circuit simulations when the local operations are given by circuit descriptions. Their generators use controlled bit flips followed by a SWAP, as in our construction.

Our more general construction also has a concrete connection to work on quantum control of causal order. Recall that Theorem 6.1 generates the transformation

$$T|b_x\rangle = |x\rangle \quad \text{where} \quad |b_x\rangle = \bigotimes_{i=1}^{n} U^{\dagger}_{i,h_i(x)}|x_i\rangle \ .$$

Each basis vector is a product state, with only two possible single-qubit operations at each coordinate. For example, take $n = 3$ and

$$h\left(x\right) = \left(\left(1 - x_2\right) x_3,\, x_1 \left(1 - x_3\right),\, \left(1 - x_1\right) x_2\right) \ .$$

With $U_{i,0} = I$ and $U_{i,1} = H$, these vectors are exactly the SHIFT basis studied in [KB23]. Steffinlongo and Dourdent [SD26] show how a quantum switch can implement the corresponding projective measurement and propose a photonic realization. Thus, a specific case of our basis construction already has a switch-based implementation proposal. The proposal concerns measurement in this basis, rather than the coherent transformation T.

These connections suggest studying physical implementations of the generated transformation directly. For USO, the target is particularly simple to state: take $U_{i,a} = X^a$, so that $|b_x\rangle = |s\left(x\right)\rangle$, and implement the map $|s\left(x\right)\rangle \mapsto |x\rangle$. The same two single-qubit gates are used for every instance; the instance enters through the efficiently computable function h. Whether this structure permits a physical implementation with resources polynomial in the description of the instance remains open. Existing simulations and switch-based proposals establish particular cases, but do not provide such a construction for arbitrary USO instances.

Acknowledgements. The main ideas in this paper were developed by the author. During its preparation, the author used ChatGPT (OpenAI) to help work through and check mathematical arguments, draft and revise proofs, explore the literature, and improve the presentation. The author takes full responsibility for all statements and proofs in this paper.

References

[Aar05] Scott Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proceedings of the Royal Society A*, 461:3473–3482, 2005.

[ACB14] Mateus Araújo, Fabio Costa, and Časlav Brukner. Computational advantage from quantum-controlled ordering of gates. *Physical review letters*, 113(25):250402, 2014.

[AFNB17] Mateus Araújo, Adrien Feix, Miguel Navascués, and Časlav Brukner. A purification postulate for quantum mechanics with indefinite causal order. *Quantum*, 1:10, 2017.

[AGB17] Mateus Araújo, Philippe Allard Guérin, and Ämin Baumeler. Quantum computation with indefinite causal structures. *Physical Review A*, 96(5):052315, 2017.

[AMP24] Alastair A. Abbott, Mehdi Mhalla, and Pierre Pocreau. Quantum query complexity of boolean functions under indefinite causal order. *Physical Review Research*, 6(3):L032020, 2024.

[AMP25] Alastair A. Abbott, Mehdi Mhalla, and Pierre Pocreau. Classical and quantum query complexity of boolean functions under indefinite causal order. In *Proceedings of the 22nd International Conference on Quantum Physics and Logic*, volume 426 of *Electronic Proceedings in Theoretical Computer Science*, pages 270–300, 2025.

[AQWR24] Michael Antesberger, Marco Túlio Quintino, Philip Walther, and Lee A. Rozema. Higher-order process matrix tomography of a passively-stable quantum switch. *PRX Quantum*, 5(1):010325, 2024.

[AW09] Scott Aaronson and John Watrous. Closed timelike curves make quantum and classical computing equivalent. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 465(2102):631–647, 2009.

[Bac17] Dave Bacon. Quantum approach to the unique sink orientation problem. *Physical Review A*, 96(1):012323, 2017.

[BCD+11] Luboš Brim, Jakub Chaloupka, Laurent Doyen, Raffaella Gentilini, and Jean-François Raskin. Faster algorithms for mean-payoff games. *Formal Methods in System Design*, 38(2):97–118, 2011.

[BCR+19] Ämin Baumeler, Fabio Costa, Timothy C. Ralph, Stefan Wolf, and Magdalena Zych. Reversible time travel with freedom of choice. *Classical and Quantum Gravity*, 36(22):224002, 2019.

