Unitary Synthesis with a PSPACE Oracle

Sichen Wang
The Chinese University of Hong Kong, Shenzhen
sichenwang@cuhk.edu.cn

Abstract

For decision problems, polynomial space is the same as polynomial time with a PSPACE oracle. The analogue for unitaries is not obvious, because a classical oracle answers questions about strings, while a unitary must be applied to a quantum state that cannot be read. We show that every unitary that can be implemented in polynomial space can also be implemented in quantum polynomial time with a PSPACE oracle, up to exponentially small error. A classical oracle helps because the unitary enters our circuits only through classical gate parameters, which can be computed from its matrix entries in polynomial space. More generally, an oracle L suffices to implement all polynomial-space unitaries in quantum polynomial time if and only if $\mathsf{PSPACE} \subseteq \mathsf{BQP}^L$, so these unitaries are exactly as hard as PSPACE decision problems. In cryptography, this means that quantum commitments imply $\mathsf{BQP} \neq \mathsf{PSPACE}$, since a commitment yields an EFI pair, whose two states can be distinguished by a measurement implemented in polynomial space. For unitary synthesis problems whose input and output sizes can be computed in polynomial time, we also show that interaction with an untrusted prover gives a polynomial-time verifier exactly the power of polynomial space, even when the verifier must be correct on every input state, just as for decision problems.

1 Introduction

Unitary synthesis is the task of applying a prescribed unitary to an unknown quantum state. A computation can neither copy such a state nor measure it without disturbing it, and the state may be entangled with a system that the computation does not hold, so the unitary has to be applied coherently. This task arises in breaking quantum commitments, compressing quantum states and decoding noisy channels [BEM+26].

Some unitaries are hard for a simple reason. Deciding a language L amounts to implementing its oracle gate $O_L|x,b\rangle = |x, b \oplus L(x)\rangle$, so a unitary can be as hard as any decision problem. Is this the only reason? If decision problems were easy, would unitaries be easy too? Aaronson and Kuperberg asked this question in 2007, in a form now called the Unitary Synthesis Problem: can every unitary, computable or not, be implemented in polynomial time with a suitable classical oracle [AK07, Section 7, Problem (4)]?

They conjectured that the answer is no [Aar16, Section 3.2], and there was good reason. A classical oracle answers questions about strings. The state to be transformed is not a string, and the computation cannot read it, so it is not clear what the computation should ask. For preparing a state this problem does not arise, because there is no input: an oracle can supply the rotations that prepare the state qubit by qubit [GR02], [Aar16, Proposition 3.3.5], and one query suffices up to exponentially small error [Ros24]. In the same way an oracle can map $|v\rangle|0\rangle$ to $|v\rangle U|v\rangle$ for a unitary U and each basis state $|v\rangle$, but this leaves v in a register, and removing it coherently is the difficulty. Indeed, no circuit with one query of polynomial length implements every unitary, even when the oracle may depend on the unitary [LMW24], and the best algorithm took time $2^{n/2}\,\mathrm{poly}(n)$ [Ros26].

A recursive construction then gave a positive answer at constant error [Ope26], with an oracle that is shown to exist but not shown to be computable. This left two questions: how powerful the oracle has to be and, since no generic error reduction is known [BEM+26, Open Problem 1], whether the error can be made exponentially small. For unitaries that can be implemented in polynomial space, the natural candidate is a PSPACE oracle, because for decision problems $\mathsf{PSPACE} = \mathsf{P}^{\mathsf{PSPACE}}$ is immediate. Bostanci, Efron, Metger, Poremba, Qian and Yuen [BEM+26, Section 1.1, footnote 3] and Kretschmer and Tang [KT26, Section 1.1] ask for the analogue: can every unitary that is implementable in polynomial space also be implemented in quantum polynomial time with a classical PSPACE oracle?

1.1 Results

The answer is yes: every polynomial-space unitary can be implemented in quantum polynomial time with an oracle for TQBF, up to exponentially small error. A language L can replace TQBF if and only if $\mathsf{PSPACE} \subseteq \mathsf{BQP}^L$, so polynomial-space unitaries are exactly as hard as PSPACE decision problems. Both statements hold relative to every classical oracle A, with a PSPACE^A-complete language in place of TQBF and PSPACE^A in place of PSPACE.

To state this precisely, we use the classes of Kretschmer and Tang [KT26], defined formally in Section 2.2. A unitary language is a sequence $U = (U_n)_{n\geq 1}$ in which U_n is a unitary on n qubits. It lies in $\mathsf{unitaryP}^A$ if a polynomial-time algorithm maps $(1^n, 1^m)$ to a circuit with oracle gates O_A, which query A in superposition, and this circuit is within 2^{-m} in operator norm of a unitary that maps $|\psi\rangle|0\rangle$ to $(U_n|\psi\rangle)|0\rangle$ for every $|\psi\rangle$, where $|0\rangle$ is the state of the ancillas. The class $\mathsf{unitaryPSPACE}^A$ allows a polynomial-space algorithm instead, with circuits of polynomial width and possibly exponentially many gates. The projective classes $\mathsf{projUnitaryP}^A$ and $\mathsf{projUnitaryPSPACE}^A$ allow the ancillas to end in a fixed state that does not depend on $|\psi\rangle$, instead of returning to zero as in the clean classes $\mathsf{unitaryP}^A$ and $\mathsf{unitaryPSPACE}^A$.

Theorem 1.1 (Main theorem)**.** *For every classical oracle A,*

$$\mathsf{unitaryPSPACE}^A = \bigcup_{L \in \mathsf{PSPACE}^A} \mathsf{unitaryP}^L,$$
$$\mathsf{projUnitaryPSPACE}^A = \bigcup_{L \in \mathsf{PSPACE}^A} \mathsf{projUnitaryP}^L.$$

Moreover, if a language K is PSPACE^A-complete under polynomial-time many-one reductions computed without access to A, then $\mathsf{unitaryPSPACE}^A = \mathsf{unitaryP}^K$ and $\mathsf{projUnitaryPSPACE}^A = \mathsf{projUnitaryP}^K$.

Thus one language, TQBF when $A = \emptyset$, suffices for all sizes and all precisions, with resources polynomial in $n + m$.

It was known that the entries of a unitary in $\mathsf{unitaryPSPACE}^A$ can be computed in polynomial space relative to A [KT26, Theorem 6.17]. Our proof uses nothing else about the unitary. A classical oracle helps because a unitary can be implemented by a circuit that depends only on the sizes, with everything specific to the unitary in classical gate parameters. Our main technical result (Theorem 3.1) is a polynomial-time algorithm that writes such a circuit from $(1^n, 1^m)$ alone. The circuit has polynomially many gates, and in those specific to the unitary a control string selects which one-qubit gate is applied, among possibly exponentially many. These one-qubit gates are computed from the entries of U_n in polynomial space, so one language $B \in \mathsf{PSPACE}^A$, which depends on U, can encode short words over a fixed gate set that approximate them. The circuit reads the selected words by querying B in superposition over the control strings, and it erases the answers after using them.

The construction works relative to every classical oracle, so it applies also to a unitary that cannot be computed at all. The language A_U that encodes the bits of its entries is the entry oracle of U (Section 2.3). Let $\mathsf{unitaryALL}$ be the class of all unitary languages and let $\mathsf{unitaryP}^{\mathsf{ALL}} = \bigcup_L \mathsf{unitaryP}^L$.

Corollary 1.2. *For every unitary language U there is a language $B_U \in \mathsf{PSPACE}^{A_U}$ with $U \in \mathsf{unitaryP}^{B_U}$, and one polynomial-time algorithm, which does not depend on U, produces the circuits. In particular, $\mathsf{unitaryALL} = \mathsf{unitaryP}^{\mathsf{ALL}}$.*

This answers the Unitary Synthesis Problem at exponential precision, with one algorithm for all unitaries and an oracle for each of them, as in the original question. Under the formalization of Lombardi, Ma and Wright [LMW24, Definition 3.16] it is a positive answer, so their Strong Non-Synthesis Conjecture [LMW24, Conjecture 1.5] is false (Proposition 5.4).

Is a PSPACE oracle necessary? The observation we started from says that it is: the oracle gate of a language in PSPACE^A is itself a polynomial-space unitary, and a circuit that implements it decides the language, so a language L can suffice only if $\mathsf{PSPACE}^A \subseteq \mathsf{BQP}^L$. Conversely, if $\mathsf{PSPACE}^A \subseteq \mathsf{BQP}^L$, each oracle gate in the circuits of Theorem 1.1 can be replaced by a coherent run of a decider in BQP^L (Section 5.1).

Corollary 1.3. *For all classical oracles A and L,*

$$\mathsf{unitaryPSPACE}^A \subseteq \mathsf{unitaryP}^L \iff \mathsf{PSPACE}^A \subseteq \mathsf{BQP}^L$$
$$\iff \mathsf{projUnitaryPSPACE}^A \subseteq \mathsf{projUnitaryP}^L.$$

In particular, $\mathsf{unitaryP}^A = \mathsf{unitaryPSPACE}^A$ if and only if $\mathsf{BQP}^A = \mathsf{PSPACE}^A$, and the same holds for the projective classes.

So, relative to classical oracles, the hardness of decision problems is the only reason why polynomial-space unitaries are hard. In particular, $\mathsf{P} = \mathsf{PSPACE}$ implies $\mathsf{unitaryP} = \mathsf{unitaryPSPACE}$, and no classical oracle can make this implication fail, which answers a question of [BEM+26, Section 1.4] and [KT26, Section 1.1].

Quantum cryptography. Classically, $\mathsf{P} = \mathsf{NP}$ rules out one-way functions, so cryptography requires hard decision problems. Quantum cryptography might require less: pseudorandom states, which are efficiently generated and cannot be distinguished efficiently from Haar-random states, exist relative to a classical oracle with $\mathsf{P} = \mathsf{NP}$, against adversaries with one copy [KQST23] and even with polynomially many [KQT25]. A weaker assumption still is the existence of EFI pairs [BCQ23], pairs of efficiently generated mixed states that are statistically far apart but computationally indistinguishable, which is equivalent to the existence of quantum commitments [BCQ23, Theorem 13], [Yan22]. Since the two states are far apart, some measurement distinguishes them, and the security of the pair says that this measurement is hard to implement. Brakerski, Canetti and Qian ask whether EFI pairs separate BQP from PSPACE, and they suggest synthesizing the measurement with a PSPACE oracle [BCQ23, Section 1.2]. Our results carry this out.

Corollary 1.4. *If EFI pairs exist relative to a classical oracle A, then $\mathsf{BQP}^A \neq \mathsf{PSPACE}^A$. In particular, EFI pairs imply $\mathsf{BQP} \neq \mathsf{PSPACE}$.*

A proof that quantum commitments exist would therefore separate BQP from PSPACE. A smooth version of the distinguishing measurement can be implemented by a uniform polynomial-time algorithm with a PSPACE^A oracle, and it distinguishes the two states from one copy with inverse-polynomial advantage (Lemma 5.3). Corollary 1.3 removes this oracle if $\mathsf{BQP}^A = \mathsf{PSPACE}^A$. The same algorithm breaks single-copy pseudorandom states that are longer than their key, which answers, for these states, the question of Lombardi, Ma and Wright whether a PSPACE oracle breaks every single-copy pseudorandom state [LMW24, Section 1.1]. So the existence of these states implies $\mathsf{BQP} \neq \mathsf{PSPACE}$, and hence $\mathsf{P} \neq \mathsf{PSPACE}$, as asked in [KQST23, Section 1.4].

Interactive synthesis. In an interactive implementation, a polynomial-time verifier holds the quantum input and interacts with an untrusted prover, and if it accepts, it should hold the transformed input. For decision problems interaction gives exactly polynomial space [JJUW11], as it does for preparing states [RY22; MY23] and for unitaries in the average case [BEM+26, Theorem 7.10]. The worst case is harder, because the verifier holds the only copy of its input, and a prover that receives this copy could alter it without being detected. It remained open [BEM+26, Open Problem 10], even given a PSPACE oracle for synthesis [Ros23, Open Problem 29]. We settle it for size-computable problems, those whose register sizes can be computed in polynomial time. Our verifier never sends its input. It runs a polynomial-time oracle circuit for the problem on its own qubits, and it uses the prover only to obtain quantum states that do not depend on the input, with which it implements the oracle gates.

We use the model of [BEM+26] (Sections 2.5 and 2.6). The target is a partial isometry U_x named by an instance x, from $n_{\text{in}}(x)$ to $n_{\text{out}}(x) \geq n_{\text{in}}(x)$ qubits, that is, a map that acts as an isometry on a subspace, its support, and as zero on the orthogonal complement. The family $(U_x)_x$ is called a unitary synthesis problem [BEM+26, Definition 3.1], not to be confused with the Unitary Synthesis Problem above. A circuit may measure and discard qubits, and at precision r its channel must be within $1/r$ in diamond norm of a channel that agrees with U_x on the support. The diamond norm measures the worst case over inputs, including inputs entangled with another register. Such circuits define $\mathsf{unitaryBQP}^{\diamond}$ when they are written in time polynomial in $|x| + r$, and $\mathsf{unitaryPSPACE}^{\diamond}$

when they are written and run in space polynomial in $|x| + r$. A problem lies in $\mathsf{unitaryQIP}^{\diamond}_{1,1/2}$ if a polynomial-time algorithm writes from $(x, 1^r)$ a verifier with two properties. Some prover is accepted with probability one on every input in the support. For every input, possibly entangled with a register that neither party holds, and every prover accepted with probability at least $1/2$, the output on acceptance, together with that register, is within trace distance $1/r$ of the output of some channel that agrees with U_x on the support, applied to that input.

Theorem 1.5. *Let U be a size-computable unitary synthesis problem. Then*

$$U \in \mathsf{unitaryPSPACE}^{\diamond} \iff U \in \mathsf{unitaryQIP}^{\diamond}_{1,1/2}.$$

This is the worst-case analogue of $\mathsf{QIP} = \mathsf{PSPACE}$ for size-computable problems. The condition on the register sizes cannot be dropped from the inclusion of $\mathsf{unitaryQIP}^{\diamond}_{1,1/2}$ in $\mathsf{unitaryPSPACE}^{\diamond}$, and it cannot be dropped from the other inclusion unless $\mathsf{P} = \mathsf{PSPACE}$ (Remark 7.11). In this model too, the collapse of the classes is determined by decision classes (Theorem 6.5). The size-computable problems in $\mathsf{unitaryPSPACE}^{\diamond}$ all lie in $\mathsf{unitaryBQP}^{\diamond}$ exactly when $\mathsf{BQP} = \mathsf{PSPACE}$, and the classes themselves coincide exactly when $\mathsf{P} = \mathsf{PSPACE}$, because $\mathsf{unitaryBQP}^{\diamond}$ requires the register sizes to be computed in deterministic polynomial time.

1.2 Overview

Every result above uses Theorem 3.1, which implements a unitary from its entries. Preparing a column of U_n leaves its index in a register, which is the difficulty described above. The index can be removed when the matrix has few nonzero entries per row, because a row then gives a short list of candidates for the index. A general U_n has no such structure, so we reduce to matrices that do. We implement U_n by applying a map, then one call to a family of unitaries of smaller dimension, selected in superposition by a control register, and then the adjoint of a second map. In a suitable basis these two maps have few nonzero entries per row, because the control register records which of the removed coordinates the computation passes through. This is one stage of a recursion. Each stage removes an inverse-polynomial fraction of the coordinates, so polynomially many stages reach dimension one. In the resulting circuit the registers and the order of the gates depend only on n and m, and the unitary enters only through the one-qubit gates applied for each value of the control registers, which we call the parameters of the circuit (Section 3). We then compute the parameters in polynomial space, where the required precision grows only additively over the stages, and collect them in one language (Section 4).

Consider a stage that implements a unitary $U^{(j)}$ on d coordinates, with $U^{(0)} = U_n$. We split the coordinates into k leading and $s = d - k$ trailing ones, with $s \geq d/\operatorname{poly}(n)$. To remove the trailing ones, we take a unitary G obtained from $U^{(j)}$ in a way described below, with blocks $G_{11}, G_{12}, G_{21}, G_{22}$ for this split, and we return the trailing part of its output to the trailing part of its input through a diagonal unitary Z, forming a loop. On a leading input u, the trailing output x then satisfies $x = G_{21}u + G_{22}Zx$, and the leading output is $G_{11}u + G_{12}Zx$. If $\|G_{22}\| \leq 3/4$, this gives $x = Xu$ and the leading output $U^{(j+1)}u$, where

$$X = (I - G_{22}Z)^{-1}G_{21}, \qquad U^{(j+1)} = G_{11} + G_{12}ZX.$$

Since G and Z preserve norms, $\|u\|^2 + \|Xu\|^2 = \|U^{(j+1)}u\|^2 + \|Xu\|^2$, so $U^{(j+1)}$ is a unitary on the leading coordinates. This is the unitary of the next stage. The inverse $(I - G_{22}Z)^{-1}$ is a geometric series in $G_{22}Z$, whose terms are sums over paths, the sequences of trailing coordinates that the loop passes through.

The trailing block of $U^{(j)}$ itself can have norm one, so we take $G = P_{\mathrm{L}} U^{(j)} P_{\mathrm{R}}$, where $P_{\mathrm{L}}, P_{\mathrm{R}}$ are unitaries with short descriptions. Random sign diagonals and Hadamard transforms achieve $\|G_{22}\| \leq 3/4$. The argument uses only moments of degree about n, so signs from seeds of polynomial length suffice, and a search over the seeds in polynomial space replaces the random choice (Section 3.2).

To implement G we relate it to $U^{(j+1)}$. In the loop, G maps the input (u, ZXu) to the output $(U^{(j+1)}u, Xu)$. With $E = \binom{I}{ZX}$ and $E' = \binom{I}{XU^{(j+1)\dagger}}$ this reads $GE = E'U^{(j+1)}$, and hence $U^{(j+1)}E^\dagger = E'^\dagger G$. This would express G through $U^{(j+1)}$ if $E^\dagger$ and $E'^\dagger$ were isometries, but they map d coordinates to k and so cannot be. To supply the missing dimensions, we let $Z = Z(z)$ depend on a control string z and take a superposition over z. Let $\mathcal{E}_N$ be the map $M^{-1/2} \sum_z |z\rangle \otimes E(z)^\dagger$ over the M control strings, with the geometric series truncated after N terms, so that only paths of length at most N occur, and let $\mathcal{E}'_N$ be the same map for $G^\dagger$, with $-z$ in place of z.

We take the entry of $Z(z)$ at a trailing coordinate λ to be $\omega^{v_\lambda \cdot z}$, where $v_\lambda = (1, \lambda, \ldots, \lambda^N)$ and ω is a root of unity whose order exceeds every coordinate of any sum of at most N of the vectors v_λ. A path then contributes to one Fourier frequency of the control register only, the sum of the vectors v_λ along the path, up to a sign. Thus the frequency records the visited coordinates: the first coordinate of this sum is the length, and the others are power sums, which determine the multiset of visited coordinates by Newton's identities.

This record gives the two properties we need. First, the contributions of different path lengths have different frequencies, so they are orthogonal, and the sum of their squared norms telescopes because G is unitary. Therefore $\mathcal{E}_N$ and $\mathcal{E}'_N$ are near-isometries, which preserve norms up to an error exponentially small in N, and the identity above gives

$$\Big(\bigoplus_z U^{(j+1)}(z)\Big)\mathcal{E}_N \approx \mathcal{E}'_N G.$$

Thus G is implemented by applying $\mathcal{E}_N$, then the family controlled by z, then the adjoint of $\mathcal{E}'_N$ (Section 3.3). Second, a path that contributes to a trailing column passes through the coordinate of that column, so a nonzero frequency leaves at most N candidates for the column index, and a row at frequency zero has its only entry in the column of its own leading coordinate. In the Fourier basis each row of these maps therefore has at most N candidate columns, computable from the row index.

The index can now be removed. After we prepare a column, reversing the preparation of the uniform superposition over the candidates takes the index to zero with inverse-polynomial amplitude. A rotation of a flag qubit makes this amplitude the same for all rows, and amplitude amplification raises it to nearly one, because all singular values of the maps are close to one (Section 3.4). With the circuits $C_{\mathcal{E},j}, C_{\mathcal{E}',j}$ obtained in this way, a stage is

$$C_j = P_{\mathrm{L}}^\dagger C_{\mathcal{E}',j}^\dagger C_{j+1} C_{\mathcal{E},j} P_{\mathrm{R}}^\dagger,$$

where C_{j+1} implements $\bigoplus_z U^{(j+1)}(z)$, with the control registers of all earlier stages as further controls. Each stage contains one call to C_{j+1}, so gate counts add over the stages, and the surrounding maps are contractions, so errors add as well. A truncation length N linear in $m + \log n$ then gives total error below 2^{-m} over all stages (Section 3.5).

The unitary enters only through the parameters, which are functions of the unitaries $U^{(j)}$, so we must compute entries of these unitaries in polynomial space. An entry of $U^{(j+1)}$ is computed from entries of $U^{(j)}$, and so on down to U_n. A term-by-term bound on the series, which has exponentially many paths, multiplies the required precision by a factor greater than one at each stage, and over polynomially many stages this would demand exponentially many bits. Instead we first round the blocks of $G\,\mathrm{diag}(I, Z)$ to a grid, bound the effect on the inverse through $\|(I - G_{22}Z)^{-1}\| \leq 4$, and then sum the truncated series exactly on the rounded numbers. An entry to t bits of precision

then needs entries of the previous stage only to $t + O(n)$ bits, so all precisions are polynomial and polynomial space suffices (Section 4.1). No error reduction is needed for exponential precision, since error 2^{-m} requires only a truncation length linear in m and parameters computed to polynomially many bits.

Each parameter is a one-qubit gate, which we approximate by short words over $\{I, H, T, T^\dagger\}$ using the Solovay–Kitaev theorem, with one further qubit for its global phase. These words can be found in polynomial space, and the language B contains their bits for all sizes, gates and control strings. The algorithm that writes the circuit needs only the format of the queries, so it never evaluates B and is the same for all A and U (Section 4.2). This gives Theorem 3.1.

Theorem 1.1 follows. The entries of a unitary in $\mathsf{unitaryPSPACE}^A$ can be computed in polynomial space relative to A, which gives one inclusion, and so can the entries of a polynomial-time circuit with an oracle in PSPACE^A, which gives the other. A complete language replaces B through the reduction from B to it. The projective classes reduce to the clean ones through $U_n \otimes U_n^\dagger$. A circuit for U_n that leaves a fixed state on its ancillas, followed by its adjoint on a second data register with the same ancillas, implements this product and returns the ancillas to zero, and setting the second data register to zero turns a circuit for the product into a circuit for U_n (Section 4.3).

The other results use Theorem 3.1 as a tool: to implement something, we construct a unitary whose entries can be computed in polynomial space. For Corollary 1.4, the optimal measurement between two states depends discontinuously on them, so we take a Hadamard test that implements a smooth version of it (Section 5.2). For Theorem 1.5 we first need such circuits in the channel model, where a polynomial-space circuit may discard exponentially many qubits, so that keeping them does not give a unitary on polynomially many qubits. The channel itself acts on polynomially many qubits, and the square root of its Choi matrix gives a Stinespring isometry, which a reflection embeds in a unitary whose entries can be computed in polynomial space (Section 6).

This gives the oracle circuit that the verifier of Theorem 1.5 runs. Up to Hadamard gates, an oracle gate is a phase unitary, a diagonal unitary with entries ± 1 given by a Boolean function f. The uniform superposition with the same signs, its phase state, can be prepared in polynomial space, so interactive state synthesis lets the verifier obtain it from the prover [Ros24; BEM+26]. State injection consumes this state and applies the phase unitary, up to the phase unitary of the Boolean derivative $u \mapsto f(u) \oplus f(u \oplus v)$, where v is the measurement outcome. Injecting the state of each correction in turn takes further derivatives, and each derivative lowers the degree of the function as a polynomial over $\mathbb{F}_2$, so after polynomially many injections the correction vanishes. One prover with a persistent memory supplies all these states, but because the target states are pure, the subnormalized state on acceptance is close to a product with the prover's memory, so the errors add over the calls (Sections 7.1 to 7.3).

For the converse, a polynomial-space circuit runs the verifier against a prover that is near-optimal on the all-zero input and computable in polynomial space [MY23]. If a prover were accepted with very different probabilities on two inputs, acceptance on a superposition of them, entangled with one further qubit, would change the distribution of this qubit, which no channel on the input can do. So soundness implies that high acceptance on one input gives high acceptance on all inputs. Applied to an honest prover, this gives high acceptance on the all-zero input, so the near-optimal prover is accepted with high probability there, and hence on all inputs. A minimax argument then gives one channel for all inputs (Section 7.4).

Every theorem, corollary and proposition of this paper has also been formalized in Lean 4, with the four facts quoted from the literature (Facts 2.5, 2.8, 7.4 and 7.8) as axioms. The code, with a table matching each statement to its Lean counterpart, is available at `https://github.com/sichen-wang/unitary-synthesis-pspace-oracle-lean`.

1.3 Related Work

The Unitary Synthesis Problem [Aar16, Question 3.2.6], [Aar21, Problem 6], [KT26, Problem 7.1] allows an oracle because, by counting, almost all unitaries need exponentially many gates [Kni95]. It has been approached from two sides. Aaronson and Kuperberg gave a first lower bound: one query does not suffice for circuits that implement some unitary on their data qubits for every oracle [AK07, Theorem 6.7]. Lombardi, Ma and Wright removed this restriction and ruled out one query and polynomially many parallel queries [LMW24], and later work treats few queries [Hua25; DLM26], the input length of the oracle [BY26] and random oracles [HRW26]. On the side of algorithms, Rosenthal gave time $2^{n/2}\,\mathrm{poly}(n)$ [Ros26], and the manuscript [Ope26] gives polynomial resources at constant diamond error. Our circuits make polynomially many sequential queries of polynomial length, at exponentially small error.

A second question is how powerful the oracle has to be. Aaronson asks whether $\mathsf{P} = \mathsf{PSPACE}$ would give small circuits for explicit unitaries [Aar16, Section 3.3.1], [Aar21, Section 6]. Rosenthal observes that polynomial-time synthesis with an oracle in PSPACE would give such circuits if $\mathsf{BQP} = \mathsf{PSPACE}$ [Ros23, Section 1.2], and Kretschmer and Tang show that efficient synthesis of a polynomial-space unitary with a classical oracle also works with an oracle for the bits of NEXP witnesses [KT26, Lemma 7.4 and Corollary 7.5]. Theorem 1.1 and Corollary 1.3 settle this question at exponential precision. Unitary oracles behave differently: an unknown unitary cannot be controlled [AFCB14], and garbage, the fixed state left on the ancillas in the projective classes, cannot always be erased, even in polynomial space [KT26, Theorem 6.23]. Kretschmer and Tang ask whether polynomial-time implementations can always erase it, up to a global phase [KT26, Question 3.9], and relative to a classical oracle A with $\mathsf{BQP}^A = \mathsf{PSPACE}^A$ they can (Section 5.1).

These questions belong to the study of complexity classes of states and unitaries, which Rosenthal and Yuen began by asking which tasks interaction can achieve [RY22]. They showed that interaction can prepare every state that can be prepared in polynomial space, and that it can implement every polynomial-space unitary acting nontrivially on a polynomial-dimensional subspace [RY22, Theorem 1.3], by sample-based Hamiltonian simulation [LMR14; KLL$^+$17]. Metger and Yuen proved the converse for states, and they showed that near-optimal provers can be implemented in polynomial space [MY23, Theorem 7.1], which our converse uses. Bostanci, Efron, Metger, Poremba, Qian and Yuen introduced the channel model and proved the equality of the average-case classes [BEM$^+$26, Theorem 7.10]. Theorem 1.5 gives the worst case for size-computable problems, with completeness one and threshold $1/2$.

Part of the interest in these classes comes from cryptography. Multi-copy pseudorandom states [JLS18] of superlogarithmic length are broken by one query to a PSPACE oracle [Kre21, Theorem 27]. For EFI pairs, prior work derived separations of unitary classes [BEM$^+$26, Theorem 8.15], [KT26, Theorem 7.14] and of polynomial time from polynomial space for decision problems with quantum inputs [CCH$^+$26, Theorem 1.19]. Corollary 1.4 extends this to decision classes with classical inputs, relative to every classical oracle, and shows that a classical oracle breaks EFI pairs, which was not known [CMY26, Section 1.3]. Relative to Haar-random state and unitary oracles, by contrast, commitments are secure against adversaries with unbounded computation and polynomially many copies or queries [CCS25; AGL25; BCN25; CM26].

On the technical side, our parameters are computed by the kind of recursion that underlies space-bounded quantum computation. It goes back to Savitch [Sav70], and through $\mathsf{BQPSPACE} = \mathsf{PSPACE}$ [Wat99; Wat03] and its relativized form [KT26, Proposition 4.8] it determines acceptance probabilities and matrix entries. Theorem 3.1 turns such entries into an implementation. For channels we must also remove measurements. Intermediate measurements can be eliminated while preserving decision probabilities [FR21], measurements whose outcomes are discarded can be eliminated in circuits

without resets [GR22], and black-box conversions must increase space or time significantly [Zha24]. We need a unitary that reproduces a channel on every input, including entangled inputs, and Lemmas 6.3 and 6.4 construct it from the Choi matrix.

2 Preliminaries

To prove Theorem 1.1, we construct polynomial-time circuits with oracle gates from polynomial-space algorithms for matrix entries.

2.1 Circuits and Oracles

Logarithms are base two, 1^n is n in unary, basis indices start at 0, and i is the imaginary unit. We write $\|\cdot\|$, $\|\cdot\|_1$ and $\|\cdot\|_F$ for the operator, trace and Frobenius norms, $D(\rho,\sigma)=\frac{1}{2}\|\rho-\sigma\|_1$ for the trace distance, and $\preceq$ for the Loewner order. A pure state $|\psi\rangle$ is also written $\psi=|\psi\rangle\langle\psi|$. The diamond norm of a difference Λ of channels is $\|\Lambda\|_\diamond=\sup_\psi\|(\Lambda\otimes\mathrm{id})(\psi)\|_1$, over pure states ψ on the input and a further register, so diamond distance ε means trace distance at most $\varepsilon/2$ on every such input.

We use the circuit model of Kretschmer and Tang [KT26]. A unitary gate acts on named wires. It is given by these wires and by encodings of the real and imaginary parts of its matrix entries, from which a fixed polynomial-time algorithm computes each number to any precision [KT26, Definitions 4.1–4.3]. The matrix it describes is exact, including its global phase. A classical oracle is a language $A\subseteq\{0,1\}^*$, and its oracle gate [KT26, Definition 4.4] on ℓ query qubits and one answer qubit is

$$O_A|x,b\rangle=|x,b\oplus A(x)\rangle \qquad (x\in\{0,1\}^\ell,\ b\in\{0,1\}).$$

A unitary circuit is a list of unitary gates and oracle gates. Its width is its number of wires, and C^A is the circuit C with its oracle gates set to O_A. The circuits we construct are over the gate set

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

that is, they use only these gates and oracle gates. Toffoli and NOT gates are exact over $\mathcal{G}$ with ancillas (Lemma 4.3), and we use them freely.

An ancilla is a qubit initialized to $|0\rangle$. We write $\iota_a|\psi\rangle=|\psi\rangle|0^a\rangle$ and call the range of ι_a the zero-ancilla subspace. A unitary circuit C on $n+a$ qubits implements a unitary U on n qubits on the zero-ancilla subspace with error ε if $\|C\iota_a-\iota_aU\|\le\varepsilon$ [KT26, Section 5.1].

Circuits that decide languages or implement channels may also measure and discard qubits.

Definition 2.1. A *general quantum circuit* [BEM+26, Section 2.3] consists of gates from $\mathcal{G}$, oracle gates, initialization of new qubits to $|0\rangle$, computational-basis measurement and discarding of qubits. A measurement is the dephasing map on its wire, and the outcome stays on that wire. The *space* of the circuit is the largest number of qubits present at any point, counting input qubits and initialized qubits that have not yet been discarded. Its description lists its input wires, its output wires and its gates in order, and its *size* is the length of the description. A general quantum circuit $\mathcal{C}$ implements a channel, which we denote by the same symbol. A general quantum circuit *with encoded gates* [KT26, Definitions 4.2 and 4.3] may in addition apply any channel to named wires, given by encodings of the entries of its matrix representation. This form is needed only for the EFI pairs of Section 5.2. The *matrix representation* of an operation Λ is the matrix with entries $\langle u|\Lambda(|u'\rangle\langle v'|)|v\rangle$ for basis indices u,v of the output and u',v' of the input.

A family of circuits indexed by strings is *time-uniform* (*space-uniform*) if a deterministic polynomial-time (polynomial-space) machine with no oracle and a write-only output tape, its *generator*, writes the description of each member on input its index. In a space-uniform family the description of each gate has length polynomial in the length of the index [BEM+26, Definition 2.7], [KT26, Definition 4.5].

Numbers in an index are written in unary: the index is $(1^n, 1^m)$ for a circuit that implements a unitary on n qubits at error 2^{-m}, and $(x, 1^r)$ in the channel model of Section 2.5. A time-uniform family has size polynomial in the length of the index, whereas a space-uniform family of polynomial width may have exponentially many gates, because the output tape does not count toward the space of its generator.

2.2 Unitary Languages and Their Complexity Classes

The objects we implement are sequences of unitaries, one for each number of qubits. A circuit with ancillas can implement a unitary in two ways: it returns the ancillas to zero, so that it can be used inside a larger coherent computation, or it leaves them in a fixed state.

Definition 2.2. A *unitary language* is a sequence $U = (U_n)_{n\geq 1}$ with $U_n \in \mathrm{U}(2^n)$ [KT26, Definition 4.9]. A unitary $\widetilde{U}$ on $n+a$ qubits is an *a-qubit extension* of U_n if $\widetilde{U}\iota_a = \iota_a U_n$ [Zha25], [KT26, Definition 5.1]. It is an *a-qubit dilation* of U_n with *garbage state* $|g\rangle$ if $\widetilde{U}(|\psi\rangle|0^a\rangle) = (U_n|\psi\rangle)|g\rangle$ for every $|\psi\rangle$ [KT26, Definition 5.11].

By linearity both identities also hold in the presence of further registers.

Definition 2.3. Let A be a classical oracle. A unitary language U belongs to $\mathsf{unitaryP}^A$ if there is a time-uniform family $(C_{n,m})_{n,m\geq 1}$ of unitary circuits of width polynomial in $n+m$ such that, for all n, m, some extension $\widetilde{U}_{n,m}$ of U_n satisfies

$$\|C^A_{n,m} - \widetilde{U}_{n,m}\| \leq 2^{-m}.$$

The class $\mathsf{unitaryPSPACE}^A$ is defined in the same way with space-uniform families. The classes $\mathsf{projUnitaryP}^A$ and $\mathsf{projUnitaryPSPACE}^A$ are defined with dilations in place of extensions. As in [KT26], we call the first two the *clean classes* and the last two the *projective classes*. The class of all unitary languages is $\mathsf{unitaryALL}$. We write $\mathsf{unitaryP}^{\mathsf{PSPACE}^A} = \bigcup_{L\in\mathsf{PSPACE}^A} \mathsf{unitaryP}^L$ and $\mathsf{unitaryP}^{\mathsf{ALL}} = \bigcup_L \mathsf{unitaryP}^L$, and likewise for the projective classes.

The norm compares matrices, so the global phase of U_n counts in the clean classes. In the projective classes it does not, because a dilation of U_n with garbage state $|g\rangle$ is also a dilation of ϕU_n with garbage state $\bar{\phi}|g\rangle$ for $|\phi| = 1$. An extension is a dilation with $|g\rangle = |0^a\rangle$, so the clean classes lie in the projective ones.

These are the classes of [KT26, Definitions 5.2, 5.6, 5.12 and 5.17] for a classical oracle. Kretschmer and Tang write the precision as a number ε with $O(\log(1/\varepsilon))$ bits, which amounts to our unary m. Their circuits may also use the adjoint, transpose and conjugate of O_A, which are all O_A itself, and in the clean classes they may use controlled queries. A controlled query with control c and answer b is obtained exactly by doing the following twice: query into a further qubit b', then apply a Toffoli gate from c and b' onto b. For every value of b' this adds $c(b' \oplus A(x)) \oplus cb' = c\,A(x)$ to b and restores b', so the replacement does not change the error.

For decision problems, BQP^A is the class of languages L for which there is a time-uniform family $(\mathcal{C}_n)_{n\geq 1}$, indexed by 1^n, of general quantum circuits with oracle gates O_A, each with n input wires

and one output wire, such that measuring the output of $\mathcal{C}_n$ on $|x\rangle$, for $x \in \{0,1\}^n$, gives $L(x)$ with probability at least $2/3$.

The class PSPACE^A consists of the languages decided by deterministic polynomial-space machines with oracle A whose query tape counts toward the space bound, so that queries have polynomial length. This is the convention of [KT26, Definitions 4.4–4.6 and footnote 5]. A language K is PSPACE^A-complete if it lies in PSPACE^A and every language in PSPACE^A reduces to it by a polynomial-time many-one reduction computed without access to A. One such language consists of the triples $(M, x, 1^s)$ such that the oracle machine M with oracle A accepts x within space s, and for $A = \emptyset$ another is TQBF [SM73].

2.3 Entries Computable in Polynomial Space

Our proofs access a unitary only through its matrix entries, and the entrywise characterization of Kretschmer and Tang (Fact 2.5) shows that this is no restriction for polynomial-space unitaries.

A dyadic number is a rational with a power of two as denominator, written in binary, and a complex dyadic number has dyadic real and imaginary parts. An algorithm computes a complex number to precision t if it outputs a complex dyadic number within 2^{-t} of it. The 2^{-t} grid consists of the complex numbers whose two components are multiples of 2^{-t}. To round a computable number to this grid, we compute it to precision $t+4$ and round each component of the result to the nearest multiple of 2^{-t}, with ties to the smaller rational. The result is within 2^{-t} of the number, and it is the same every time we compute it.

Definition 2.4. Let A be a classical oracle. A unitary language U has *entries computable in polynomial space relative to* A if a deterministic polynomial-space algorithm with oracle A, its *entry algorithm*, computes $\langle u|U_n|v\rangle$ to precision t on input $(1^n, u, v, 1^t)$, for $u, v \in \{0,1\}^n$, with $O(t+1)$ bits per component.

Fact 2.5 ([KT26, Theorem 6.17, items 1 and 6])**.** *For every classical oracle A, a unitary language lies in* $\mathsf{unitaryPSPACE}^A$ *if and only if it has entries computable in polynomial space relative to A.*

Kretschmer and Tang state this for the real and imaginary parts of $\langle v|U_n|u\rangle$ on input (u, v) [KT26, Definition 6.10]. We obtain the form above by swapping the indices, computing each part to precision $t+3$ and rounding it to the 2^{-t-2} grid.

The entries of an arbitrary unitary language need not be computable, but an oracle can encode them. The entry oracle A_U of a unitary language U is the set of strings $\langle 1^n, u, v, 1^t, \mathrm{part}, j\rangle$ with $u, v \in \{0,1\}^n$, $t \geq 0$, $\mathrm{part} \in \{\mathrm{re}, \mathrm{im}\}$ and $j \geq 0$ such that bit j, least significant first, of the integer $\lfloor(\alpha+1)2^{t+2}\rfloor$ is one, where α is the part of $\langle u|U_n|v\rangle$ selected by part. Malformed strings are not in A_U. Since $|\alpha| \leq 1$, the integer lies in $[0, 2^{t+3}]$, so its bits with $j \leq t+3$ determine α within 2^{-t-2}. Hence one polynomial-time oracle machine dec, the same for all U, queries these $2(t+4)$ bits and computes $\langle u|U_n|v\rangle$ to precision t, and dec^{A_U} is an entry algorithm for every U.

In polynomial space one can sum exponentially many terms and multiply exponentially many matrices, and hence compute the matrix representation of a space-uniform circuit of polynomial space. Below, $\mathrm{poly}(\cdot)$ stands for a polynomial in the listed quantities that depends on nothing else.

Lemma 2.6. *The following hold relative to every classical oracle A.*

(i) $\mathsf{PSPACE}^{\mathsf{PSPACE}^A} = \mathsf{PSPACE}^A$.

(ii) The number π, square roots of nonnegative rationals, and sines and cosines of rational angles and of rational multiples of π can be computed to precision t in space polynomial in t and the bit length of the input [Bre76].

(iii) *A sum of exponentially many dyadic numbers of polynomial bit length, each computable in polynomial space from an index of polynomial length, can be computed exactly in polynomial space.*

(iv) *The lexicographically first string of polynomial length that passes a polynomial-space test can be found in polynomial space.*

(v) *Let $S_1, \dots, S_\ell$ be $2^w \times 2^w$ matrices such that every product $S_k \cdots S_j$ with $1 \le j \le k \le \ell$ has entries of modulus at most one. Suppose that an algorithm computes the entries of each S_i to precision t, with $O(t+1)$ bits per component, in space $\mathrm{sp}(w,t)$ on input the index i, the two basis indices and 1^t, where sp is nondecreasing in t. Then an algorithm computes the entries of $S_\ell \cdots S_1$ to precision t, with $O(t+1)$ bits per component, in space*

$$\mathrm{sp}(w,\, t + (w+2)\lceil \log \ell \rceil) + \mathrm{poly}(w, t, \log \ell).$$

(vi) *Let d be a power of two. The quantum Fourier transform $F_d|u\rangle = d^{-1/2} \sum_{v<d} e^{2\pi i uv/d}|v\rangle$ has entries that can be computed to precision t in space $\mathrm{poly}(\log d, t)$.*

(vii) *Let $(\mathcal{C}_x)_x$ be a space-uniform family of general quantum circuits, possibly with encoded gates, with oracle gates O_A and with space polynomial in $|x|$. Then the entries of the matrix representation of $\mathcal{C}_x$ can be computed to precision t, with $O(t+1)$ bits per component, on input x, the four basis indices and 1^t, in space polynomial in $|x| + t$.*

Proof. A polynomial-space machine can run a subroutine on an input of polynomial length, keep a counter over exponentially many indices, and keep a running sum or the current candidate. This proves (iii) and (iv), and it proves (i) because the queries of the outer machine have polynomial length. Item (ii) is standard multiple-precision arithmetic [Bre76], and (vi) follows from (ii).