[BFG+24] Michaela Borzechowski, John Fearnley, Spencer Gordon, Rahul Savani, Patrick Schnider, and Simon Weber. Two choices are enough for P-LCPs, USOs, and colorful tangents. In *Proceedings of ICALP*, 2024. Full version: arXiv:2402.07683v2.

[BKM+25] Jessica Bavaresco, Hlér Kristjánsson, Mio Murao, Tatsuki Odake, Marco Túlio Quintino, Philip Taranto, and Satoshi Yoshida. Simulating the quantum switch with quantum circuits is computationally hard. *Nature Communications*, 16:10216, 2025.

[BKZ25] Raphaël Berthon, Joost-Pieter Katoen, and Zihan Zhou. A direct reduction from stochastic parity games to simple stochastic games. In *Proceedings of CONCUR*, 2025. Full version: arXiv:2506.06223.

[BW12] Todd A Brun and Mark M Wilde. Perfect state distinguishability and computational speedups with postselected closed timelike curves. *Foundations of Physics*, 42:341–361, 2012.

[BW17] Ämin Baumeler and Stefan Wolf. Non-causal computation. *Entropy*, 19(7):326, 2017.

[BW18] Ämin Baumeler and Stefan Wolf. Computational tameness of classical non-causal models. *Proceedings of the Royal Society A*, 474(2209):20170698, 2018.

[CBB+21] Giulio Chiribella, Manik Banik, Some Sankar Bhattacharya, Tamal Guha, Mir Alimuddin, Arup Roy, Sutapa Saha, Sristy Agrawal, and Guruprasad Kar. Indefinite causal order enables perfect quantum communication with zero capacity channels. *New Journal of Physics*, 23(3):033039, 2021.

[CDP08] Giulio Chiribella, Giacomo Mauro D'Ariano, and Paolo Perinotti. Transforming quantum operations: Quantum supermaps. *EPL*, 83(3):30004, 2008.

[CDPV13] Giulio Chiribella, Giacomo Mauro D'Ariano, Paolo Perinotti, and Benoit Valiron. Quantum computations without definite causal structure. *Physical Review A—Atomic, Molecular, and Optical Physics*, 88(2):022318, 2013.

[CGK+18] Edmund M. Clarke, Jr., Orna Grumberg, Daniel Kroening, Doron Peled, and Helmut Veith. *Model Checking*. MIT Press, 2 edition, 2018.

[Chi12] Giulio Chiribella. Perfect discrimination of no-signalling channels via quantum superposition of causal structures. *Physical Review A—Atomic, Molecular, and Optical Physics*, 86(4):040301, 2012.

[Deu91] David Deutsch. Quantum mechanics near closed timelike lines. *Physical Review D*, 44(10):3197, 1991.

[DKSS14] Tushar Deshpande, Panagiotis Katsaros, Scott A. Smolka, and Scott D. Stoller. Stochastic game-based analysis of the DNS bandwidth amplification attack using probabilistic model checking. In *Proceedings of the 10th European Dependable Computing Conference (EDCC)*, pages 226–237. IEEE, 2014.

[ESC18] Daniel Ebler, Sina Salek, and Giulio Chiribella. Enhanced communication with the assistance of indefinite causal order. *Physical review letters*, 120(12):120502, 2018.

[FJS09] John Fearnley, Marcin Jurdziński, and Rahul Savani. Linear complementarity algorithms for infinite games. arXiv:0909.5653, 2009.

[FV20] David Felce and Vlatko Vedral. Quantum refrigeration with indefinite causal order. *Physical review letters*, 125(7):070603, 2020.

[GFAB16] Philippe Allard Guérin, Adrien Feix, Mateus Araújo, and Časlav Brukner. Exponential communication complexity advantage from quantum superposition of the direction of communication. *Physical review letters*, 117(10):100502, 2016.

[GGK+18] Kaumudibikash Goswami, Christina Giarmatzi, Michael Kewming, Fabio Costa, Cyril Branciard, Jacquiline Romero, and Andrew G White. Indefinite causal order in a quantum switch. *Physical review letters*, 121(9):090503, 2018.