Item (v) is the recursion of Savitch [Sav70] applied to matrices. To compute an entry of $S_k \cdots S_{k'}$ to precision t', we split the product at its midpoint, compute the entries of the two halves to precision $t' + w + 2$, sum the 2^w products of entries, and round each component of the sum to $t' + 2$ fractional bits. The true entries have modulus at most one, so each product has error at most $3 \cdot 2^{-t'-w-2}$ and the sum at most $\frac{3}{4} 2^{-t'}$, and the rounding adds at most $\sqrt{2}\, 2^{-t'-3} < \frac{1}{4} 2^{-t'}$. The depth of the recursion is $\lceil \log \ell \rceil$, so the factors are computed at precision at most $t + (w+2)\lceil \log \ell \rceil$. Each level stores the ends of its interval, its basis indices, a counter over the 2^w intermediate indices, one computed entry and the partial sum, and only one entry of a factor is computed at a time, which gives the stated space.

For item (vii), let p be a polynomial bound on the space of $\mathcal{C}_x$. A table, updated with each operation of $\mathcal{C}_x$, assigns the at most p qubits present at each point to p fixed wires, and the other wires are in state $|0\rangle$. Each operation is then a channel on these wires: a unitary or oracle gate acts by conjugation, an encoded gate by its given matrix representation, a measurement of a wire has the two projectors of this wire as Kraus operators, and a new qubit and a discarded qubit both act as the reset of a wire, with Kraus operators $|0\rangle\langle 0|$ and $|0\rangle\langle 1|$. The matrix representations of these channels are the factors for item (v), with $2p$-bit indices, and their entries can be computed in polynomial space, since the generator, run with a counter, gives any single operation. The hypothesis of item (v) holds because a product of these factors is the matrix representation of a channel, with Kraus operators M_i say, and

$$\Big| \sum_i \langle u | M_i | u' \rangle \langle v' | M_i^\dagger | v \rangle \Big| \le 1$$

by the Cauchy–Schwarz inequality and $\sum_i M_i^\dagger M_i = I$. A bound on the operator norm would not suffice, since the matrix representation of a reset on p wires has norm $2^{p/2}$. An entry of the matrix

representation of $\mathcal{C}_x$ is the sum, over the at most 2^p basis states of the wires outside the output, of entries of the product with the wires outside the input at $|0\rangle$. We compute these to precision $t+p+1$, with error at most 2^{-t-1} in the sum, and round to $t+2$ fractional bits. $\square$

2.4 Approximation of Unitaries and Channels

The classes of Definition 2.3 require a bound on the whole space, while our constructions give a bound on the zero-ancilla subspace. The next lemma converts the second bound into the first.

Lemma 2.7. *Let $Y_1, Y_2 \colon \mathcal{K} \to \mathcal{H}$ be isometries with $\|Y_1 - Y_2\| \le \varepsilon < 1$. There is a unitary W on $\mathcal{H}$ with $WY_1 = Y_2$ and $\|W - I\| \le 2\varepsilon + \varepsilon^2$. Consequently, if a unitary circuit C implements a unitary U on the zero-ancilla subspace with error $\varepsilon < 1$, then U has an extension $\widetilde{U}$ with $\|C - \widetilde{U}\| \le 3\varepsilon$. The same holds for a dilation with garbage state $|g\rangle$ if $\|C\iota_a - U \otimes |g\rangle\| \le \varepsilon$.*

Proof. We extend $Y_2 Y_1^\dagger$ from the range of Y_1 to the whole space and take the unitary part of its polar decomposition. Let $\Pi_i = Y_i Y_i^\dagger$ and $X = Y_2 Y_1^\dagger + (I - \Pi_2)(I - \Pi_1)$. Since $(I - \Pi_1)Y_1 = 0$ and $(I - \Pi_2)Y_2 = 0$, we have

$$XY_1 = Y_2, \qquad X - I = (Y_2 - Y_1)Y_1^\dagger - \Pi_2(I - \Pi_1), \qquad X^\dagger X = I - (I - \Pi_1)\Pi_2(I - \Pi_1).$$

Put $E = (I - \Pi_1)Y_2 = (I - \Pi_1)(Y_2 - Y_1)$, so that $\|E\| \le \varepsilon$. Then $\Pi_2(I - \Pi_1) = Y_2 E^\dagger$ and $(I - \Pi_1)\Pi_2(I - \Pi_1) = EE^\dagger$, hence $\|X - I\| \le 2\varepsilon$ and $(1 - \varepsilon^2)I \preceq X^\dagger X \preceq I$. So X is invertible and $W = X(X^\dagger X)^{-1/2}$ is unitary. The identity for $X^\dagger X$ gives $(X^\dagger X)Y_1 = Y_1$, so $(X^\dagger X)^{-1/2} Y_1 = Y_1$ and $WY_1 = XY_1 = Y_2$. Finally $X = W(X^\dagger X)^{1/2}$ gives $\|W - X\| = \|I - (X^\dagger X)^{1/2}\| \le 1 - \sqrt{1 - \varepsilon^2} \le \varepsilon^2$, so $\|W - I\| \le 2\varepsilon + \varepsilon^2$.

For the second statement, apply this to $Y_1 = C\iota_a$ and $Y_2 = \iota_a U$ and put $\widetilde{U} = WC$. Then $\widetilde{U}\iota_a = \iota_a U$ and $\|C - \widetilde{U}\| = \|W - I\| \le 3\varepsilon$. For a dilation, take $Y_2 = U \otimes |g\rangle$ instead. $\square$

Isometries that are close also give channels that are close. For unit vectors, $\|\psi - \psi'\|_1 \le 2\||\psi\rangle - |\psi'\rangle\|$, because $\psi - \psi' = (|\psi\rangle - |\psi'\rangle)\langle\psi| + |\psi'\rangle(\langle\psi| - \langle\psi'|)$. So for isometries Y, Y' at distance ε into an output register and a register E, the channels $\mathrm{Tr}_{\mathsf{E}}(Y \cdot Y^\dagger)$ and $\mathrm{Tr}_{\mathsf{E}}(Y' \cdot Y'^\dagger)$ are at diamond distance at most 2ε. Likewise an acceptance probability changes by at most 2ε when the output vector moves by ε, and composing with a channel does not increase the diamond distance.

The Solovay–Kitaev theorem says that short words over a suitable finite set approximate every one-qubit gate of determinant one. Here a word over a set $\mathcal{S}$ of matrices is a finite list of its elements, multiplied in application order, and its length is its number of letters.

Fact 2.8 ([DN06, Theorem 1])**.** *Let $\mathcal{S} \subset \mathrm{SU}(2)$ be a finite inverse-closed set that generates a dense subgroup of $\mathrm{SU}(2)$. There is a polynomial $p_{\mathcal{S}}$ such that every $W \in \mathrm{SU}(2)$ is within 2^{-t} of a word over $\mathcal{S}$ of length at most $p_{\mathcal{S}}(t)$.*

2.5 Unitary Synthesis Problems and the Channel-Model Classes

In Sections 6 and 7 we use the model of Bostanci, Efron, Metger, Poremba, Qian and Yuen [BEM+26], in which their questions about collapse and interaction are posed. The map to be implemented is indexed by a string and may be defined only on a subspace, circuits may measure and discard, and the error is $1/r$ in diamond norm. In this model circuits have no oracle gates unless stated otherwise.

A partial isometry from a register A to a register B is a map $U = Y\Pi$ with Π a projector on A and $Y \colon \mathsf{A} \to \mathsf{B}$ an isometry [BEM+26, Definition 2.3]. Its support $\operatorname{supp} U$ is the range of Π. A

channel completion of U is a channel Φ with $\Phi(\Pi\rho\Pi) = U\rho U^\dagger$ for every state ρ [BEM+26, Definition 2.4], so it is prescribed on the support and arbitrary off it.

Definition 2.9. A *unitary synthesis problem* [BEM+26, Definition 3.1] is a family $U = (U_x)_{x\in\{0,1\}^*}$ of partial isometries from a register A_x of $n_{\text{in}}(x)$ qubits to a register B_x of $n_{\text{out}}(x)$ qubits, both polynomial in $|x|$. The string x is the *instance*, and Π_x is the projector onto $\operatorname{supp} U_x$. The problem is *size-computable* if $x \mapsto (n_{\text{in}}(x), n_{\text{out}}(x))$ is computable in polynomial time.

A problem U belongs to $\mathsf{unitaryPSPACE}^\diamond$ [BEM+26, Definitions 3.2 and 3.8] if there is a space-uniform family $(\mathcal{C}_{x,r})_{x,r}$ of general quantum circuits of space polynomial in $|x| + r$, with input register A_x and output register B_x, such that, for all x and $r \geq 1$, some channel completion $\Phi_{x,r}$ of U_x satisfies

$$\|\mathcal{C}_{x,r} - \Phi_{x,r}\|_\diamond \leq 1/r.$$

The class $\mathsf{unitaryBQP}^\diamond$ is defined in the same way with time-uniform families, and $\mathsf{unitaryBQP}^{\diamond,L}$ allows oracle gates O_L as well. We call this setting the *channel model*.

Since $T^\dagger = T^7$, these are the classes of [BEM+26], whose gate set is $\{H, T, \text{CNOT}\}$. The marker $\diamond$ distinguishes them from the classes of Definition 2.3, since the name $\mathsf{unitaryPSPACE}$ denotes both in the literature.

2.6 The Interactive Class

In an interactive implementation a polynomial-time verifier keeps the quantum input and interacts with an untrusted prover. The input may be entangled with a register that neither party holds, called the purifying register R, and the output of an accepted run, together with R, must be correct.

Definition 2.10. A *verifier* is a sequence of general quantum circuits, one for each round, on an ancilla register, a message register, an acceptance qubit and the output register B_x. It starts with all its registers at zero except the input, which occupies the first $n_{\text{in}}(x)$ qubits of its ancilla register [BEM+26, Section 2.4]. A *prover* $\mathcal{P}$ consists of an initial state of a private memory M, independent of $\mathsf{A}_x\mathsf{R}$, and a sequence of channels on the message register and M, so the memory persists through the interaction. A unitary synthesis problem U belongs to $\mathsf{unitaryQIP}^\diamond_{1,1/2}$ [BEM+26, Definition 4.1] if a deterministic polynomial-time machine maps $(x, 1^r)$ to a verifier $V_{x,r}$ such that the following hold for all x and $r \geq 1$.

Completeness. Some prover, the *honest prover*, is accepted with probability one on every input supported in $\operatorname{supp} U_x$.

Soundness. For every pure state ψ on $\mathsf{A}_x\mathsf{R}$ and every prover $\mathcal{P}$ that is accepted on ψ with probability at least $1/2$, there is a channel completion Φ of U_x with

$$D(\sigma, (\Phi \otimes \mathrm{id})(\psi)) \leq 1/r,$$

where σ is the state of $\mathsf{B}_x\mathsf{R}$ conditioned on acceptance.

Completeness asks only for acceptance, because soundness then bounds the error of the honest output.

3 Implementing Unitaries from Their Entries

By Fact 2.5, every language in $\mathsf{unitaryPSPACE}^A$ has entries computable in polynomial space relative to A. To place it in $\mathsf{unitaryP}^{\mathsf{PSPACE}^A}$, which is the main inclusion of Theorem 1.1, it therefore suffices to implement a unitary language from its entries.

Theorem 3.1. *Let A be a classical oracle and let U be a unitary language with entries computable in polynomial space relative to A. There are a language $B \in \mathsf{PSPACE}^A$ and a polynomial-time algorithm that maps $(1^n, 1^m)$ to a unitary circuit $C_{n,m}$ over $\mathcal{G}$ such that, for all $n, m \geq 1$, some extension $\widetilde{U}$ of U_n satisfies*

$$\|C^B_{n,m} - \widetilde{U}\| \leq 2^{-m}.$$

The algorithm that produces $C_{n,m}$ is the same for every A and every U, and only B depends on them.

The circuits come from a recursion on the dimension. One stage implements a unitary on d coordinates by one call to a family of unitaries on $d - s$ coordinates, with $s \geq d/\operatorname{poly}(n)$, and a control register selects the member of this family in superposition. Polynomially many stages therefore reach dimension one, and since the error of the call enters with coefficient one, the errors of the stages only add. The call is preceded by one map and followed by the adjoint of another. In a Fourier basis these two maps have few nonzero entries per row, so the index of a prepared column can be removed coherently. This recursive scheme comes from [Ope26], and we prove here every fact we use about it.

The gate list of the resulting circuit depends only on the sizes, and the unitary enters only through the one-qubit gates that the circuit applies for each value of its control registers. We therefore construct the circuit here with these gates left as parameters, and prove it correct for their exact values. In Section 4 we compute the parameters in polynomial space and compile them into the language B.

3.1 Parameterized Circuits and the Recursion

The one-qubit gates selected by control strings are uniformly controlled gates [BVMS05], also called multiplexors [SBM06]. Such a gate has a control register, a target register and a unitary W_y on the target register for each control string y, and it acts as

$$\bigoplus_y W_y := \sum_y |y\rangle\langle y| \otimes W_y.$$

For example, a diagonal unitary on a register is one such gate, with one qubit as the target and the others as controls. Since the norm of a direct sum of matrices is the largest norm of its summands, a bound that holds for every control string holds for the whole gate.

A parameterized circuit is a unitary circuit over $\mathcal{G}$ and uniformly controlled gates with one-qubit targets in which the unitaries W_y are left unspecified. These unitaries are its parameters. There can be exponentially many of them, but the registers, the gate order and the control lists form a short description. In our circuits this description depends only on the sizes, and a polynomial-time algorithm writes it (Lemma 3.8).

The recursion works in an n-qubit register, the data register, on the subspace $\mathcal{H}_d$ spanned by $|0\rangle, \ldots, |d-1\rangle$ for some $d \leq 2^n$. For a unitary T on a subspace of the state space of the non-ancilla qubits, such as $\mathcal{H}_d$, a circuit C implements T on the zero-ancilla subspace with error ε if $\|C\iota_a - \iota_a T\| \leq \varepsilon$ on the subspace of T. A parameterized circuit can implement a whole family of

unitaries with the same description: if, with parameters that depend on a string y, it implements T_y with error ε_y, then adding a control register that holds y to each uniformly controlled gate gives a parameterized circuit that implements $\bigoplus_y T_y$ with error $\max_y \varepsilon_y$.

For a split $d = k + s$, the leading coordinates $c = 1, \ldots, k$ are the basis states $|c-1\rangle$ and the trailing coordinates $\lambda = 1, \ldots, s$ are the basis states $|k+\lambda-1\rangle$. A matrix on $\mathcal{H}_d$ then has four blocks, and its lower right $s \times s$ block is its trailing block.

Definition 3.2. Fix n, an integer $N \geq 1$ and $\beta = nN + \lceil \log(N+1) \rceil$, so that $2^\beta > N 2^{nN}$. The recursion runs in *stages* $j = 0, 1, \ldots$. Stage j implements a unitary $U^{(j)}$ on $\mathcal{H}_{d_j}$, with $d_0 = 2^n$ and $U^{(0)} = U_n$, and $d_j = k_j + s_j$ is split into leading and trailing coordinates. It has a *control register* that holds a *control string* $z \in \mathbb{Z}_{2^\beta}^{N+1}$, and z selects the unitary $U^{(j+1)}(z)$ on $\mathcal{H}_{k_j}$ that stage $j+1$ implements. Thus $U^{(j)}$ depends on the control strings $z_{<j}$ of all earlier stages, and we write $U^{(j)}$ for $U^{(j)}(z_{<j})$ when these are fixed.

Given $U^{(j)}$ on $\mathcal{H}_d$ with $d \geq 2$, we find unitaries $P_{\mathrm{L}}, P_{\mathrm{R}}$ with short descriptions such that the trailing block of $G = P_{\mathrm{L}} U^{(j)} P_{\mathrm{R}}$ has norm at most $3/4$ (Section 3.2). From the blocks of G we define, by a geometric series in the trailing block, the unitaries $U^{(j+1)}(z)$ and two maps $\mathcal{E}_N, \mathcal{E}'_N$ from $\mathcal{H}_d$ to the control register and the leading coordinates. These maps are close to isometries, and $(\bigoplus_z U^{(j+1)}(z))\mathcal{E}_N$ is close to $\mathcal{E}'_N G$ (Section 3.3). Hence G is close to $\mathcal{E}'^{\dagger}_N(\bigoplus_z U^{(j+1)}(z))\mathcal{E}_N$. In the Fourier basis of the control register the two maps have few nonzero entries per row, so they can be implemented by circuits $C_{\mathcal{E},j}, C_{\mathcal{E}',j}$ (Section 3.4), and

$$C_j = P_{\mathrm{L}}^\dagger C_{\mathcal{E}',j}^\dagger C_{j+1} C_{\mathcal{E},j} P_{\mathrm{R}}^\dagger \tag{3.1}$$

implements $U^{(j)}$ if C_{j+1} implements $\bigoplus_z U^{(j+1)}(z)$ (Section 3.5). At dimension one $U^{(j)}$ is a phase, and C_j is one diagonal gate that applies it.

3.2 A Contractive Trailing Block

The unitaries of the next stage are defined by a series in the trailing block of the current unitary, and this series converges geometrically when the block has norm bounded away from one. The trailing block can have norm one, as the identity shows. We therefore multiply the unitary on both sides by unitaries with short descriptions that reduce this norm to at most $3/4$. We also want the block to be large, because its coordinates are the ones that a stage removes: with $s \geq d/\operatorname{poly}(n)$, polynomially many stages reach dimension one. For $2 \leq d \leq 2^n$, we take

$$d' = 2^{\lfloor \log d \rfloor}, \qquad s = \max\{1, \lfloor d'/(2^{20}(n+1)^2) \rfloor\}, \qquad k = d - s, \qquad \kappa = \frac{3}{4}. \tag{3.2}$$

Lemma 3.3. *Let $2 \leq d \leq 2^n$, and let U be a unitary on $\mathcal{H}_d$ whose entries a deterministic algorithm computes to any requested precision. There are unitaries $P_{\mathrm{L}}, P_{\mathrm{R}}$ on $\mathcal{H}_d$ such that the trailing block of $G = P_{\mathrm{L}} U P_{\mathrm{R}}$ has norm at most κ. Extended by the identity outside $\mathcal{H}_d$, these unitaries and their adjoints are implemented exactly, with ancillas returned to zero, by parameterized circuits of $O(n)$ uniformly controlled gates on $O(n)$ ancillas whose gate lists depend only on n and d. The parameters can be computed with polynomial additional space, calling the algorithm for the entries of U only at the precision $t_0 = 16(n+1)$.*

For $s = 1$, swapping two rows is enough. For $s > 1$ we take each of $P_{\mathrm{L}}, P_{\mathrm{R}}$ to be a sign diagonal and a Hadamard transform, and random signs suffice. The signs of P_{R} make all rows of the trailing $d' \times s$ submatrix Γ of $U P_{\mathrm{R}}$ small. The trailing block of G is then a sum of matrices built from these

rows, with independent signs from P_{L}, and small rows make the variance of this sum small. The second moment alone is not enough, because it controls only the Frobenius norm, whose square has expectation as large as s^2/d'. That would limit s to about $\sqrt{d'}$ and require about $2^{n/2}$ stages, whereas moments of degree about n allow $s = d'/\operatorname{poly}(n)$. A moment of degree 2ν is a polynomial of degree 2ν in the signs, so 2ν-wise independent signs have the same moments. Signs from seeds of polynomial length therefore suffice, and a search over seeds replaces the random choice.

Lemma 3.4. *Let ν be a positive integer and let $(\xi_i)_i$ be finitely many 2ν-wise independent unbiased signs.*

(i) If $\alpha_i \in \mathbb{C}$ and $\sum_i |\alpha_i|^2 \le \tau$, then $\mathbb{E}|\sum_i \xi_i \alpha_i|^{2\nu} \le (4\nu\tau)^\nu$.

(ii) If Θ_i are Hermitian $d \times d$ matrices and $\sum_i \Theta_i^2 \preceq \tau I$, then $\mathbb{E}\operatorname{Tr}(\sum_i \xi_i \Theta_i)^{2\nu} \le d(4\nu\tau)^\nu$.

Proof. The quantities in both bounds are polynomials of degree 2ν in the signs, so their expectations are the same as for independent signs, and we assume independence. Part (i) is the case $d = 2$ of part (ii): for $\Theta_i = \left(\begin{smallmatrix} 0 & \alpha_i \\ \bar{\alpha}_i & 0 \end{smallmatrix}\right)$ we have $\Theta_i^2 = |\alpha_i|^2 I$ and $(\sum_i \xi_i \Theta_i)^2 = |\sum_i \xi_i \alpha_i|^2 I$.