[GR05] Bernd Gärtner and Leo Rüst. Simple stochastic games and P-matrix generalized linear complementarity problems. In *Proceedings of FCT*, 2005.

[GTW+26] Yu Guo, Hao Tang, Bo-Xuan Wang, Min-Yu Lv, Jia-Wen Fan, Xiao-Min Hu, Yun-Feng Huang, Chuan-Feng Li, Guang-Can Guo, Giulio Chiribella, and Bi-Heng Liu. Experimental violation of a bell-like inequality for causal order. *Science Advances*, 12(24):eaee2912, 2026.

[Har05] Lucien Hardy. Probability theories with dynamic causal structure: a new framework for quantum gravity. *arXiv preprint gr-qc/0509120*, 2005.

[Har07] Lucien Hardy. Towards quantum gravity: a framework for probabilistic theories with non-fixed causal structure. *Journal of Physics A: Mathematical and Theoretical*, 40(12):3081–3099, 2007.

[Har09] Lucien Hardy. Quantum gravity computers: On the theory of computation with indefinite causal structure. In *Quantum reality, relativistic causality, and closing the epistemic circle: essays in honour of Abner Shimony*, pages 379–401. Springer, 2009.

[JS08] Marcin Jurdziński and Rahul Savani. A simple P-matrix linear complementarity problem for discounted games. In *Proceedings of CiE*, volume 5028 of *LNCS*, pages 283–293, 2008.

[Kar10] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 2010.

[KB23] Ravi Kunjwal and Ämin Baumeler. Trading causal order for locality. *Physical Review Letters*, 131:120201, 2023.

[KPW18] Marta Kwiatkowska, David Parker, and Clemens Wiltsche. PRISM-games: Verification and strategy synthesis for stochastic multi-player games with multiple objectives. *International Journal on Software Tools for Technology Transfer*, 20(2):195–210, 2018.

[LMGP+11a] Seth Lloyd, Lorenzo Maccone, Raul Garcia-Patron, Vittorio Giovannetti, and Yutaka Shikano. Quantum mechanics of time travel through post-selected teleportation. *Physical Review D—Particles, Fields, Gravitation, and Cosmology*, 84(2):025007, 2011.

[LMGP+11b] Seth Lloyd, Lorenzo Maccone, Raul Garcia-Patron, Vittorio Giovannetti, Yutaka Shikano, Stefano Pirandola, Lee A Rozema, Ardavan Darabi, Yasaman Soudagar, Lynden K Shalm, et al. Closed timelike curves via postselection: theory and experimental test of consistency. *Physical review letters*, 106(4):040403, 2011.

[OCB12] Ognyan Oreshkov, Fabio Costa, and Časlav Brukner. Quantum correlations with no causal order. *Nature communications*, 3(1):1092, 2012.

[QLX+26] Dengke Qu, Quan Lin, Lei Xiao, Xiang Zhan, and Peng Xue. Experimental violation of a bell-like causal inequality in a photonic quantum switch. *Science Advances*, 12(32):eaee9271, 2026.

[RAC+26] Carla M. D. Richter, Michael Antesberger, Huan Cao, Philip Walther, and Lee A. Rozema. Toward an experimental device-independent verification of indefinite causal order. *PRX Quantum*, 7(1):010354, 2026.

[RB22] Martin J. Renner and Časlav Brukner. Computational advantage from a quantum superposition of qubit gate orders. *Physical Review Letters*, 128(23):230503, 2022.

[RRF+17] Giulia Rubino, Lee A Rozema, Adrien Feix, Mateus Araújo, Jonas M Zeuner, Lorenzo M Procopio, Časlav Brukner, and Philip Walther. Experimental verification of an indefinite causal order. *Science advances*, 3(3):e1602589, 2017.

[SD26] Anna Steffinlongo and Hippolyte Dourdent. Simulating noncausality with quantum control of causal orders. *Physical Review Research*, 8:013127, 2026.

[SW01] Tibor Szabó and Emo Welzl. Unique sink orientations of cubes. In *Proceedings of FOCS*, pages 547–555, 2001.

[VV85] Leslie G Valiant and Vijay V Vazirani. Np is as easy as detecting unique solutions. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, pages 458–463, 1985.