For part (ii), let $X = \sum_i \xi_i \Theta_i$. We show that $\mathbb{E}\operatorname{Tr} X^{2k} \le (2k-1)\tau\, \mathbb{E}\operatorname{Tr} X^{2k-2}$ for $k \ge 1$. Starting from $\operatorname{Tr} X^0 = d$, this gives $\mathbb{E}\operatorname{Tr} X^{2\nu} \le d(2\nu-1)!!\,\tau^\nu \le d(4\nu\tau)^\nu$. For $k = 1$, independence gives $\mathbb{E}\operatorname{Tr} X^2 = \sum_i \operatorname{Tr}\Theta_i^2 \le d\tau$. For $k \ge 2$, let $h = 2k-2$. Fix i and let X_+, X_- be X with $\xi_i = 1, -1$, so that $X_+ - X_- = 2\Theta_i$. Averaging over ξ_i and writing $X_+^{h+1} - X_-^{h+1} = \sum_{l=0}^h X_+^l (X_+ - X_-) X_-^{h-l}$ gives

$$\mathbb{E}\operatorname{Tr} X^{2k} = \sum_i \mathbb{E}\,\xi_i \operatorname{Tr}(\Theta_i X^{h+1}) = \sum_i \mathbb{E} \sum_{l=0}^h \operatorname{Tr}(\Theta_i X_+^l \Theta_i X_-^{h-l}).$$

In eigenbases $X_+|e_u\rangle = \lambda_u |e_u\rangle$ and $X_-|f_v\rangle = \lambda'_v |f_v\rangle$, the weighted inequality of arithmetic and geometric means, for even h, gives

$$\operatorname{Tr}(\Theta_i X_+^l \Theta_i X_-^{h-l}) = \sum_{u,v} |\langle f_v|\Theta_i|e_u\rangle|^2 \lambda_u^l \lambda_v'^{h-l} \le \frac{l}{h}\operatorname{Tr}(\Theta_i^2 X_+^h) + \frac{h-l}{h}\operatorname{Tr}(\Theta_i^2 X_-^h).$$

Summing over l gives $h+1$ times the average over ξ_i of $\operatorname{Tr}(\Theta_i^2 X^h)$. Summing over i and using $\sum_i \Theta_i^2 \preceq \tau I$ and $X^h \succeq 0$ proves the claim. □

Proof of Lemma 3.3. Suppose first that $s > 1$, and let $\nu = 4(n+1)$. Let U' be the trailing $d' \times d'$ submatrix of U and let H_s consist of the last s columns of the Hadamard transform $H^{\otimes \log d'}$, whose entries are $\pm 1/\sqrt{d'}$. For sign diagonals Ξ_1, Ξ_2 of size d', let P_{L} and P_{R} act on the last d' coordinates of $\mathcal{H}_d$ as $H^{\otimes \log d'}\Xi_1$ and $\Xi_2 H^{\otimes \log d'}$, and as the identity on the other coordinates. The Hadamard transform is real and symmetric, so its last s rows form $H_s^\dagger$, and the trailing block of G is

$$G_{22} = H_s^\dagger \Xi_1 \Gamma, \qquad \Gamma = U' \Xi_2 H_s. \tag{3.3}$$

To bound its norm, let $\Theta = \left(\begin{smallmatrix} 0 & G_{22} \\ G_{22}^\dagger & 0 \end{smallmatrix}\right)$. The eigenvalues of Θ are $\pm$ the singular values of G_{22}, so $\|G_{22}\|^{2\nu} \le \operatorname{Tr}\Theta^{2\nu}$, and it suffices to make $\operatorname{Tr}\Theta^{2\nu}$ smaller than $2^{-2\nu}$.

To search for the signs in polynomial space, we take them from seeds of polynomial length. A seed is a polynomial f of degree below 2ν over the field with d' elements, and we represent this field by the first monic irreducible binary polynomial of degree $\log d'$. Its signs are $\xi_i = (-1)^{\phi(f(i))}$, where i runs over the field, identified with the last d' coordinates, and ϕ is a fixed nonzero $\mathbb{F}_2$-linear functional. For a uniformly random seed these signs are 2ν-wise independent and unbiased [Vad12, Remark 3.25, Construction 3.32 and Proposition 3.33]. To test a pair of seeds, we use the entries of U at precision

t_0. Let $\widehat{U}'$ be the matrix of these computed entries, let $\widehat{G}_{22} = H_s^\dagger \Xi_1 \widehat{U}' \Xi_2 H_s$, and let $\widehat{\Theta} = \left(\begin{smallmatrix} 0 & \widehat{G}_{22} \\ \widehat{G}_{22}^\dagger & 0 \end{smallmatrix}\right)$. Two entries of H_s multiply to $\pm 1/d'$, so the entries of $\widehat{G}_{22}$ are dyadic numbers within $d'2^{-t_0}$ of those of G_{22}, and $\|\widehat{G}_{22} - G_{22}\| \le \varepsilon$ with $\varepsilon = sd'2^{-t_0} \le 2^{2n-t_0}$. Since $\|\widehat{\Theta} - \Theta\| \le \varepsilon$, $\|\Theta\| \le 1$ and $(1+\varepsilon)^{2\nu} \le 2$, writing $\widehat{\Theta}^{2\nu} - \Theta^{2\nu}$ as a sum of 2ν terms with one factor $\widehat{\Theta} - \Theta$ each gives

$$\left|\operatorname{Tr}\widehat{\Theta}^{2\nu} - \operatorname{Tr}\Theta^{2\nu}\right| \le 2s \cdot 2\nu\,\varepsilon(1+\varepsilon)^{2\nu} \le 2^{n+1} \cdot 4\nu \cdot 2^{2n-t_0} < 2^{-2\nu}/4.$$

The number $\operatorname{Tr}\widehat{\Theta}^{2\nu}$ is a sum of products of entries of $\widehat{\Theta}$ and can be computed exactly. The algorithm takes the lexicographically first pair of seeds with $\operatorname{Tr}\widehat{\Theta}^{2\nu} \le 2^{-2\nu}/2$. For this pair $\operatorname{Tr}\Theta^{2\nu} \le \frac{3}{4}2^{-2\nu}$, and hence $\|G_{22}\| < 1/2$.

It remains to show that some pair of seeds satisfies this inequality. We choose the seed of Ξ_2 first. An entry of Γ is a sum over the signs of Ξ_2 whose squared coefficients sum to at most $1/d'$, because the rows of U' have norm at most one. By Markov's inequality for the moment of degree 2ν and Lemma 3.4(i) with $\tau = 1/d'$, the event $|\Gamma_{il}|^2 > 32(n+1)/d'$ has probability at most $(4\nu/(32(n+1)))^\nu = 2^{-\nu}$. There are at most 2^{2n} entries and $2^{2n-\nu} < 1$, so some seed gives $|\Gamma_{il}|^2 \le 32(n+1)/d'$ for all entries. For this seed, every row Γ_i of Γ satisfies $\|\Gamma_i\|^2 \le \tau$, now with $\tau = 32s(n+1)/d'$.

Now fix this Ξ_2 and choose the seed of Ξ_1. With $(H_s)_i$ the rows of H_s and ξ_i the signs of Ξ_1, Equation (3.3) reads $G_{22} = \sum_i \xi_i (H_s)_i^\dagger \Gamma_i$, so $\Theta = \sum_i \xi_i \Theta_i$ with $\Theta_i = \left(\begin{smallmatrix} 0 & (H_s)_i^\dagger \Gamma_i \\ \Gamma_i^\dagger (H_s)_i & 0 \end{smallmatrix}\right)$. Since $\|\Gamma_i\|^2 \le \tau$, $\|(H_s)_i\|^2 = s/d' \le \tau$, $H_s^\dagger H_s = I$ and $\Gamma^\dagger\Gamma \preceq I$,

$$\sum_i \Theta_i^2 = \begin{pmatrix} \sum_i \|\Gamma_i\|^2 (H_s)_i^\dagger (H_s)_i & 0 \\ 0 & (s/d')\,\Gamma^\dagger\Gamma \end{pmatrix} \preceq \tau I.$$

Since $s \le d'/(2^{20}(n+1)^2)$, we have $4\nu\tau \le 2^{-11}$, and Lemma 3.4(ii) in dimension $2s \le 2^{n+1}$ gives $\mathbb{E}\operatorname{Tr}\Theta^{2\nu} \le 2s\,2^{-11\nu} < 2^{-2\nu}/4$. Some seed of Ξ_1 therefore has $\operatorname{Tr}\Theta^{2\nu} < 2^{-2\nu}/4$. Then $\operatorname{Tr}\widehat{\Theta}^{2\nu} < 2^{-2\nu}/2$, as required.

Suppose now that $s = 1$. Some entry of the last column of U has squared modulus at most $1/d \le 1/2$. The squared moduli of the entries of this column at precision t_0 are within $3 \cdot 2^{-t_0} \le 1/64$ of the true ones, and the algorithm selects the first row whose computed value is at most $1/2 + 1/64$. Its entry has squared modulus at most $1/2 + 1/32 < \kappa^2$. Let P_{L} be the transposition of this row and the last row, and let $P_{\mathrm{R}} = I$. The trailing block of G is then this entry, of modulus below κ.

It remains to implement $P_{\mathrm{L}}, P_{\mathrm{R}}$. A permutation π of n-bit strings uses $O(n)$ uniformly controlled gates and n ancillas returned to zero: gates controlled on u write $\pi(u)$ on the ancillas, gates controlled on $v = \pi(u)$ add $\pi^{-1}(v)$ to the data register, which clears it, and CNOT gates swap the two registers. For $s = 1$ this gives the transposition. For $s > 1$, the permutation $u \mapsto u - (d - d') \bmod 2^n$ moves the last d' coordinates of $\mathcal{H}_d$ to the first d' basis states. Then, controlled on the high $n - \log d'$ bits being zero, one diagonal gate applies the signs and $\log d'$ gates apply the Hadamard transform on the low bits, the signs first for P_{L} and the Hadamard transform first for P_{R}. The inverse permutation follows, and the result acts as the identity outside $\mathcal{H}_d$. An adjoint reverses the gates and takes the adjoints of the parameters.

The gate lists depend only on n and d, and the parameters are determined by the selected pair of seeds or the selected row. This selection is a function of the outputs at precision t_0 alone. The seeds, their signs, the terms of $\widehat{G}_{22}$ and the products in $\operatorname{Tr}\widehat{\Theta}^{2\nu}$ are enumerated with polynomially many bits, so by Lemma 2.6(iii) and (iv) the selection takes polynomial additional space. □

Because rounding down a number that is at least one loses at most a factor of two, and because $d' > d/2$, Equation (3.2) gives $s \ge d/(2^{22}(n+1)^2)$ in all cases. Each stage therefore multiplies the

dimension by at most $1-1/(2^{22}(n+1)^2)$, and dimension one is reached in fewer than $J = 2^{23}(n+1)^3$ stages, because J such factors would reduce 2^n to at most $2^n e^{-2(n+1)} < 1$.

3.3 Smaller Unitaries and Two Near-Isometries

Let G be a unitary on $\mathcal{H}_d$ with blocks $G_{11}, G_{12}, G_{21}, G_{22}$ for the split $d = k + s$ and with $\|G_{22}\| \le \kappa$. This bound lets us eliminate the trailing coordinates and still obtain a unitary. For a diagonal unitary Z on the trailing coordinates, $I - G_{22}Z$ is invertible, and we put

$$X = (I - G_{22}Z)^{-1}G_{21}, \qquad U^{(j+1)} = G_{11} + G_{12}ZX. \tag{3.4}$$

To see that $U^{(j+1)}$ is a unitary on the leading coordinates, let $E_\infty = \binom{I}{ZX}$. Since $(I-G_{22}Z)X = G_{21}$, the lower block of GE_∞ is $G_{21} + G_{22}ZX = X$, so

$$GE_\infty = \begin{pmatrix} U^{(j+1)} \\ X \end{pmatrix}. \tag{3.5}$$

As G is unitary, $I + X^\dagger X = E_\infty^\dagger E_\infty = U^{(j+1)\dagger}U^{(j+1)} + X^\dagger X$, and hence $U^{(j+1)\dagger}U^{(j+1)} = I$.

To relate G to $U^{(j+1)}$, let $X' = XU^{(j+1)\dagger}$ and $E'_\infty = \binom{I}{X'}$, so that $GE_\infty = E'_\infty U^{(j+1)}$. Taking adjoints and multiplying by $U^{(j+1)}$ on the left and by G on the right gives

$$U^{(j+1)}E_\infty^\dagger = E_\infty'^\dagger G. \tag{3.6}$$

This would express G through $U^{(j+1)}$ if $E_\infty^\dagger$ and $E_\infty'^\dagger$ were isometries, but they map d coordinates to $k < d$ and so cannot be. To supply the missing dimensions, we let Z depend on a control string z and take a superposition over all control strings. For a suitable family of diagonal unitaries, averaging over the control strings cancels the cross terms between different degrees of the series, and the resulting maps are close to isometries.

Let $\omega = e^{2\pi \mathrm{i}/2^\beta}$ with β as in Definition 3.2. Each trailing coordinate λ has the frequency vector $v_\lambda = (1, \lambda, \dots, \lambda^N)$, and a control string $z \in \mathbb{Z}_{2^\beta}^{N+1}$ gives

$$Z(z) = \operatorname{diag}\big(\omega^{v_\lambda \cdot z}\big)_{\lambda=1,\dots,s}.$$

We write $X(z)$, $U^{(j+1)}(z)$, $E_\infty(z)$, $E'_\infty(z)$ for the objects above at $Z = Z(z)$. We truncate the Neumann series of $X(z)$ after N terms, so that only paths of at most N trailing coordinates occur in it:

$$E_N(z) = \begin{pmatrix} I \\ \sum_{h<N} Z(z)(G_{22}Z(z))^h G_{21} \end{pmatrix}, \qquad \|E_\infty(z) - E_N(z)\| \le \frac{\kappa^N}{1-\kappa} = 4\kappa^N.$$

The truncation of $E'_\infty(z)$ comes from the same formula applied to $G^\dagger$. Indeed, the lower block of the identity $G^\dagger E'_\infty = E_\infty U^{(j+1)\dagger}$ reads $G_{12}^\dagger + G_{22}^\dagger X' = ZX'$, so $X' = (I - Z^\dagger G_{22}^\dagger)^{-1}Z^\dagger G_{12}^\dagger = Z^\dagger(I - G_{22}^\dagger Z^\dagger)^{-1}G_{12}^\dagger$. Since $Z(z)^\dagger = Z(-z)$, this says that $E'_\infty(z)$ is $E_\infty(-z)$ for the unitary $G^\dagger$, whose trailing block also has norm at most κ. We therefore let $E'_N(z)$ be $E_N(-z)$ for $G^\dagger$, and again $\|E'_\infty(z) - E'_N(z)\| \le 4\kappa^N$.

The superpositions over control strings are the two maps of the recursion, from $\mathcal{H}_d$ to the control register and the leading coordinates:

$$\mathcal{E}_N = 2^{-\beta(N+1)/2}\sum_z |z\rangle \otimes E_N(z)^\dagger, \qquad \mathcal{E}'_N = 2^{-\beta(N+1)/2}\sum_z |z\rangle \otimes E'_N(z)^\dagger. \tag{3.7}$$

By the definition of E'_N, the map $\mathcal{E}'_N$ is $\mathcal{E}_N$ for $G^\dagger$ followed by the permutation $z \mapsto -z$ of the control register.

To implement these maps we need rows with few nonzero entries, and these appear in the Fourier basis $F_{2^\beta}^{\otimes(N+1)}|\zeta\rangle = 2^{-\beta(N+1)/2}\sum_z \omega^{\zeta\cdot z}|z\rangle$ of the control register. There the maps have coefficient matrices $\Lambda_{\mathcal{E}}, \Lambda_{\mathcal{E}'}$, defined by

$$\mathcal{E}_N = (F_{2^\beta}^{\otimes(N+1)} \otimes I)\Lambda_{\mathcal{E}}, \qquad \mathcal{E}'_N = (F_{2^\beta}^{\otimes(N+1)} \otimes I)\Lambda_{\mathcal{E}'},$$

with rows indexed by a frequency $\zeta \in \mathbb{Z}_{2^\beta}^{N+1}$ and a leading coordinate c. When the series is expanded, a path through the trailing coordinates contributes to one frequency only, plus or minus the sum of the frequency vectors along the path. The powers in v_λ are chosen so that this sum determines the coordinates that the path visits. Thus the control register, introduced to supply the missing dimensions, also records the visited coordinates, and each row has few candidate columns. We call a matrix Λ an ε-near-isometry if $(1-\varepsilon)I \preceq \Lambda^\dagger\Lambda \preceq I$. A sparsity pattern of a matrix is a set of positions that contains all its nonzero entries and may contain others.

Lemma 3.5. *Let G be a unitary on $\mathcal{H}_d$, $d \le 2^n$, with $\|G_{22}\| \le \kappa$, and let $N \ge 1$. Then $\mathcal{E}_N$ and $\mathcal{E}'_N$ are κ^{2N}-near-isometries, and*

$$\Big\|\Big(\bigoplus_z U^{(j+1)}(z)\Big)\mathcal{E}_N - \mathcal{E}'_N G\Big\| \le 8\kappa^N.$$

Each of $\Lambda_{\mathcal{E}}, \Lambda_{\mathcal{E}'}$ has a sparsity pattern of at most N positions per row, which does not depend on G and is computable from the row index in space polynomial in $n+N$.

Proof. For matrices $M(z)$, the map $2^{-\beta(N+1)/2}\sum_z |z\rangle \otimes M(z)$ has as its Gram matrix the average of $M(z)^\dagger M(z)$, and therefore has norm at most $\max_z \|M(z)\|$.

Thus $\mathcal{E}_N^\dagger\mathcal{E}_N$ is the average over z of $E_N(z)E_N(z)^\dagger$, and averaging over a common phase cancels its cross terms. Every frequency vector has first coordinate one, so $Z(z) = \omega^{z_0}Z'$, where Z' depends only on $z_1, \dots, z_N$. With $R = Z'G_{22}$ we have $Z'(G_{22}Z')^h = R^hZ'$, so the lower block of $E_N(z)$ is $\sum_{h<N}\omega^{(h+1)z_0}R^hZ'G_{21}$. Fix $z_1, \dots, z_N$ and average over z_0. The average of ω^{hz_0} vanishes for $0 < |h| \le N$ because $N < 2^\beta$, so the off-diagonal blocks and all cross terms vanish, and the upper left block is I. The identity $GG^\dagger = I$ gives $G_{21}G_{21}^\dagger = I - G_{22}G_{22}^\dagger$, hence $(Z'G_{21})(Z'G_{21})^\dagger = I - RR^\dagger$, and the lower right block telescopes:

$$\sum_{h<N} R^h(I - RR^\dagger)R^{\dagger h} = I - R^NR^{\dagger N}.$$

As $\|R\| \le \kappa$, this lies between $(1-\kappa^{2N})I$ and I, and averaging over $z_1, \dots, z_N$ keeps these bounds. Thus $\mathcal{E}_N$ is a κ^{2N}-near-isometry, and so is $\mathcal{E}'_N$, which is $\mathcal{E}_N$ for $G^\dagger$ followed by a permutation.

The matrix in the second bound has the same form, and Equation (3.6) turns its summands into tails:

$$M(z) = U^{(j+1)}(z)E_N(z)^\dagger - E'_N(z)^\dagger G = U^{(j+1)}(z)(E_N(z) - E_\infty(z))^\dagger - (E'_N(z) - E'_\infty(z))^\dagger G.$$

Each tail has norm at most $4\kappa^N$, so $\|M(z)\| \le 8\kappa^N$.

For the sparsity, consider the columns of $E_N(z)^\dagger$. The column of a leading coordinate is the corresponding basis vector and does not depend on z, so in $\Lambda_{\mathcal{E}}$ it has its only nonzero entry, equal to one, at frequency $\zeta = 0$ and the same leading coordinate. The column of a trailing coordinate λ is column λ of $\sum_{h<N} G_{21}^\dagger(Z(z)^\dagger G_{22}^\dagger)^h Z(z)^\dagger$. Expanding over paths $\lambda_1, \dots, \lambda_\ell$ with $\lambda_\ell = \lambda$, the diagonal phases multiply to $\omega^{-(v_{\lambda_1}+\cdots+v_{\lambda_\ell})\cdot z}$, so the entry of $\Lambda_{\mathcal{E}}$ in row (ζ, c) and the column of λ is

$$\sum_{\ell=1}^{N} \sum_{\lambda_1,\dots,\lambda_{\ell-1}} \mathbf{1}\Big[\zeta = -\sum_{i\le\ell} v_{\lambda_i}\Big]\, \overline{(G_{21})_{\lambda_1,c}} \prod_{i<\ell} \overline{(G_{22})_{\lambda_{i+1},\lambda_i}}, \tag{3.8}$$

where the equality of frequencies is in $\mathbb{Z}_{2^\beta}^{N+1}$.

Suppose that this entry is nonzero. Then $-\zeta$ is a sum of $\ell \le N$ vectors v_{λ_i}, one of them v_λ. Each coordinate of the sum is an integer at most $N2^{nN} < 2^\beta$, so the sum is the representative of $-\zeta$ in $\{0, \dots, 2^\beta - 1\}^{N+1}$. Its first coordinate is ℓ, and its next ℓ coordinates are the power sums of $\lambda_1, \dots, \lambda_\ell$, which determine the multiset $\{\lambda_1, \dots, \lambda_\ell\}$ by Newton's identities [Mac95, Chapter I, Section 2]. Hence ζ determines a set of at most $\ell \le N$ candidates for λ, which are found by enumerating nondecreasing tuples and comparing power sums, in space polynomial in $n + N$. Moreover $1 \le \ell \le N < 2^\beta$, so $\zeta \ne 0$ for every nonzero entry in a trailing column. Hence a row $(0, c)$ has only the entry in the column of c, and a row with $\zeta \ne 0$ has no nonzero entry in a leading column, which gives the pattern of $\Lambda_{\mathcal{E}}$.