[Wil01] Thomas Wilke. Alternating tree automata, parity games, and modal μ-calculus. *Bulletin of the Belgian Mathematical Society – Simon Stevin*, 8, 2001.

[ZYC20] Xiaobin Zhao, Yuxiang Yang, and Giulio Chiribella. Quantum metrology with indefinite causal order. *Physical Review Letters*, 124(19):190503, 2020.

A Supplementary Proofs

We prove the pairwise characterization of USO, show that the USO property is necessary for our particular generating circuit, and bound the effect of approximating the single-qubit gates in our more general construction.

A.1 The USO characterization and validity of our construction

The characterization in Lemma 6.1 compares pairs of vertices. We first show that, on every face of a USO, the outmap restricted to the free coordinates takes each possible value exactly once. The pairwise condition will then follow by considering the smallest face containing the two vertices.

Proof of Lemma 6.1. Suppose that s is a USO outmap, and fix a face with free coordinates $J \subseteq [n]$. We want to show that

$$x \longmapsto (s_i(x))_{i \in J}$$

is a bijection between the vertices of this face and $\{0,1\}^J$.

First, fix $K \subseteq J$. Partition the face into the $2^{|J|-|K|}$ smaller faces whose free coordinates are precisely K. A vertex satisfies $s_i(x) = 0$ for every $i \in K$ if and only if it is the sink of its smaller face. Since each smaller face has exactly one sink, there are exactly $2^{|J|-|K|}$ such vertices.

Now fix $Z \subseteq J$. We count the vertices whose outmap is zero exactly on the coordinates in Z, among those in J. By inclusion–exclusion, this number is

$$\sum_{K \subseteq J \setminus Z} (-1)^{|K|} 2^{|J|-|Z|-|K|} = 2^{|J|-|Z|} \left(1 - \frac{1}{2}\right)^{|J|-|Z|} = 1 \ .$$

Thus every possible outmap value occurs exactly once. For any distinct vertices x, x', consider the smallest face containing both. Its free coordinates are exactly those where x and x' differ. Their outmap values on these coordinates must also differ, proving Equation (6.1).

Conversely, suppose Equation (6.1) holds. Apply it first to two vertices differing only at coordinate i. Their i-th outmap bits must differ, so s assigns a consistent direction to their shared edge. Next, fix any face with free coordinates J. The same condition says that its restricted outmap is injective. Since the face and $\{0,1\}^J$ have the same size, this map is bijective. Exactly one vertex maps to the all-zero string on J, and this vertex is the unique sink of the face. Taking the entire cube also shows that s is a permutation.

Finally, whenever $x_i \neq x'_i$, we have

$$x_i \oplus s_i(x) = x'_i \oplus s_i(x') \quad \Longleftrightarrow \quad s_i(x) \neq s_i(x') \quad .$$

Substituting $h(x) = x \oplus s(x)$ gives Equation (6.2). □

Why our generating circuit needs the USO property. Theorem 6.2 shows that a USO gives a valid pure PMG. For the particular circuit used in that theorem, the converse also holds. To see why, we choose local channels that fix some feedback inputs and therefore restrict them to a face. On external input $|0^n\rangle$, the trace of the resulting output counts the sinks of that face. Validity requires this trace to be exactly one.

Proposition A.1 (The exact condition needed by our USO generator)**.** *Let $s : \{0,1\}^n \to \{0,1\}^n$ be any Boolean function, and let C_s be the unitary circuit in Equation (6.10). Then (n, C_s) is a PMG if and only if s is a USO outmap.*

Proof. One direction is Theorem 6.2. For the other, suppose (n, C_s) is a PMG. Fix a face by setting the coordinates in $I \subseteq [n]$ to a string $c \in \{0,1\}^I$, and write $J := [n] \setminus I$ for its free coordinates. At each feedback pair $i \in I$, insert the channel that discards its input and prepares $|c_i\rangle$. At each feedback pair $j \in J$, insert the identity. Set the external input to $|0^n\rangle$.