The permutation $z \mapsto -z$ sends $F_{2^\beta}^{\otimes(N+1)}|\zeta\rangle$ to $F_{2^\beta}^{\otimes(N+1)}|-\zeta\rangle$. Therefore the row (ζ, c) of $\Lambda_{\mathcal{E}'}$ is the row $(-\zeta, c)$ of $\Lambda_{\mathcal{E}}$ for $G^\dagger$, and the pattern of $\Lambda_{\mathcal{E}'}$ follows. □

3.4 Implementing a Row-Sparse Near-Isometry

To apply a matrix Λ to $|v\rangle$, we can prepare its column, but v then remains in a register. To remove it, we apply, controlled on the row u, the inverse preparation of the uniform superposition over the set I_u of positions of that row. This gives each $v \in I_u$ amplitude $|I_u|^{-1/2}$ at zero, and a flag rotation, also controlled on the row, compensates for this factor, so that every entry of Λ appears with the same factor. Thus Λ is a block of a circuit, up to an inverse-polynomial factor. We choose this factor so that N iterations of amplitude amplification raise it to one on a singular vector with singular value one, and since all singular values of Λ are close to one, the iterations give Λ up to a small error. These two steps adapt the sparse-access block encoding [GSLW19, Lemma 48], which assumes row and column access, and oblivious amplitude amplification [BCC+14, Lemma 3.6], which is stated for a unitary.

The columns and the superpositions are prepared with uniformly controlled gates, as in [GR02]. For a unit vector $\sum_u a_u|u\rangle$ on ℓ qubits, the prefix probability of a bit string y is $\mu(y) = \sum_u |a_u|^2$ over the strings u that extend y. Processing the qubits in order, at prefix y we apply to the next qubit the rotation

$$\begin{pmatrix} \sqrt{\mu(y0)/\mu(y)} & -\sqrt{\mu(y1)/\mu(y)} \\ \sqrt{\mu(y1)/\mu(y)} & \sqrt{\mu(y0)/\mu(y)} \end{pmatrix} \tag{3.9}$$

if $\mu(y) > 0$ and the identity otherwise, so that by induction the amplitude at each prefix y is $\sqrt{\mu(y)}$. A final diagonal gate applies the phases $a_u/|a_u|$, taken to be one when $a_u = 0$. This prepares the vector exactly from $|0^\ell\rangle$ with $\ell + 1$ uniformly controlled gates.

Lemma 3.6. *Let Λ be a matrix with columns indexed by $v < d$, where $d \le 2^n$, and rows indexed by $u < 2^w$. Suppose that Λ is an ε-near-isometry with $0 \le \varepsilon < 1$ and has a sparsity pattern of at most N positions per row. Consider an index register of n qubits, a row register of w qubits and two flag qubits, with the embeddings*

$$\iota_{\text{in}}|v\rangle = |v\rangle|0^w\rangle|0\rangle|0\rangle, \qquad \iota_{\text{out}}|u\rangle = |0^n\rangle|u\rangle|1\rangle|1\rangle.$$

There is a parameterized circuit C_Λ on these registers such that

$$\|C_\Lambda \iota_{\text{in}} - \iota_{\text{out}}\Lambda\| \le 5\varepsilon.$$

It has $O(N(n + w))$ uniformly controlled gates, its gate list depends only on n, w, N, and its parameters are determined by Λ and the pattern, as given in Equations (3.10) and (3.11).

Proof. Let $\theta = \pi/(4N+2)$, so that $(2N+1)\theta = \pi/2$. We first build a unitary $C_\Lambda^{(0)}$ whose block between the two embeddings is $\sin\theta \cdot \Lambda$. For $v < d$, let C_v be the preparation on the row register and the first flag with

$$C_v|0^w\rangle|0\rangle = \sum_u \Lambda_{uv}|u\rangle|1\rangle + \sqrt{1 - \|\Lambda|v\rangle\|^2}\,|0^w\rangle|0\rangle, \tag{3.10}$$

and let $C_v = I$ for $v \geq d$. For a row u, let I_u be its set of positions in the pattern, let C'_u be the preparation on the index register with

$$C'_u|0^n\rangle = |I_u|^{-1/2} \sum_{i \in I_u} |i\rangle, \tag{3.11}$$

and let $C'_u = I$ if I_u is empty. Let $W_\theta = \bigoplus_u W_{\theta,u}$ be the uniformly controlled gate with the row register as its control and the second flag as its target, where $W_{\theta,u}|0\rangle = \sqrt{1 - |I_u|\sin^2\theta}\,|0\rangle + \sqrt{|I_u|}\sin\theta\,|1\rangle$. This rotation exists because $|I_u| \leq N$ and $\sqrt{N}\sin\theta \leq \sqrt{N}\theta \leq 1$. Put

$$C_\Lambda^{(0)} = W_\theta\Big(\sum_u |u\rangle\langle u| \otimes C'^{\dagger}_u\Big)\Big(\sum_v |v\rangle\langle v| \otimes C_v\Big),$$

where the preparations C_v are controlled on the index register and the preparations $C'^{\dagger}_u$ on the row register. Apply this to $\iota_{\text{in}}|v\rangle$. The term of Equation (3.10) with first flag zero stays orthogonal to the range of ι_{out}. On a term $\Lambda_{uv}|u\rangle|1\rangle$ with $\Lambda_{uv} \neq 0$ we have $v \in I_u$, so the amplitude of zero on the index register is $\overline{\langle v|C'_u|0^n\rangle} = |I_u|^{-1/2}$, and the second flag contributes $\sqrt{|I_u|}\sin\theta$. Hence

$$\iota_{\text{out}}^\dagger C_\Lambda^{(0)} \iota_{\text{in}} = \sin\theta \cdot \Lambda. \tag{3.12}$$

To raise the factor $\sin\theta$ to nearly one, let $\Pi_{\text{in}} = \iota_{\text{in}}\iota_{\text{in}}^\dagger$ and $\Pi_{\text{out}} = \iota_{\text{out}}\iota_{\text{out}}^\dagger$, and let

$$C_\Lambda = \Big((2C_\Lambda^{(0)}\Pi_{\text{in}}C_\Lambda^{(0)\dagger} - I)(I - 2\Pi_{\text{out}})\Big)^N C_\Lambda^{(0)}.$$

This circuit acts separately on each singular vector of Λ. Take a singular value decomposition $\Lambda|v_i\rangle = \sigma_i|e_i\rangle$, with $\{|v_i\rangle\}$ an orthonormal basis of $\mathbb{C}^d$, $\{|e_i\rangle\}$ orthonormal and $\sqrt{1-\varepsilon} \leq \sigma_i \leq 1$, and identify vectors with their embeddings. By Equation (3.12),

$$C_\Lambda^{(0)}|v_i\rangle = \sin\alpha_i|e_i\rangle + \cos\alpha_i|e'_i\rangle, \qquad \alpha_i = \arcsin(\sin\theta \cdot \sigma_i),$$

with $|e'_i\rangle$ a unit vector orthogonal to the range of ι_{out}. Since $C_\Lambda^{(0)}$ is unitary and $\cos\alpha_i > 0$, the $|e'_i\rangle$ are orthonormal, so the planes spanned by $|e_i\rangle, |e'_i\rangle$ are mutually orthogonal, and the range of $C_\Lambda^{(0)}\Pi_{\text{in}}$ is spanned by one vector in each plane. Both reflections therefore preserve each plane. On the plane of i, write a unit vector as $\sin\gamma|e_i\rangle + \cos\gamma|e'_i\rangle$. Then $I - 2\Pi_{\text{out}}$ maps γ to $-\gamma$, and the other reflection, which fixes $C_\Lambda^{(0)}|v_i\rangle$ and negates the vector of the plane orthogonal to it, maps γ to $2\alpha_i - \gamma$, so one iteration maps γ to $\gamma + 2\alpha_i$. After N iterations, $C_\Lambda|v_i\rangle$ has angle $(2N+1)\alpha_i$.

Since $\sigma_i \leq 1$, we have $\alpha_i \leq \theta$ and $(2N+1)\alpha_i \leq (2N+1)\theta = \pi/2$. Thus $C_\Lambda|v_i\rangle$ is within arc length $(2N+1)(\theta - \alpha_i)$ of $|e_i\rangle$, hence within $(2N+1)(\theta-\alpha_i) + 1 - \sigma_i$ of $\sigma_i|e_i\rangle$. On $[0, \sin\theta]$, a subset of $[0, 1/2]$, the derivative of arcsin is at most 2, so $\theta - \alpha_i \leq 2\sin\theta(1-\sigma_i) \leq 2\theta(1-\sigma_i)$, and therefore

$$\|C_\Lambda|v_i\rangle - \sigma_i|e_i\rangle\| \leq (\pi+1)(1-\sigma_i) \leq 5\varepsilon,$$

using $1 - \sigma_i \leq 1 - \sqrt{1-\varepsilon} \leq \varepsilon$. These error vectors lie in mutually orthogonal planes, so the bound holds for every superposition of the $|v_i\rangle$.

Each preparation uses one gate per qubit and one diagonal gate, so $C_\Lambda^{(0)}$ uses $O(n+w)$ gates. Since $2C_\Lambda^{(0)}\Pi_{\text{in}}C_\Lambda^{(0)\dagger} - I = C_\Lambda^{(0)}(2\Pi_{\text{in}} - I)C_\Lambda^{(0)\dagger}$, the circuit C_Λ has $2N+1$ occurrences of $C_\Lambda^{(0)}$ or its adjoint and $2N$ diagonal reflections. □

3.5 The Recursive Circuit

The parameters of Lemma 3.6 are discontinuous functions of Λ, since the rotation of Equation (3.9) changes form when a prefix probability vanishes. To compute them in Section 4 we need exact zero tests, so we apply the lemma to rounded coefficient matrices, whose prefix probabilities are exact rational numbers.

Let δ be the largest power of two with $\delta \le 2^{-m}/(4096J)$, let $N = 4\log(1/\delta)$, and let β be as in Definition 3.2, the same at every stage. Then $\kappa^N < \delta$, because $\kappa^4 < 1/2$. The row register of Lemma 3.6 has $w = \beta(N+1) + n$ qubits and holds a frequency and an n-qubit basis state, and rows whose basis state is not a leading coordinate are zero. Let Λ be $\Lambda_{\mathcal{E}}$ or $\Lambda_{\mathcal{E}'}$ and let $t_\Lambda = \log(1/\delta) + n + w + 4$. Round each entry in the sparsity pattern to the 2^{-t_Λ} grid, set the other entries to zero, and let $\widehat{\Lambda}$ be the result divided by $1+\delta$. There are at most 2^{n+w} entries, so the rounded matrix differs from Λ by at most $2^{(n+w)/2-t_\Lambda} \le \delta$ in Frobenius norm. Dividing by $1+\delta$ makes it a contraction, $\|\widehat{\Lambda}\| \le 1$, at the cost of $\|\widehat{\Lambda} - \Lambda\| \le 2\delta$. For a unit vector $|\psi\rangle$,

$$\left|\|\widehat{\Lambda}|\psi\rangle\|^2 - \|\Lambda|\psi\rangle\|^2\right| \le \|(\widehat{\Lambda}-\Lambda)|\psi\rangle\|\left(\|\widehat{\Lambda}|\psi\rangle\| + \|\Lambda|\psi\rangle\|\right) \le 4\delta,$$

so by Lemma 3.5 the matrix $\widehat{\Lambda}$ is a $(\kappa^{2N}+4\delta)$-near-isometry with the same pattern.

The two circuits of a stage are $C_{\mathcal{E},j} = (F_{2^\beta}^{\otimes(N+1)} \otimes I) C_{\widehat{\Lambda}_{\mathcal{E}}}$ and $C_{\mathcal{E}',j} = (F_{2^\beta}^{\otimes(N+1)} \otimes I) C_{\widehat{\Lambda}_{\mathcal{E}'}}$, where the Fourier transforms act on the frequency part of the row register, which is then the control register. Since these transforms commute with ι_{out}, Lemma 3.6 with $\varepsilon = \kappa^{2N} + 4\delta < 1$ gives $\|C_{\mathcal{E},j}\iota_{\text{in}} - \iota_{\text{out}}\mathcal{E}_N\| \le 5\varepsilon + 2\delta = 5\kappa^{2N} + 22\delta$, and the same for $C_{\mathcal{E}',j}$ and $\mathcal{E}'_N$. After $C_{\mathcal{E},j}$, the last n qubits of the row register are the data register of the next stage.

Lemma 3.7. *Let $U^{(j)}$ be a unitary of stage j, on $\mathcal{H}_d$ with $d \ge 2$, let $P_{\text{L}}, P_{\text{R}}$ be as in Lemma 3.3, and let $G = P_{\text{L}} U^{(j)} P_{\text{R}}$. Let C_{j+1} be a unitary circuit that acts only on the control register, the data register of the next stage and its own ancillas, and that implements $\bigoplus_z U^{(j+1)}(z)$ on the zero-ancilla subspace with error ε_{j+1}. Then the circuit C_j of Equation (3.1) implements $U^{(j)}$ on the zero-ancilla subspace with error at most $\varepsilon_{j+1} + 18\kappa^N + 44\delta$.*

Proof. Let $W = \bigoplus_z U^{(j+1)}(z)$, and let $\iota_{\text{in}}, \iota_{\text{out}}$ also set the ancillas of C_{j+1} to zero. Since C_{j+1} acts as the identity on the index register and the flags, the bound $\|C_{j+1}\iota_{\text{out}} - \iota_{\text{out}}W\| \le \varepsilon_{j+1}$ holds on the control register and the leading coordinates, which contain the range of $\mathcal{E}_N$. Inserting $\iota_{\text{out}}\mathcal{E}_N$ and $\iota_{\text{out}}\mathcal{E}'_N$,

$$\begin{aligned}\|C_{j+1}C_{\mathcal{E},j}\iota_{\text{in}} - C_{\mathcal{E}',j}\iota_{\text{in}}G\| \le{}& \|C_{j+1}(C_{\mathcal{E},j}\iota_{\text{in}} - \iota_{\text{out}}\mathcal{E}_N)\| + \|(C_{j+1}\iota_{\text{out}} - \iota_{\text{out}}W)\mathcal{E}_N\| \\ &+ \|\iota_{\text{out}}(W\mathcal{E}_N - \mathcal{E}'_N G)\| + \|(\iota_{\text{out}}\mathcal{E}'_N - C_{\mathcal{E}',j}\iota_{\text{in}})G\|.\end{aligned}$$

The first and last terms are bounded by the estimate before the lemma, the second by ε_{j+1} since $\|\mathcal{E}_N\| \le 1$, and the third by $8\kappa^N$ by Lemma 3.5. Their sum is at most $\varepsilon_{j+1} + 8\kappa^N + 10\kappa^{2N} + 44\delta \le \varepsilon_{j+1} + 18\kappa^N + 44\delta$. The circuits of $P_{\text{L}}^\dagger, P_{\text{R}}^\dagger$ are exact on zero ancillas and preserve $\mathcal{H}_d$, and $U^{(j)} = P_{\text{L}}^\dagger G P_{\text{R}}^\dagger$, so

$$C_j\iota - \iota U^{(j)} = P_{\text{L}}^\dagger(C_{\mathcal{E}',j}^\dagger C_{j+1} C_{\mathcal{E},j}\iota_{\text{in}} - \iota_{\text{in}}G)P_{\text{R}}^\dagger,$$

where ι sets all ancillas of C_j to zero. The norm of this difference is the norm just bounded. □

The recursion ends at dimension one, where one diagonal gate on the data register, controlled on all control registers, applies the phase exactly. Lemma 3.7 holds for each fixed $z_{<j}$, so it holds for the direct sum over $z_{<j}$ with the same error, by adding $z_{<j}$ to the controls as in Section 3.1. Induction from the last stage gives, with fewer than J stages and $\kappa^N < \delta \le 2^{-m}/(4096J)$,

$$\varepsilon_0 \le J(18\kappa^N + 44\delta) < 2^{-m}/64. \tag{3.13}$$

The precision m enters only through $\log(1/\delta)$, which is linear in $m + \log n$, so exponential precision needs no error reduction. Nothing in the gate list of C_0 depends on U_n, so it can be written from the sizes alone.

Lemma 3.8. *There is a polynomial-time algorithm, which does not depend on A or U, that on input $(1^n, 1^m)$ outputs the registers, gate order and control lists of the parameterized circuit C_0. The circuit has $S = \mathrm{poly}(n+m)$ uniformly controlled gates and width $w' = \mathrm{poly}(n+m)$. For each gate, the algorithm also records its stage, the component it belongs to (P_{L}, P_{R}, the circuit for $\widehat{\Lambda}_{\mathcal{E}}$ or $\widehat{\Lambda}_{\mathcal{E}'}$, a Fourier transform, or the gate at dimension one), its position in that component, and whether the component occurs as an adjoint.*

Proof. The sizes d_j, s_j, k_j are computed from n by Equation (3.2), and δ, N, β, w from (n, m). The gate lists of Lemmas 3.3 and 3.6 depend only on these sizes, and the sparsity patterns enter only through the parameters. The transform F_{2^β} uses $O(\beta)$ gates, because in the textbook circuit [NC10, Section 5.1] the controlled rotations on one target commute and merge into one diagonal uniformly controlled gate. Each stage contains exactly one recursive call, so gate counts and widths add over fewer than J stages, and the gates of stage j have $j\beta(N+1)$ further control qubits. All these numbers are polynomial in $n + m$. □

4 Computing the Parameters

Let A and U be as in Theorem 3.1. With its exact parameters, the circuit C_0 of Lemma 3.8 implements U_n with error below $2^{-m}/64$. To prove the theorem, it therefore remains to compute these parameters in polynomial space and to encode them in one language. The parameters are functions of the unitaries $U^{(j)}$, and $U^{(j+1)}$ is defined from $U^{(j)}$, so computing an entry of $U^{(j+1)}$ requires entries of $U^{(j)}$, and so on down to stage zero. If an entry of $U^{(j+1)}$ at precision t needed entries of $U^{(j)}$ at precision ct for a constant $c > 1$, the precision at stage zero would be exponential in J. We show that precision $t + O(n)$ suffices. The truncated series for the inverse in Equation (3.4) has exponentially many paths, and bounding the error of each path separately would need entries of $U^{(j)}$ at a precision of order nt. We instead round the matrices before expanding the series and multiply the rounded numbers exactly, so the length of the series does not enter the precision.

4.1 The Parameters in Polynomial Space

Fix n and m. In Lemma 3.3, P_{L} and P_{R} are selected from the entries of $U^{(j)}$ computed to precision t_0, so the unitaries $U^{(j+1)}(z)$ depend on the algorithm that computes these entries and not only on $U^{(j)}$. We therefore define the unitaries and the algorithms together, by induction on j. For $j = 0$ they are U_n and the given entry algorithm. Given a deterministic algorithm for the entries of $U^{(j)}$ and fixed earlier control strings $z_{<j}$, the outputs of this algorithm at precision t_0 determine P_{L}, P_{R} and G, and Equation (3.4) defines $U^{(j+1)}(z)$. The algorithm for the entries of $U^{(j+1)}(z)$ is the one in the proof below. It uses entries of $U^{(j)}$ at other precisions, but $U^{(j+1)}(z)$ does not depend on them.

The unitaries $U^{(j)}$ determine the parameters of C_0. We number the uniformly controlled gates of C_0 by $i \in [S]$ and write the i-th gate as $W_i = \bigoplus_y W_{i,y}$.

Lemma 4.1. *There are deterministic algorithms with oracle A that use space polynomial in $n+m+t$ and, on input $1^n, 1^m, 1^t$ and the following data, compute to precision t:*

(i) the entry $\langle u|U^{(j)}(z_{<j})|v\rangle$, given $j, z_{<j}, u, v$;

(ii) each entry of the parameter $W_{i,y}$, given i, y and the two basis indices.

Proof. For item (i), we compute an entry of $U^{(j+1)}(z)$ to precision t from entries of $U^{(j)}$, with $z_{<j}$ fixed, and write $U^{(j)}$ for $U^{(j)}(z_{<j})$. Let $Z = Z(z)$ and $G' = G\,\mathrm{diag}(I, Z)$. In the blocks of G', Equation (3.4) reads

$$X = (I - G'_{22})^{-1} G'_{21}, \qquad U^{(j+1)}(z) = G'_{11} + G'_{12} X.$$

An entry of $G = P_{\mathrm{L}} U^{(j)} P_{\mathrm{R}}$ or of G' is a sum of at most 2^{2n} products of an entry of $U^{(j)}$ with at most two entries of $P_{\mathrm{L}}, P_{\mathrm{R}}$ and at most one entry of Z. All these factors have modulus at most one, and those of $P_{\mathrm{L}}, P_{\mathrm{R}}, Z$ are computed by Lemma 2.6(ii). Hence, if all factors are computed to precision $t' + 2n + 3$, then the exact sum of the products is within $2^{-t'}$ of the entry.

Let $t_1 = t + n + 10$, and let $\widehat{G}'$ be G' with its entries rounded to the 2^{-t_1} grid. The blocks have dimensions at most 2^n, so each block of $\widehat{G}'$ is within $\varepsilon = 2^{-t-10}$ of the exact block in operator norm, and $\|\widehat{G}'_{22}\| < 4/5$ because $\|G'_{22}\| \le \kappa$. Let $N' = 4(t+10)$ and $\widehat{X} = \sum_{h<N'} (\widehat{G}'_{22})^h \widehat{G}'_{21}$, so that $\|\widehat{X}\| \le 6$. Since $(I - \widehat{G}'_{22})\widehat{X} = \widehat{G}'_{21} - (\widehat{G}'_{22})^{N'} \widehat{G}'_{21}$ and $(4/5)^{N'} < \varepsilon$,

$$\|(I - G'_{22})\widehat{X} - G'_{21}\| \le \varepsilon \|\widehat{X}\| + \varepsilon + (4/5)^{N'}(1+\varepsilon) \le 9\varepsilon.$$

As $\|(I - G'_{22})^{-1}\| \le 4$, this gives $\|\widehat{X} - X\| \le 36\varepsilon$, and therefore

$$\|\widehat{G}'_{11} + \widehat{G}'_{12}\widehat{X} - U^{(j+1)}(z)\| \le \varepsilon + 6\varepsilon + 36\varepsilon = 43\varepsilon.$$

The output of the algorithm is the required entry of $\widehat{G}'_{11} + \widehat{G}'_{12}\widehat{X}$, rounded to the 2^{-t-4} grid. Its error is at most $43\varepsilon + \sqrt{2}\,2^{-t-5} < 2^{-t}$, and it has $O(t+1)$ bits per component.

The entry of $\widehat{G}'_{11} + \widehat{G}'_{12}\widehat{X}$ is a sum, over paths through at most N' trailing coordinates, of products of at most $N'+1$ rounded numbers. These products are exact, so N' does not enter the precision at which entries of $U^{(j)}$ are needed. Rounding to the 2^{-t_1} grid needs entries of G' at precision $t_1 + 4$, hence entries of $U^{(j)}$ at precision $t + 3n + 17$. By Lemma 3.3, P_{L} and P_{R} need entries of $U^{(j)}$ at precision t_0 only. So computing an entry of $U^{(j+1)}(z)$ to precision t requires entries of $U^{(j)}$ at precision at most $\max\{t + 3n + 17, t_0\}$.

Let $\mathrm{sp}_j(t)$ be the space of the algorithm for stage j at precisions at most t. The input, the seeds or the row selected in Lemma 3.3, and the counters, partial products and partial sums of the nested sums take at most $q(n+m+t)$ bits, by Lemma 2.6(iii), for a polynomial q that does not depend on j. Since the entries of $U^{(j)}$ are computed one at a time,

$$\mathrm{sp}_{j+1}(t) \le \mathrm{sp}_j(\max\{t + 3n + 17, t_0\}) + q(n+m+t).$$

Since $j < J$, every precision that occurs is at most $t_{\max} = \max\{t, t_0\} + J(3n+17)$, and $\mathrm{sp}_j(t) \le \mathrm{sp}_0(t_{\max}) + J\,q(n+m+t_{\max})$. This is polynomial in $n+m+t$, because J and t_0 are polynomial in n and sp_0 is the space of the given entry algorithm.

For item (ii), Lemma 3.8 gives the stage j and the component of gate i, and the control string y contains $z_{<j}$, which determines $U^{(j)}$, P_{L}, P_{R} and G. A component that occurs as an adjoint has the conjugate transposes of the parameters given below. In P_{L} and P_{R} the parameters are NOT or the identity in the permutation, signs in the diagonal gate, and H or the identity in the Hadamard transform, all fixed by the selected seeds or row. In the Fourier transforms and in the reflections of Lemma 3.6 they are fixed by the sizes, with phases at rational multiples of π, and at dimension one the gate applies a phase, the entry of $U^{(j)}$ given by item (i). This leaves the circuits for $\widehat{\Lambda}_{\mathcal{E}}$ and $\widehat{\Lambda}_{\mathcal{E}'}$, whose parameters come from these matrices and their patterns, and the patterns are computable from the row index by Lemma 3.5.

In the column of a leading coordinate, the entries of $\Lambda_{\mathcal{E}}$ are 0 or 1. By Equation (3.8), an entry in the column of a trailing coordinate is a sum over at most $N 2^{nN}$ paths of products of at most N

conjugated entries of G_{21}, G_{22}, which have modulus at most one. Let t_2 be the least integer with $2N^2 2^{nN-t_2} \le 2^{-t_\Lambda-4}$. Rounding the entries of G_{21}, G_{22} to the 2^{-t_2} grid changes each path product by at most $N2^{-t_2}(1+2^{-t_2})^N \le 2N2^{-t_2}$, so the exact sum of the rounded products is within $2^{-t_\Lambda-4}$ of the entry, which is the precision that the rounding in Section 3.5 requires. The same bound holds for $\Lambda_{\mathcal{E}'}$, because its row (ζ, c) is the row $(-\zeta, c)$ of $\Lambda_{\mathcal{E}}$ for $G^\dagger$. These computations use item (i) at one precision, polynomial in $n+m$ and independent of t, so $\widehat{\Lambda}$ does not depend on t. Each of its entries is a Gaussian integer divided by $D_\Lambda = 2^{t_\Lambda}(1+\delta) = 2^{t_\Lambda} + 2^{n+w+4}$.

The rounding makes the prefix probabilities exact rational numbers. In Equation (3.10) every squared amplitude, including $1 - \|\widehat{\Lambda}|v\rangle\|^2$, is an integer divided by D_Λ^2. The prefix probability $\mu(y)$ is the sum of the squared amplitudes of the basis states that extend y, so $D_\Lambda^2 \mu(y)$ is an integer: the sum of $|D_\Lambda \widehat{\Lambda}_{uv}|^2$ over the rows u whose pattern contains v and for which $u1$ extends y, plus $D_\Lambda^2(1 - \|\widehat{\Lambda}|v\rangle\|^2)$ if 0^{w+1} extends y. By Lemma 2.6(iii) it can be computed exactly. Therefore the condition $\mu(y) > 0$ in Equation (3.9) is decided exactly, and each ratio there is a rational of polynomial bit length, whose square root can be computed to precision t by Lemma 2.6(ii). A nonzero amplitude a/D_Λ, with a a Gaussian integer, has the phase $a/|a|$, whose real and imaginary parts are $\pm\sqrt{(\operatorname{Re} a)^2/|a|^2}$ and $\pm\sqrt{(\operatorname{Im} a)^2/|a|^2}$, with the signs of $\operatorname{Re} a$ and $\operatorname{Im} a$. The amplitude at $|0^w\rangle|0\rangle$ is nonnegative and has phase one.

In Equation (3.11), for nonempty I_u, the prefix probability of y is the number of positions in I_u that extend y, divided by $|I_u|$, and the phases are one. We take $W_{\theta,u}$ to be the rotation of the form Equation (3.9) with the first column given in Lemma 3.6, and its entries are computed from $\sin\theta$ by Lemma 2.6(ii). □

4.2 The Language B

The circuit C_0 has polynomially many gates and exponentially many parameters, each a matrix in U(2). We approximate each parameter by short words over $\mathcal{G}$, and the language B contains the bits of these words. The determinant of a word over $\{H, T, T^\dagger\}$ is an eighth root of unity, so such a word approximates a parameter only up to a phase, and the phases of the parameters $W_{i,y}$ for different y are relative phases of the gate W_i. We therefore use two words, one on the target of the gate and one on an ancilla, the phase qubit. If $W = e^{i\gamma} W_1$ with $W_1 \in \mathrm{SU}(2)$, then

$$(W_1 \otimes \operatorname{diag}(e^{i\gamma}, e^{-i\gamma}))(|\psi\rangle|0\rangle) = (W|\psi\rangle)|0\rangle, \tag{4.1}$$

and both factors on the left have determinant one. By Fact 2.8, words over the set of the next lemma approximate them.

Lemma 4.2. *There is a finite inverse-closed set $\mathcal{S}$ of words over $\{H, T, T^\dagger\}$, each of determinant one, that generates a dense subgroup of* SU(2).

Proof. Let Γ_0 be the group generated by H and T, which is dense in U(2) modulo global phases [NC10, Section 4.5.3]. The determinant maps Γ_0 to the eighth roots of unity, so its kernel Γ has finite index in Γ_0, and Γ is finitely generated by Schreier's lemma [Sch27]. Let $\mathcal{S}$ consist of generators of Γ, written as words over $\{H, T, T^\dagger\}$, and their inverses. To show that Γ is dense in SU(2), let $W \in \mathrm{SU}(2)$, and let $W_k \in \Gamma_0$ and phases ϕ_k satisfy $\phi_k W_k \to W$. Taking determinants gives $\phi_k^{16} = \det(\phi_k W_k)^8 \to 1$, so $W_k^{16} \to W^{16}$, and $W_k^{16} \in \Gamma$. Thus the closure of Γ contains the sixteenth power of every element of SU(2), and every element of SU(2) is such a power. □

Lemma 4.3. *The NOT gate, the Toffoli gate and the uniformly controlled gate with a two-bit control that applies $I, H, T, T^\dagger$ for the strings $00, 01, 10, 11$ are implemented exactly over $\mathcal{G}$ with ancillas returned to zero.*

Proof. The matrix T^4 is $\mathrm{diag}(1,-1)$, and NOT is HT^4H. For bits u_1,u_2,u_3,

$$4u_1u_2u_3 = u_1+u_2+u_3-(u_1\oplus u_2)-(u_1\oplus u_3)-(u_2\oplus u_3)+(u_1\oplus u_2\oplus u_3).$$

Hence the gate with phase $e^{\mathrm{i}\pi u_1u_2u_3}$, which is T^4 with two controls, is obtained by computing each parity on an ancilla with CNOT gates, applying T or $T^\dagger$ according to the sign, and uncomputing the parity. Conjugating its target by H gives the Toffoli gate. A controlled T or $T^\dagger$ is T or $T^\dagger$ applied to an ancilla that holds the conjunction of control and target, computed and uncomputed by a Toffoli gate. Let $W=T^2HT$, which satisfies $W\,\mathrm{NOT}\,W^\dagger=H$. Then $W^\dagger$ on the target, followed by a CNOT and by W on the target, is the controlled H. Finally, for each control string, NOT and Toffoli gates compute on an ancilla whether the control has this value, the corresponding unitary is applied with this ancilla as control, and the ancilla is uncomputed. □

Lemma 4.4. *Let $\eta=2^{-m}/(64(S+1))$. There are a language $B\in\mathsf{PSPACE}^A$ and a polynomial-time algorithm, which does not depend on A or U, that on input $(1^n,1^m,i)$ outputs a unitary circuit $\widehat{W}_i$ over $\mathcal{G}$ such that $\widehat{W}_i^B$ implements W_i on the zero-ancilla subspace with error η.*

Proof. Fix the set $\mathcal{S}$ of Lemma 4.2, the polynomial $p_{\mathcal{S}}$ of Fact 2.8 and the largest number $\ell_{\mathcal{S}}$ of letters of an element of $\mathcal{S}$, and let $\ell=\ell_{\mathcal{S}}\,p_{\mathcal{S}}(\lceil\log(16/\eta)\rceil)$. Fix i,y and let $W=W_{i,y}$. For a pair (W',W'') of words of length ℓ over $\{I,H,T,T^\dagger\}$, let $\mathrm{dist}(W',W'')=\|(W'\otimes W'')\iota_1-W\otimes|0\rangle\|_F^2$. From the entries of W and of the two words at a precision t' with $2^{8-t'}\le\eta^2/64$, we compute a number $\widehat{\mathrm{dist}}(W',W'')$ within $\eta^2/64$ of $\mathrm{dist}(W',W'')$. The entries of W are given by Lemma 4.1(ii), and those of the words by Lemma 2.6(v), whose hypothesis holds because the letters are unitary.

By Fact 2.8, the factors in Equation (4.1) are within $\eta/16$ of words W_0',W_0'' over $\mathcal{S}$ with at most $p_{\mathcal{S}}(\lceil\log(16/\eta)\rceil)$ elements, which become words of length ℓ after padding with I. Then $\|(W_0'\otimes W_0'')\iota_1-W\otimes|0\rangle\|\le\eta/8$, so $\mathrm{dist}(W_0',W_0'')\le 2(\eta/8)^2$ and $\widehat{\mathrm{dist}}(W_0',W_0'')\le\eta^2/32+\eta^2/64<\eta^2/4$. Let (W',W'') be the lexicographically first pair with $\widehat{\mathrm{dist}}(W',W'')\le\eta^2/4$, the selected pair for (i,y). Then $\mathrm{dist}(W',W'')\le\eta^2/4+\eta^2/64<\eta^2$, and hence $\|(W'\otimes W'')\iota_1-W\otimes|0\rangle\|<\eta$.

A query string of B consists of 1^n0, 1^m0, the index i, the control string y padded with zeros to the width w', a bit that selects one of the two words, the position of a letter, and a bit that selects one of the two bits of the index of this letter, where $00,01,10,11$ index $I,H,T,T^\dagger$. The fields have fixed lengths for given (n,m). The string is in B if the bit that these fields specify in the selected pair for (i,y) is one, and malformed strings are not in B. The gate list of Lemma 3.8 is computable from (n,m), and by Lemma 2.6(iv) the selected pair can be computed in space polynomial in $n+m$. The string contains 1^n and 1^m, so this space is polynomial in the length of the string. Thus $B\in\mathsf{PSPACE}^A$.

For a position and a word, let Q be the circuit that, for each of the two values of the last bit of the query string, writes this string on a register of ancillas, by NOT gates and by CNOT gates from the controls of gate i, applies a gate O_B with its answer on one of two further ancillas, and repeats the NOT and CNOT gates, which clears the register. Thus Q writes the index of the letter on the two further ancillas. The circuit for this position and word is Q, followed by the gate of Lemma 4.3 controlled on these ancillas and acting on the target of gate i or on the phase qubit, followed by Q^{-1}. Let $\widehat{W}_i$ be the product of these circuits over all positions and both words. On a control string y with the ancillas at zero, $\widehat{W}_i^B$ applies the selected pair to the target and the phase qubit and returns the other ancillas to zero. Hence $\widehat{W}_i^B\iota-\iota W_i$ is a direct sum over y of matrices of norm below η. The circuit is computed from (n,m,i) through S, w', η and ℓ, without evaluating B. □

Proof of Theorem 3.1. Let B be the language of Lemma 4.4. Write C_0 with its exact parameters as a product $\Gamma_h\cdots\Gamma_1$ of gates, and let $C_{n,m}$ be the product $\widehat{\Gamma}_h\cdots\widehat{\Gamma}_1$, with $\widehat{\Gamma}_k=\widehat{W}_i$ if $\Gamma_k=W_i$ and

$\widehat{\Gamma}_k = \Gamma_k$ otherwise. If ι sets the ancillas of Lemma 4.4 to zero, then

$$C^B_{n,m}\iota - \iota C_0 = \sum_{k=1}^{h} \widehat{\Gamma}^B_h \cdots \widehat{\Gamma}^B_{k+1}(\widehat{\Gamma}^B_k \iota - \iota\Gamma_k)\Gamma_{k-1}\cdots\Gamma_1,$$

which has norm at most $S\eta$ by Lemma 4.4. With Equation (3.13), $C^B_{n,m}$ implements U_n on the zero-ancilla subspace with error less than $S\eta + 2^{-m}/64 < 2^{-m}/32$, and Lemma 2.7 gives an extension $\widetilde{U}$ of U_n with $\|C^B_{n,m} - \widetilde{U}\| < 3 \cdot 2^{-m}/32 < 2^{-m}$. The set $\mathcal{S}$ and the polynomial $p_{\mathcal{S}}$ do not depend on A or U, and neither do the algorithms of Lemmas 3.8 and 4.4, so the algorithm that produces $C_{n,m}$ is the same for every A and U. □

4.3 The Main Theorem

Theorem 3.1 and Fact 2.5 give the clean case of Theorem 1.1 directly. For the projective classes we reduce to the clean ones through $U_n \otimes U_n^\dagger$, a construction of Kretschmer and Tang [KT26, Theorems 5.20 and 5.23]. The garbage state of a dilation does not depend on the input, so a circuit for a dilation of U_n on one data register, followed by its adjoint on a second data register with the same ancillas, implements $U_n \otimes U_n^\dagger$ with the ancillas returned to zero. Let $U \otimes U^\dagger$ be the unitary language whose member of length $2n$ is $U_n \otimes U_n^\dagger$ and whose members of odd length are identities.

Lemma 4.5. *Let L be a classical oracle. If $U \in \mathsf{projUnitaryP}^L$, then $U \otimes U^\dagger \in \mathsf{unitaryP}^L$, and if $U \in \mathsf{projUnitaryPSPACE}^L$, then $U \otimes U^\dagger \in \mathsf{unitaryPSPACE}^L$.*

Proof. Let $C = C_{n,m+1}$ be a circuit of the given family on $n + a$ qubits, and let $\widetilde{U}$ be a dilation of U_n with garbage state $|g\rangle$ and $\|C^L - \widetilde{U}\| \le 2^{-m-1}$. Let C' act on n-qubit registers X, Y and the ancilla register W as C on XW followed by $C^\dagger$ on YW, and let $\widetilde{W} = \widetilde{U}^\dagger_{\mathsf{YW}}\widetilde{U}_{\mathsf{XW}}$. Then $\|C'^L - \widetilde{W}\| \le 2\|C^L - \widetilde{U}\| \le 2^{-m}$. Since $\widetilde{U}(|\chi\rangle|0^a\rangle) = (U_n|\chi\rangle)|g\rangle$ for every $|\chi\rangle$, also $\widetilde{U}^\dagger(|\chi\rangle|g\rangle) = (U_n^\dagger|\chi\rangle)|0^a\rangle$. Both identities hold in the presence of further registers, so for $|\Psi\rangle$ on XY,

$$\widetilde{W}(|\Psi\rangle|0^a\rangle) = \widetilde{U}^\dagger_{\mathsf{YW}}(((U_n \otimes I)|\Psi\rangle)|g\rangle) = ((U_n \otimes U_n^\dagger)|\Psi\rangle)|0^a\rangle.$$

Thus $\widetilde{W}$ is an extension of $U_n \otimes U_n^\dagger$. The circuits C', with empty circuits at odd lengths, form a uniform family: the adjoint of a gate list is the reversed list of adjoint gates, with $O_L^\dagger = O_L$, and in polynomial space the h-th gate from the end can be found by running the generator with a counter. □

Lemma 4.6. *Let L be a classical oracle. If $U \otimes U^\dagger \in \mathsf{unitaryP}^L$, then $U \in \mathsf{projUnitaryP}^L$, and if $U \otimes U^\dagger \in \mathsf{unitaryPSPACE}^L$, then $U \in \mathsf{projUnitaryPSPACE}^L$.*

Proof. Let $\widetilde{W}$ be an extension of $U_n \otimes U_n^\dagger$ on $2n + a$ qubits. For every $|\psi\rangle$ on the first n qubits,

$$\widetilde{W}(|\psi\rangle|0^n\rangle|0^a\rangle) = (U_n|\psi\rangle)(U_n^\dagger|0^n\rangle)|0^a\rangle,$$

so $\widetilde{W}$ is a dilation of U_n on $n + a$ ancillas, with garbage state $(U_n^\dagger|0^n\rangle)|0^a\rangle$. Hence the circuits of the given family at length $2n$, with the last $n + a$ qubits as ancillas, form a family for U. □

Proof of Theorem 1.1. For the clean classes, if $U \in \mathsf{unitaryPSPACE}^A$, then Fact 2.5 and Theorem 3.1 give $B \in \mathsf{PSPACE}^A$ with $U \in \mathsf{unitaryP}^B$. Conversely, let $L \in \mathsf{PSPACE}^A$ and $U \in \mathsf{unitaryP}^L$. A time-uniform family is space-uniform, so $U \in \mathsf{unitaryPSPACE}^L$, and by Fact 2.5 it has entries computable in polynomial space relative to L. By Lemma 2.6(i) these entries are computable in polynomial space relative to A, so $U \in \mathsf{unitaryPSPACE}^A$ by Fact 2.5.

For the projective classes, the first and last equivalences of

$$\begin{aligned} U \in \mathsf{projUnitaryPSPACE}^A &\iff U \otimes U^\dagger \in \mathsf{unitaryPSPACE}^A \\ &\iff U \otimes U^\dagger \in \mathsf{unitaryP}^L \text{ for some } L \in \mathsf{PSPACE}^A \\ &\iff U \in \mathsf{projUnitaryP}^L \text{ for some } L \in \mathsf{PSPACE}^A \end{aligned}$$

are Lemmas 4.5 and 4.6, and the middle one is the clean case.

Now let K be PSPACE^A-complete, let $L \in \mathsf{PSPACE}^A$, and let f be a reduction from L to K, computed without access to A, with $|f(x)| \le p(|x|)$. Let O_L be a gate on ℓ query qubits and an answer qubit. There is a circuit R of Toffoli, CNOT and NOT gates [Ben73; Tof80], computable in polynomial time without an oracle, that computes on ancillas the string $f(x)$, padded to $p(\ell)$ wires, and the bits $\mathbf{1}[|f(x)| = \ell']$ for $\ell' \le p(\ell)$. For each ℓ', let $Q_{\ell'}$ be the query to O_K on the first ℓ' wires of $f(x)$, with the answer qubit of O_L as its answer, controlled on the bit $\mathbf{1}[|f(x)| = \ell']$ as in Section 2.2. On a basis state $|x, b\rangle$ with the ancillas at zero, only $Q_{|f(x)|}$ acts, and it adds $K(f(x)) = L(x)$ to b. Hence $R^{-1}(\prod_{\ell'} Q_{\ell'})R$ acts as O_L on the zero-ancilla subspace.