The reset channel at coordinate i has Kraus operators

$$\{|c_i\rangle\langle b_i| : b_i \in \{0,1\}\} \quad .$$

For each $b \in \{0,1\}^I$, let K_b be the connected Kraus operator corresponding to these choices of b_i. Write $h(x) = x \oplus s(x)$. Expanding the operator partial trace gives

$$K_b|0^n\rangle = \sum_{\substack{x_I = c;\ s_j(x)=0 \text{ for every } j \in J; \\ h_i(x) = b_i \text{ for every } i \in I}} |x\rangle \quad .$$

Indeed, the reset channels force $x_I = c$, and their Kraus operators select the feedback outputs $b_i = h_i(x)$. At a free coordinate j, the identity connection requires $h_j(x) = x_j$, which is equivalent to $s_j(x) = 0$.

Every vertex satisfying $x_I = c$ and $s_j(x) = 0$ for all $j \in J$ appears for exactly one value of b, with amplitude one. Distinct vertices give orthogonal output states. Therefore,

$$\sum_{b \in \{0,1\}^I} \|K_b|0^n\rangle\|_2^2 = \#\{x \in \{0,1\}^n : x_I = c,\ s_j(x) = 0 \text{ for every } j \in J\} \quad .$$

Since the connected transformation is trace-preserving, this number must be one. On a one-dimensional face, this says that exactly one endpoint has outmap bit zero along its edge, so s defines a consistent orientation. On any face, it then says that there is exactly one sink. Thus s is a USO outmap. □

Approximating the single-qubit gates. Theorem 6.1 applies to any choice of the single-qubit unitaries. Thus, replacing them by unitary approximations still gives a pure PMG. What changes is the generated unitary. The next lemma bounds this change in terms of the approximation error of the individual gates.

Lemma A.1 (Precision of the basis transformation)**.** *Fix h as in Theorem 6.1. Let $T, \widetilde{T}$ be the generated matrices for two collections of single-qubit unitaries $U_{i,a}, \widetilde{U}_{i,a}$. If*

$$\left\| U_{i,a} - \widetilde{U}_{i,a} \right\|_{\mathrm{op}} \leq \eta \quad \textit{for every } i \in [n],\ a \in \{0,1\},$$

then

$$\left\|T-\widetilde{T}\right\|_{\mathrm{op}} \le \left\|T-\widetilde{T}\right\|_{\mathrm{F}} \le 2^{n/2} n\eta \ . \tag{A.1}$$

Here $\|M\|_{\mathrm{op}} := \sup_{\|v\|_2=1} \|Mv\|_2$ *is the operator norm, and* $\|M\|_{\mathrm{F}}^2 := \sum_{x,y} |M_{x,y}|^2$ *is the squared Frobenius norm. If both collections are implemented by unitary circuits, both generating circuits describe pure PMGs exactly.*

Proof. Fix a row x of T. By Equation (6.3), this row is

$$\bigotimes_{i=1}^{n} \langle x_i | U_{i,h_i(x)} \ .$$

Replace these factors one at a time by the corresponding factors from $\widetilde{T}$. Each replacement changes the row by Euclidean norm at most η: the changed factor differs by norm at most η, and every other factor has norm one. By the triangle inequality, the two rows differ by norm at most $n\eta$. Summing the squared differences over the 2^n rows gives

$$\left\|T-\widetilde{T}\right\|_{\mathrm{F}}^2 \le 2^n n^2 \eta^2 \ .$$

Taking square roots and using that the operator norm is at most the Frobenius norm proves Equation (A.1). Both generating circuits describe pure PMGs by Theorem 6.1.

The same bound controls the output trace distance, including when the input is entangled with a reference register. Tensoring with the identity on that register leaves the operator norm unchanged. For unit vectors $|u\rangle, |v\rangle$, the trace distance between their density matrices is

$$\sqrt{1-|\langle u|v\rangle|^2} \le \| |u\rangle - |v\rangle \|_2 \ .$$

Applying this inequality to the two output states gives the claim for pure joint inputs. For a mixed joint input, first purify it and then discard only the additional purifying register. Trace distance cannot increase under this last operation, so the bound also holds for mixed inputs, with the original reference register included in the output. □

The approximation compares the unitary matrices with their specified phases. These phases matter here: a gate is selected according to the feedback input, so changing its phase can change the relative phases between different branches of the computation.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il