Let C be a circuit of a $\mathsf{unitaryP}^L$ family at precision $m+2$, and let C' be C with each O_L replaced by $R^{-1}(\prod_{\ell'} Q_{\ell'})R$. Then C'^K implements U_n on the zero-ancilla subspace with error 2^{-m-2}, so Lemma 2.7 gives an extension at distance $3 \cdot 2^{-m-2} < 2^{-m}$. For a family of a projective class, the last statement of Lemma 2.7 gives a dilation at distance $3 \cdot 2^{-m-2}$. Hence $\mathsf{unitaryP}^L \subseteq \mathsf{unitaryP}^K$ and $\mathsf{projUnitaryP}^L \subseteq \mathsf{projUnitaryP}^K$. Since $K \in \mathsf{PSPACE}^A$, both unions are attained at K. □

Proof of Corollary 1.2. Theorem 3.1 with $A = A_U$ and the entry algorithm dec^{A_U} of Section 2.3 gives $B_U \in \mathsf{PSPACE}^{A_U}$ with $U \in \mathsf{unitaryP}^{B_U}$, and the algorithm that produces the circuits does not depend on U. In particular every unitary language lies in $\mathsf{unitaryP}^{\mathsf{ALL}}$, and the reverse inclusion holds by definition. □

5 Oracle Characterization and Cryptography

By Theorem 1.1, any PSPACE^A-complete language suffices as an oracle for all of $\mathsf{unitaryPSPACE}^A$. In fact a language L suffices exactly when $\mathsf{PSPACE}^A \subseteq \mathsf{BQP}^L$. The oracle gate of a language in PSPACE^A is itself a polynomial-space unitary, so a circuit that implements it decides the language. Conversely, an oracle gate for a language in BQP^L can be replaced inside a unitary circuit by a coherent run of its decider.

By synthesizing a measurement, we also show that a polynomial-time algorithm with a PSPACE oracle distinguishes any two efficiently generated states at inverse-polynomial trace distance, from one copy and with inverse-polynomial advantage. If BQP = PSPACE this oracle can be removed, so EFI pairs imply BQP ≠ PSPACE (Section 5.2). In the Oracle State Distinguishing Game of Lombardi, Ma and Wright [LMW24] the oracle may depend on the states, and a synthesized unitary wins the game (Section 5.3).

5.1 The Oracle Characterization

To replace an oracle gate O_B for a language $B \in \mathsf{BQP}^L$, we need a unitary circuit that acts correctly on superpositions of queries and returns its ancillas to zero. The compute-copy-uncompute construction [BBBV97, Theorem 4.14 and Corollary 4.15] does this, and its error in operator norm is exponentially small, because the majority of many runs of the decider is wrong with exponentially small probability.

Lemma 5.1. *Let L be a classical oracle and $B \in \mathsf{BQP}^L$. A polynomial-time algorithm with no oracle maps $(1^\ell, 1^m)$, for $\ell, m \geq 1$, to a unitary circuit $C^{\mathrm{sim}}_{\ell,m}$ over $\mathcal{G}$ with oracle gates O_L, on ℓ query qubits, one answer qubit and ancillas, that implements the oracle gate O_B on the zero-ancilla subspace with error 2^{-m}.*

Proof. Let $(\mathcal{C}_\ell)_\ell$ be a family for B as in the definition of BQP^L. A measurement is a CNOT onto a new ancilla that is never used again, followed by discarding this ancilla. Hence, using ancillas for new qubits and keeping all discarded qubits, we get a unitary circuit C_ℓ over $\mathcal{G}$ with oracle gates O_L such that measuring its output wire on $|x\rangle$ and zero ancillas gives $B(x)$ with probability at least $2/3$, for $x \in \{0,1\}^\ell$.

Let C be the circuit that copies the query register to $k = 36(m+1)$ further registers by CNOT gates, runs C_ℓ on each of them with its own ancillas, and computes the majority of the k output wires on a further ancilla by Toffoli, CNOT and NOT gates [Ben73; Tof80]. Let $C^{\mathrm{sim}}_{\ell,m} = C^{-1}\,\mathrm{CNOT}\,C$, where the CNOT goes from this ancilla to the answer qubit.

The circuit C uses the query register only as a control, so C maps $|x\rangle$ and zero ancillas to $|x\rangle|\varphi_x\rangle$. Write $|\varphi_x\rangle = |\varphi_x^+\rangle + |\varphi_x^-\rangle$, where the majority ancilla equals $B(x)$ in the first term and $1 - B(x)$ in the second. The k runs are independent and each is wrong with probability at most $1/3$. A wrong majority requires at least $k/2$ wrong runs, so by Hoeffding's inequality

$$\||\varphi_x^-\rangle\|^2 \leq e^{-2(k/6)^2/k} = e^{-2(m+1)} < 2^{-2m-2}.$$

For a unit vector $|\beta\rangle$ on the answer qubit, the two terms give

$$\mathrm{CNOT}(|\beta\rangle|\varphi_x\rangle) - (\mathrm{NOT}^{B(x)}|\beta\rangle)|\varphi_x\rangle = ((\mathrm{NOT}^{1-B(x)} - \mathrm{NOT}^{B(x)})|\beta\rangle)|\varphi_x^-\rangle,$$

which has norm at most $2\||\varphi_x^-\rangle\| < 2^{-m}$. Applying C^{-1} to both vectors on the left shows that $C^{\mathrm{sim}}_{\ell,m}$ on $|x\rangle|\beta\rangle$ and zero ancillas is within 2^{-m} of $O_B|x\rangle|\beta\rangle$ with zero ancillas. The circuit $C^{\mathrm{sim}}_{\ell,m}$ also uses the query register only as a control, so the difference between $C^{\mathrm{sim}}_{\ell,m}\iota_a$ and $\iota_a O_B$ is a direct sum over x of matrices of norm below 2^{-m}. □

Corollary 5.2. *Let L be a classical oracle and $B \in \mathsf{BQP}^L$. Then $\mathsf{unitaryP}^B \subseteq \mathsf{unitaryP}^L$ and $\mathsf{projUnitaryP}^B \subseteq \mathsf{projUnitaryP}^L$. If the circuits of the given family are over $\mathcal{G}$, then so are those of the new family.*

Proof. Let C be a circuit of the given family at precision $m+3$, with Q oracle gates, let $\widetilde{U}$ be an extension or a dilation of U_n with $\|C^B - \widetilde{U}\| \leq 2^{-m-3}$, and let $m' = m + 3 + \lceil \log(Q+1) \rceil$. Let C' be C with each oracle gate on $\ell \geq 1$ query qubits replaced by $C^{\mathrm{sim}}_{\ell,m'}$ with its own ancillas, and each gate with no query qubits replaced by NOT or the identity, and let ι set the new ancillas to zero. The telescoping sum in the proof of Theorem 3.1 gives $\|C'\iota - \iota C^B\| \leq Q 2^{-m'} \leq 2^{-m-3}$, so $\|C'\iota - \iota\widetilde{U}\| \leq 2^{-m-2}$. On the old ancillas at zero, $\widetilde{U}$ acts as U_n and leaves them at zero, or at $|g\rangle$ for a dilation. So Lemma 2.7 gives an extension or a dilation of U_n at distance $3 \cdot 2^{-m-2} < 2^{-m}$ from C'. The circuit C' is computed in polynomial time with no oracle. □

For all classical oracles A and L, Corollary 1.3 states that

$$\begin{aligned} \mathsf{unitaryPSPACE}^A \subseteq \mathsf{unitaryP}^L &\iff \mathsf{PSPACE}^A \subseteq \mathsf{BQP}^L \\ &\iff \mathsf{projUnitaryPSPACE}^A \subseteq \mathsf{projUnitaryP}^L, \end{aligned}$$

and that $\mathsf{unitaryP}^A = \mathsf{unitaryPSPACE}^A$ if and only if $\mathsf{BQP}^A = \mathsf{PSPACE}^A$, and the same for the projective classes.

Proof of Corollary 1.3. Suppose that $\mathsf{PSPACE}^A \subseteq \mathsf{BQP}^L$. A language in $\mathsf{unitaryPSPACE}^A$ lies, by Theorem 1.1, in $\mathsf{unitaryP}^B$ for some $B \in \mathsf{PSPACE}^A \subseteq \mathsf{BQP}^L$, and hence in $\mathsf{unitaryP}^L$ by Corollary 5.2. The same argument applies to the projective classes.

Conversely, suppose that one of the two inclusions of unitary classes holds, and let $L' \in \mathsf{PSPACE}^A$. The oracle gates $O_{L'}$ form a unitary language, whose member of length n is the gate on $n-1$ query qubits and one answer qubit. Its entries are 0 or 1 and are computable in polynomial space relative to A, so it lies in $\mathsf{unitaryPSPACE}^A$ by Fact 2.5. Since the clean classes lie in the projective ones, either inclusion puts it in $\mathsf{projUnitaryP}^L$. For $x \in \{0,1\}^\ell$, let C be the circuit of the family at $(1^{\ell+1}, 1^3)$. Then C^L is within $1/8$ of a dilation of $O_{L'}$ with garbage state $|g\rangle$, so $C^L|x,0\rangle|0^a\rangle$ is within $1/8$ of $|x, L'(x)\rangle|g\rangle$.

The circuit C may contain encoded gates, while a decider for BQP^L is over $\mathcal{G}$. An encoded gate on k wires has a description of length at least 4^k, so its decomposition into one-qubit gates and CNOT gates [NC10, Section 4.5] has polynomial size, and the effective Solovay–Kitaev theorem [KT26, Section 2.2] compiles these gates in polynomial time. This gives a circuit over $\mathcal{G}$ with oracle gates O_L that is within $1/8$ of C^L up to a global phase. Its output vector on $|x,0\rangle|0^a\rangle$ is therefore within $1/4$ of $|x, L'(x)\rangle|g\rangle$ up to a phase, so measuring the answer qubit gives $L'(x)$ except with probability at most $1/16$. Hence $L' \in \mathsf{BQP}^L$.

For the last statement, take $L = A$. A time-uniform family is space-uniform, so $\mathsf{unitaryP}^A \subseteq \mathsf{unitaryPSPACE}^A$ and $\mathsf{projUnitaryP}^A \subseteq \mathsf{projUnitaryPSPACE}^A$. Moreover $\mathsf{BQP}^A \subseteq \mathsf{PSPACE}^A$, because by Lemma 2.6(vii) the probability that a decider outputs one can be computed to precision 3. □

Kretschmer and Tang ask whether polynomial-time implementations can always erase garbage, that is, whether every $U \in \mathsf{projUnitaryP}$ equals some $U' \in \mathsf{unitaryP}$ up to global phases [KT26, Question 3.9], and they show that the answer is negative relative to a unitary oracle [KT26, Theorem 6.23]. Relative to a classical oracle A with $\mathsf{BQP}^A = \mathsf{PSPACE}^A$ the answer is positive: if $U \in \mathsf{projUnitaryP}^A \subseteq \mathsf{projUnitaryPSPACE}^A$, then $U_n = e^{\mathrm{i}\gamma_n} U'_n$ for some phases $e^{\mathrm{i}\gamma_n}$ and some $U' \in \mathsf{unitaryPSPACE}^A$ [KT26, Theorem 6.19], and $U' \in \mathsf{unitaryP}^A$ by Corollary 1.3.

5.2 Distinguishing States with a PSPACE Oracle

An EFI pair [BCQ23] is a pair of mixed states that can be generated efficiently and are statistically far apart, yet computationally indistinguishable. Following Kretschmer and Tang [KT26, Definition 7.13], an EFI pair relative to a classical oracle A consists of two time-uniform families of general quantum circuits with encoded gates and oracle gates O_A. Their output states $\rho_{0,n}, \rho_{1,n}$ on $w(n)$ qubits satisfy $D(\rho_{0,n}, \rho_{1,n}) \geq 1/p(n)$ for a positive polynomial p and all large n. For every nonuniform polynomial-size family $(\mathcal{D}_n)_n$ of such circuits with one output qubit, however, the advantage $|\Pr[\mathcal{D}_n(\rho_{0,n}) = 1] - \Pr[\mathcal{D}_n(\rho_{1,n}) = 1]|$ is negligible, that is, eventually smaller than n^{-c} for every constant c.

Brakerski, Canetti and Qian ask whether EFI pairs separate BQP from PSPACE, and they suggest synthesizing the Helstrom measurement with a PSPACE oracle [BCQ23, Section 1.2]. For two states with difference Δ, the Helstrom measurement [Hel76] accepts on the positive eigenspace of Δ and has advantage equal to the trace distance. The projector onto this eigenspace depends discontinuously on Δ, and we can compute Δ only approximately. Kretschmer and Tang avoid this with phase estimation of $e^{\mathrm{i}\Delta}$, which shows that EFI pairs relative to A separate $\mathsf{unitaryP}^A$ from $\mathsf{unitaryPSPACE}^A$ [KT26, Theorem 7.14]. We instead replace this projector by a smooth function of Δ, the POVM element for outcome zero of one Hadamard test.

Fix a classical oracle A and an integer-valued polynomial $p \geq 1$, and let $\rho_{0,n}, \rho_{1,n}$ be states on $w = w(n)$ qubits generated by two families of circuits as above. Let $\alpha = 1/p(n)$ and $\Delta = \rho_{0,n} - \rho_{1,n}$,

and let τ be the largest power of two with $2^{w-1}\tau \le \alpha/4$. Since Δ is Hermitian, the matrix

$$W_n = (\Delta + \mathrm{i}\tau I)(\Delta^2 + \tau^2 I)^{-1/2}$$

is unitary. Its Hermitian part $\Delta(\Delta^2 + \tau^2 I)^{-1/2}$ is close to the sign of Δ on eigenvalues of modulus much larger than τ. The Hadamard test of W_n is the unitary

$$(H \otimes I)(|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes W_n)(H \otimes I)$$

on a test qubit and a data register of w qubits. A unitary language has one member of each length, so we fix an integer c with $(n+1)^c \ge w+1$ for all $n \ge 1$, and let U_Δ be the unitary language whose member of length $(n+1)^c$ is this unitary tensored with the identity on the remaining qubits, and whose other members are identities.

Lemma 5.3. *With the notation above, the following hold.*

(i) Suppose that $D(\rho_{0,n}, \rho_{1,n}) \ge \alpha$, and let C be a unitary circuit that implements the member of U_Δ of length $(n+1)^c$ on the zero-ancilla subspace with error at most $\alpha/8$. The circuit that applies C to its input on the data register with all other wires at zero, and outputs the test qubit, has advantage at least $\alpha/4$ between $\rho_{0,n}$ and $\rho_{1,n}$.

(ii) The language U_Δ has entries computable in polynomial space relative to A.

(iii) There are a language $B \in \mathsf{PSPACE}^A$ and a time-uniform family $(\mathcal{D}_n)_n$ of general quantum circuits over $\mathcal{G}$ with oracle gates O_B such that $\mathcal{D}_n$ has advantage at least $\alpha/4$ between $\rho_{0,n}$ and $\rho_{1,n}$ whenever $D(\rho_{0,n}, \rho_{1,n}) \ge \alpha$.

Proof. For item (i), the $(0,0)$ block of the Hadamard test on the test qubit is $(I + W_n)/2$, so the test qubit is zero with probability $\mathrm{Tr}(E\rho)$ on an input state ρ, where

$$E = \Big(\frac{I + W_n}{2}\Big)^\dagger \frac{I + W_n}{2} = \frac{1}{2}\Big(I + \Delta(\Delta^2 + \tau^2 I)^{-1/2}\Big).$$

Since $\mathrm{Tr}\,\Delta = 0$, summing over the eigenvalues λ of Δ with multiplicity gives

$$\mathrm{Tr}(E\Delta) = \frac{1}{2}\sum_\lambda \frac{\lambda^2}{\sqrt{\lambda^2 + \tau^2}} \ge \frac{1}{2}\sum_\lambda (|\lambda| - \tau) = \frac{1}{2}\|\Delta\|_1 - 2^{w-1}\tau \ge \frac{3}{4}\alpha,$$

where the first inequality uses $\sqrt{\lambda^2 + \tau^2} \le |\lambda| + \tau$ and $\lambda^2 \ge (|\lambda| - \tau)(|\lambda| + \tau)$. Replacing the Hadamard test by C moves the output vector on a pure input by at most $\alpha/8$, so it changes each probability by at most $\alpha/4$ on pure inputs (Section 2.4) and, by convexity, on mixed inputs. This leaves an advantage of at least $\alpha/4$.

For item (ii), all computations below use space polynomial in $n+t$ with oracle A. A time-uniform family is space-uniform, so by Lemma 2.6(vii), for circuits with no input, we can compute the entries of $\rho_{0,n}$ and $\rho_{1,n}$, and hence those of Δ. Let $X = (1-\tau^2)I - \Delta^2$, so that $I - X = \Delta^2 + \tau^2 I$. Since $0 \preceq \Delta^2 \preceq I$ and $\tau^2 \le 1/4$, we have $\|X\| \le 1 - \tau^2$, and the binomial series of $(1-z)^{-1/2}$ gives

$$W_n = \sum_{j \ge 0} c_j (\Delta + \mathrm{i}\tau I) X^j, \qquad c_j = \prod_{r=1}^{j} \frac{2r-1}{2r} \le 1.$$

To compute an entry to precision t, let $M = \tau^{-2}(t + 3 + 2\log(1/\tau))$. Since $\|\Delta + \mathrm{i}\tau I\| \le 2$, the terms with $j > M$ have total norm at most $2(1-\tau^2)^{M+1}/\tau^2 \le 2\tau^{-2}e^{-M\tau^2} \le 2^{-t-2}$. Term j is twice the

product of the contractions $(\Delta + \mathrm{i}\tau I)/2$ and $\frac{2r-1}{2r}X$ for $r \le j$, so its entries can be computed by Lemma 2.6(v). The space used depends on j only through $\log(j+1)$, which is polynomial in $n+t$ for $j \le M$. We compute the entries of the terms with $j \le M$ to precision $t + \lceil \log(M+1) \rceil + 3$, so that their errors add up to at most 2^{-t-3}, sum them exactly by Lemma 2.6(iii), and round to $t+3$ fractional bits, which adds less than 2^{-t-3}. The total error is below 2^{-t}.

The blocks of the Hadamard test are $(I \pm W_n)/2$, so its entries follow, and at lengths not of the form $(n+1)^c$ the entries are those of the identity.

For item (iii), Theorem 3.1 applied to U_Δ gives $B \in \mathsf{PSPACE}^A$ and circuits over $\mathcal{G}$. The circuit at length $(n+1)^c$ and precision $\lceil \log(8p(n)) \rceil$ implements the member on the zero-ancilla subspace with error at most $\alpha/8$, so item (i) gives the family $(\mathcal{D}_n)_n$. □

Corollary 1.4 states that if EFI pairs exist relative to a classical oracle A, then $\mathsf{BQP}^A \neq \mathsf{PSPACE}^A$, and in particular that EFI pairs imply $\mathsf{BQP} \neq \mathsf{PSPACE}$.

Proof of Corollary 1.4. Suppose that $\mathsf{BQP}^A = \mathsf{PSPACE}^A$ and that an EFI pair relative to A exists. Enlarging p, we may assume that it is integer-valued with $p \ge 1$. By Lemma 5.3(ii) and Theorem 3.1, $U_\Delta \in \mathsf{unitaryP}^B$ for some $B \in \mathsf{PSPACE}^A = \mathsf{BQP}^A$, and then $U_\Delta \in \mathsf{unitaryP}^A$ by Corollary 5.2, in both cases with circuits over $\mathcal{G}$. These circuits at length $(n+1)^c$ and precision $\lceil \log(8p(n)) \rceil$ give, by Lemma 5.3(i), a time-uniform family of polynomial size with advantage at least $1/(4p(n))$ for all large n. This contradicts the definition of an EFI pair. The second statement is the case $A = \emptyset$. □

For $A = \emptyset$ this distinguisher is a polynomial-time quantum algorithm over $\mathcal{G}$ with no advice. The corollary therefore also holds for the EFI pairs of Brakerski, Canetti and Qian, whose generators are polynomial-time quantum algorithms and whose distinguishers may have quantum advice [BCQ23]. They show that EFI pairs exist if and only if commitment schemes do [BCQ23, Theorem 13], building on [Yan22], so computationally hiding and binding quantum commitments also imply $\mathsf{BQP} \neq \mathsf{PSPACE}$.

The same argument breaks single-copy pseudorandom states that are longer than their key [KQST23, Definition 7]. These are pure states $|\psi_k\rangle$ on $q(n) > n$ qubits, for keys $k \in \{0,1\}^n$, generated in polynomial time, such that polynomial-time quantum adversaries cannot distinguish one copy of $|\psi_k\rangle$ for a random key from one copy of a Haar-random state. The average ρ_n of $|\psi_k\rangle\langle\psi_k|$ over the keys has rank at most 2^n, and the average of a Haar-random state is $I/2^{q(n)}$. With Π_{ρ_n} the projector onto the support of ρ_n, their trace distance is at least $\mathrm{Tr}(\Pi_{\rho_n}(\rho_n - I/2^{q(n)})) \ge 1 - 2^{n-q(n)} \ge 1/2$. Both are generated by time-uniform families, so Lemma 5.3(iii) with $p = 2$ gives a uniform distinguisher with a PSPACE oracle and advantage at least $1/8$. Lombardi, Ma and Wright ask whether a PSPACE oracle breaks every single-copy pseudorandom state [LMW24, Section 1.1], and for states longer than their key it does. Under $\mathsf{BQP} = \mathsf{PSPACE}$, Corollary 5.2 removes the oracle from this distinguisher, so the existence of these states implies $\mathsf{BQP} \neq \mathsf{PSPACE}$, and hence $\mathsf{P} \neq \mathsf{PSPACE}$, as asked in [KQST23, Section 1.4].

5.3 A Polynomial-Query Adversary for the Oracle State Distinguishing Game

Lombardi, Ma and Wright study the Unitary Synthesis Problem through the following game [LMW24, Definition 1.3]. For $f\colon \{0,1\}^\ell \to \{0,1\}$, write $Z_f|u\rangle = (-1)^{f(u)}|u\rangle$ for its phase unitary and $|\varphi_f\rangle = 2^{-\ell/2}\sum_u (-1)^{f(u)}|u\rangle$ for its binary phase state. A uniformly random family $R = (R_k)_k$ of Boolean functions on $\{0,1\}^n$, indexed by keys, is drawn. A challenger sends the adversary $|\varphi_{R_k}\rangle$ for a uniformly random key k, or $|\varphi_f\rangle$ for a uniformly random function f, each with probability $1/2$, and the adversary must output which of the two it received. The adversary is an oracle circuit,

fixed before R, whose queries apply the phase unitary of a Boolean function that may depend on R arbitrarily.

The Strong Non-Synthesis Conjecture [LMW24, Conjecture 1.5] states that whenever the number of keys is superpolynomial in n, every polynomial-query adversary wins with probability at most $1/2 + \mathrm{negl}(n)$, and its authors note that it would imply a negative answer to the Unitary Synthesis Problem. Because the queried Boolean function may depend on R, it may encode a unitary built from R, so Corollary 1.2 supplies the implementation that the reduction of [LMW24, Section 3.3] needs.

Proposition 5.4. *For the key set $\{0,1\}^{n-1}$, there is a polynomial-query oracle circuit such that, for every R, there is a Boolean function f_R for which the circuit with queries to f_R wins the game with probability at least $11/16$. Hence the Strong Non-Synthesis Conjecture is false.*

Proof. Fix n and R. The states $|\varphi_{R_k}\rangle$ span a subspace $\mathcal{H}_R$ of dimension at most 2^{n-1}, so there is a unitary T_R on n qubits that maps $\mathcal{H}_R$ into the span of the basis states with first qubit zero. Applying T_R and outputting the first qubit gives zero with certainty on $|\varphi_{R_k}\rangle$. A uniformly random function gives the average state $I/2^n$, since $(-1)^{f(u)+f(u')}$ has expectation zero for $u \neq u'$, and T_R preserves this state, so the output is then zero with probability $1/2$. Applying T_R would therefore win the game with probability $3/4$.

To approximate T_R with queries, let B_R be the language of Corollary 1.2 for the unitary language with member T_R at length n and identities at other lengths, and let C be its circuit at $(1^n, 1^5)$. The algorithm that produces C does not depend on the unitary language, so C does not depend on R. The queries of an oracle circuit in [LMW24, Definition 3.4] have one length, so let ℓ exceed the query lengths of all oracle gates of C, and pad a query string x to $x' = x10^{\ell-|x|-1}$. Let $f_R(x', b) = b \cdot B_R(x)$ for a bit b, and let f_R be zero on all other strings of length $\ell+1$. Then $Z_{f_R}|x', b\rangle = (-1)^{b\,B_R(x)}|x', b\rangle$, so with the padding on ancillas, Z_{f_R} conjugated by H on the qubit of b acts as the gate O_{B_R}, as in [LMW24, Remark 3.6].

The adversary runs C on the challenge and zero ancillas with each oracle gate replaced in this way, and outputs the first qubit. The circuit C^{B_R} is within 2^{-5} of an extension of T_R, which changes each probability by at most $1/16$, so the adversary wins with probability at least $3/4 - 1/16 = 11/16$. The number 2^{n-1} of keys is superpolynomial in n, so this contradicts the conjecture. □

This adversary makes polynomially many sequential queries, whereas the lower bounds of [LMW24] concern one query and, through their Remark 3.6, polynomially many parallel queries.

6 The Channel Model

Bostanci, Efron, Metger, Poremba, Qian and Yuen pose their questions about the collapse of unitary classes and about interactive proofs for unitaries in the channel model of Section 2.5 [BEM+26]. To answer them we need the counterpart of Theorem 1.1 in this model, namely polynomial-time circuits with a PSPACE oracle for the problems in $\mathsf{unitaryPSPACE}^{\diamond}$.

Such circuits exist only for size-computable problems, because every problem in $\mathsf{unitaryBQP}^{\diamond,L}$, for any language L, is size-computable: its generator runs in polynomial time with no oracle and writes the input and output wires of its circuit at $(x, 1^1)$. The register sizes of a problem in $\mathsf{unitaryPSPACE}^{\diamond}$, however, can encode a language in PSPACE.

Example 6.1. Let $L' \in \mathsf{PSPACE}$, and let $U_x|\psi\rangle = |\psi\rangle|0^{L'(x)}\rangle$ from one qubit to $1 + L'(x)$ qubits. This problem lies in $\mathsf{unitaryPSPACE}^{\diamond}$ with error zero: a generator decides $L'(x)$ in polynomial space and writes the circuit that adds $L'(x)$ new qubits. It is size-computable only if $L' \in \mathsf{P}$.

6.1 Implementing Channels with a PSPACE Oracle

Theorem 6.2. *Let $U \in \mathsf{unitaryPSPACE}^{\diamond}$ be size-computable. There is a language $B \in \mathsf{PSPACE}$ with $U \in \mathsf{unitaryBQP}^{\diamond,B}$.*

A circuit of this model implements a channel, while Theorem 3.1 concerns unitaries. A space-uniform circuit may discard exponentially many qubits, so keeping them does not give a unitary on polynomially many qubits. The channel, however, acts on polynomially many qubits, and by Lemma 2.6(vii) its matrix representation can be computed in polynomial space. So we write down a Stinespring isometry of the channel in terms of this matrix, with an environment of polynomially many qubits, and embed it in a unitary whose entries are computable in polynomial space. Theorem 3.1 then applies to this unitary.

Let $(\mathcal{C}_{x,r})_{x,r}$ be a space-uniform family of general quantum circuits of space polynomial in $|x|+r$, with input register A_x of $n_{\text{in}} = n_{\text{in}}(x)$ qubits and output register B_x of $n_{\text{out}} = n_{\text{out}}(x)$ qubits. Fix x and r, and write $\mathcal{C} = \mathcal{C}_{x,r}$ and $d_{\text{in}} = 2^{n_{\text{in}}}$. A Stinespring isometry of a channel $\mathcal{N}$ is an isometry Y into its output register and an environment E with $\mathcal{N}(\rho) = \operatorname{Tr}_{\mathsf{E}}(Y\rho Y^\dagger)$. To obtain one for $\mathcal{C}$ we use its Choi matrix [Cho75]

$$J_{\mathcal{C}} = \sum_{u,u'} |u\rangle\langle u'| \otimes \mathcal{C}(|u\rangle\langle u'|).$$

Its entries are $\langle u, v|J_{\mathcal{C}}|u', v'\rangle = \langle v|\mathcal{C}(|u\rangle\langle u'|)|v'\rangle$, and it is positive semidefinite, because it is $\mathrm{id} \otimes \mathcal{C}$ applied to the positive semidefinite matrix $\sum_{u,u'} |u, u\rangle\langle u', u'|$.

Lemma 6.3. *Let E be a register of $n_{\text{in}} + n_{\text{out}}$ qubits, with basis indices i as for $\mathsf{A}_x\mathsf{B}_x$. The map $Y\colon \mathsf{A}_x \to \mathsf{B}_x\mathsf{E}$ with $\langle v, i|Y|u\rangle = \langle u, v|\sqrt{J_{\mathcal{C}}}|i\rangle$ is a Stinespring isometry of $\mathcal{C}$.*

Proof. Since $\sqrt{J_{\mathcal{C}}}$ is Hermitian,

$$\langle v|\operatorname{Tr}_{\mathsf{E}}(Y|u\rangle\langle u'|Y^\dagger)|v'\rangle = \sum_i \langle u, v|\sqrt{J_{\mathcal{C}}}|i\rangle\langle i|\sqrt{J_{\mathcal{C}}}|u', v'\rangle = \langle u, v|J_{\mathcal{C}}|u', v'\rangle = \langle v|\mathcal{C}(|u\rangle\langle u'|)|v'\rangle.$$

Thus $\operatorname{Tr}_{\mathsf{E}}(Y\rho Y^\dagger) = \mathcal{C}(\rho)$ for every matrix ρ. Since $\mathcal{C}$ preserves the trace, this gives $\operatorname{Tr}(Y^\dagger Y\rho) = \operatorname{Tr}\rho$ for every ρ, that is, $Y^\dagger Y = I$. □

To embed Y in a unitary, let the first n_{out} wires of $\mathsf{B}_x\mathsf{E}$ be those of B_x, and let $\iota\colon \mathsf{A}_x \to \mathsf{B}_x\mathsf{E}$ place the input on the first n_{in} wires, $\iota|\psi\rangle = |\psi\rangle|0^{2n_{\text{out}}}\rangle$. With one further qubit F, the isometries $|0\rangle \otimes \iota$ and $|1\rangle \otimes Y$ have orthogonal ranges, and a reflection exchanges them. Indeed, let

$$Z = |0\rangle \otimes \iota - |1\rangle \otimes Y, \qquad W = I - ZZ^\dagger.$$

Then $Z^\dagger Z = 2I$, so W is a reflection, and $Z^\dagger(|0\rangle \otimes \iota|\psi\rangle) = |\psi\rangle$ gives

$$W(|0\rangle \otimes \iota|\psi\rangle) = |0\rangle \otimes \iota|\psi\rangle - Z|\psi\rangle = |1\rangle \otimes Y|\psi\rangle.$$

Hence applying W to an input placed by ι and keeping only the wires of B_x implements $\mathcal{C}$ exactly. It remains to compute the entries of W. We say that a quantity is computable in polynomial space if it can be computed to precision t, with $O(t+1)$ bits per component, in space polynomial in $|x| + r + t$.

Lemma 6.4. *The entries of W are computable in polynomial space.*

Proof. The main step is the square root of $J_{\mathcal{C}}$, which we obtain from the binomial series of $\sqrt{1-z}$ applied to a matrix between 0 and I. By Lemma 2.6(vii) the entries of $J_{\mathcal{C}}$ are computable in polynomial space, and $\operatorname{Tr} J_{\mathcal{C}} = \sum_u \operatorname{Tr} \mathcal{C}(|u\rangle\langle u|) = d_{\text{in}}$, so $X = I - d_{\text{in}}^{-1} J_{\mathcal{C}}$ satisfies $0 \preceq X \preceq I$ and $\sqrt{J_{\mathcal{C}}} = 2^{n_{\text{in}}/2}\sqrt{I-X}$.

Let $c_k = \prod_{j=1}^{k} \frac{2j-1}{2j}$. For $0 \le z < 1$, Newton's binomial series gives

$$\sqrt{1-z} = 1 - \sum_{k \ge 1} \frac{c_{k-1}}{2k} z^k.$$

Since $\frac{c_{k-1}}{2k} = c_{k-1} - c_k$, the coefficients are positive and those with $k \le M$ sum to $1 - c_M$. From $(2j-1)(2j+1) \le (2j)^2$ we get $c_M^2 \le \prod_{j \le M} \frac{2j-1}{2j+1} = \frac{1}{2M+1}$. So the coefficients sum to one, the identity holds also at $z = 1$, and the terms with $k > M$ sum to at most c_M for all $0 \le z \le 1$. Applying this to the eigenvalues of X with $M = 4^{t+3}$ gives

$$\left\| \sqrt{I-X} - I + \sum_{k=1}^{M} \frac{c_{k-1}}{2k} X^k \right\| \le c_M < 2^{-t-3}.$$

The number of terms is exponential in t, but the space used by Lemma 2.6(v) and (iii) depends on it only through its logarithm. Term k is the product of $X/(2k)$ and the matrices $\frac{2j-1}{2j} X$ for $j < k$. Every product of consecutive factors is a number in $[0,1]$ times a power of X, so its entries have modulus at most one, and by Lemma 2.6(v) the entries of term k can be computed in polynomial space. Computing each term to precision $t + \log M + 3$, summing exactly by Lemma 2.6(iii) and rounding to $t+3$ fractional bits gives the entries of $\sqrt{I-X}$ with error below 2^{-t}, since the truncation, the combined error of the M computed terms and the rounding each contribute at most 2^{-t-3}.

The entries of Y are entries of $\sqrt{J_{\mathcal{C}}} = 2^{n_{\text{in}}/2}\sqrt{I-X}$, so they are computable in polynomial space by Lemma 2.6(ii). The entries of Z are entries of ι or of $-Y$, and an entry of $ZZ^\dagger$ is a sum of d_{in} products of entries of Z and their conjugates, so it can be computed by Lemma 2.6(iii). □

Proof of Theorem 6.2. Theorem 3.1 concerns one unitary for each length, while we have one unitary W for each instance and precision, so we collect them with a control register. Let $(\mathcal{C}_{x,r})_{x,r}$ be a family for U. For $r \ge 1$, let $y = 1^r 0 x$ encode $(x, 1^r)$, and let W_y be the unitary W for $\mathcal{C}_{x,r}$. For all other strings y, let W_y be the identity. Since n_{in} and n_{out} are polynomially bounded, we can fix an integer c such that $(|y|+1)^c$ wires hold $|y|$ control wires and the wires of W_y, for all y. Let $U_{\mathcal{C}}$ be the unitary language whose member of length $(\ell+1)^c$, for $\ell \ge 1$, is

$$\bigoplus_{y \in \{0,1\}^\ell} W_y$$

on ℓ control wires and the remaining data wires, where W_y acts on the first data wires, which are F followed by $\mathsf{B}_x\mathsf{E}$. Its members of all other lengths are identities. An entry of a member is zero if its two control strings differ, and otherwise it is an entry of $W_y \otimes I$, given by Lemma 6.4. Thus $U_{\mathcal{C}}$ has entries computable in polynomial space in the sense of Definition 2.4, and Theorem 3.1 with $A = \emptyset$ gives $B \in \mathsf{PSPACE}$ and time-uniform circuits $C_{n,m}$ over $\mathcal{G}$ such that $\|C^B_{n,m} - \widetilde{U}\| \le 2^{-m}$ for some extension $\widetilde{U}$ of the member of length n.

Given $(x, 1^r)$, let $y = 1^{2r}0x$, $n = (|y|+1)^c$ and $m = \lceil \log r \rceil + 2$. The general quantum circuit $\mathcal{C}'_{x,r}$ has as input wires the first n_{in} wires of $\mathsf{B}_x\mathsf{E}$, and all other wires of $C_{n,m}$ are new qubits. It sets the control register to $|y\rangle$ by NOT gates, runs $C_{n,m}$, and discards all wires except the first n_{out} wires of $\mathsf{B}_x\mathsf{E}$, which are its output wires. Since the problem is size-computable and $(C_{n,m})_{n,m}$ is time-uniform, the family $(\mathcal{C}'_{x,r})_{x,r}$ is time-uniform.

With the extension $\widetilde{U}$ in place of $C^B_{n,m}$, this circuit applies W_y with all ancillas at zero, so it implements $\mathcal{C}_{x,2r}$ exactly. Since $\|C^B_{n,m} - \widetilde{U}\| \le 2^{-m}$, we get $\|\mathcal{C}'_{x,r} - \mathcal{C}_{x,2r}\|_\diamond \le 2^{1-m} \le 1/(2r)$ (Section 2.4). Some channel completion Φ of U_x satisfies $\|\mathcal{C}_{x,2r} - \Phi\|_\diamond \le 1/(2r)$, so $\|\mathcal{C}'_{x,r} - \Phi\|_\diamond \le 1/r$. □

6.2 The Collapse Criterion

The collapse question of [BEM+26, Section 1.4] is whether $\mathsf{P} = \mathsf{PSPACE}$ can fail to imply $\mathsf{unitaryBQP}^\diamond = \mathsf{unitaryPSPACE}^\diamond$, and [BEM+26, Open Problem 4] asks whether these classes can be separated unconditionally. On size-computable problems the criterion is that of Corollary 1.3. On all problems the register sizes add one requirement, since they must be computable in polynomial time in $\mathsf{unitaryBQP}^\diamond$, but only in polynomial space in $\mathsf{unitaryPSPACE}^\diamond$.

Theorem 6.5. *The following hold in the channel model.*

(i) For every language L, every size-computable problem in $\mathsf{unitaryPSPACE}^\diamond$ lies in $\mathsf{unitaryBQP}^{\diamond,L}$ if and only if $\mathsf{PSPACE} \subseteq \mathsf{BQP}^L$. In particular, the size-computable problems in $\mathsf{unitaryPSPACE}^\diamond$ all lie in $\mathsf{unitaryBQP}^\diamond$ if and only if $\mathsf{BQP} = \mathsf{PSPACE}$, and they all lie in $\mathsf{unitaryBQP}^{\diamond,K}$ for every PSPACE-complete language K.

(ii) $\mathsf{unitaryBQP}^\diamond = \mathsf{unitaryPSPACE}^\diamond$ if and only if $\mathsf{P} = \mathsf{PSPACE}$.

Proof. For (i), suppose that $\mathsf{PSPACE} \subseteq \mathsf{BQP}^L$, and let $U \in \mathsf{unitaryPSPACE}^\diamond$ be size-computable. The proof of Theorem 6.2 gives a unitary language $U_\mathcal{C} \in \mathsf{unitaryP}^B$ with circuits over $\mathcal{G}$, for some $B \in \mathsf{PSPACE} \subseteq \mathsf{BQP}^L$. By Corollary 5.2, $U_\mathcal{C} \in \mathsf{unitaryP}^L$ with circuits over $\mathcal{G}$. The rest of that proof applies to these circuits and gives a time-uniform family for U with oracle gates O_L.

Conversely, suppose that every size-computable problem in $\mathsf{unitaryPSPACE}^\diamond$ lies in $\mathsf{unitaryBQP}^{\diamond,L}$, and let $L' \in \mathsf{PSPACE}$. Let U_x be the oracle gate $O_{L'}$ on $|x|$ query qubits and one answer qubit, so that the circuits for U_{1^n} decide L' on strings of length n. A generator can enumerate the strings z of length $|x|$ and, for each $z \in L'$, write a NOT on the answer qubit controlled on the query register being z, by Toffoli and NOT gates with ancillas that return to zero. Thus U is size-computable and lies in $\mathsf{unitaryPSPACE}^\diamond$ with error zero, so it has a time-uniform family with oracle gates O_L. The only channel completion of the unitary U_x is its unitary channel. Hence, for $z \in \{0,1\}^n$, the output state ρ of the circuit at $(1^n, 1^8)$ on input $|z, 0\rangle$ satisfies $D(\rho, |z, L'(z)\rangle\langle z, L'(z)|) \le 1/16$, and measuring the answer qubit gives $L'(z)$ except with probability at most $1/16$. These circuits form a time-uniform family indexed by 1^n, so $L' \in \mathsf{BQP}^L$.

The statement about $\mathsf{BQP} = \mathsf{PSPACE}$ is the case $L = \emptyset$, since $\mathsf{BQP} \subseteq \mathsf{PSPACE}$ by Lemma 2.6(vii). For a PSPACE-complete language K we have $\mathsf{PSPACE} \subseteq \mathsf{P}^K \subseteq \mathsf{BQP}^K$.

For (ii), a time-uniform family is space-uniform, so $\mathsf{unitaryBQP}^\diamond \subseteq \mathsf{unitaryPSPACE}^\diamond$. Suppose that $\mathsf{P} = \mathsf{PSPACE}$, and let $U \in \mathsf{unitaryPSPACE}^\diamond$. Counting the wires in the input and output lists that the generator writes on $(x, 1^1)$ computes $n_{\text{in}}(x)$ and $n_{\text{out}}(x)$ in polynomial space. So each bit of these numbers is decided by a language in $\mathsf{PSPACE} = \mathsf{P}$, and U is size-computable. Since $\mathsf{PSPACE} = \mathsf{P} \subseteq \mathsf{BQP}$, part (i) gives $U \in \mathsf{unitaryBQP}^\diamond$. Conversely, if the two classes are equal and $L' \in \mathsf{PSPACE}$, then the problem of Example 6.1 lies in $\mathsf{unitaryBQP}^\diamond$, so it is size-computable and $L' \in \mathsf{P}$. □

Remark 6.6. By Theorem 6.5(ii), $\mathsf{P} = \mathsf{PSPACE}$ does imply $\mathsf{unitaryBQP}^\diamond = \mathsf{unitaryPSPACE}^\diamond$, and separating these classes is the same as proving $\mathsf{P} \neq \mathsf{PSPACE}$. The possibility raised in [BEM+26, Section 3.7] that the classes differ while $\mathsf{BQP} = \mathsf{PSPACE}$ occurs exactly when $\mathsf{BQP} = \mathsf{PSPACE} \neq \mathsf{P}$, and then only problems that are not size-computable separate them.

7 Interactive Synthesis

By Theorem 6.2, a size-computable problem in $\mathsf{unitaryPSPACE}^{\diamond}$ can be implemented in polynomial time with a language in PSPACE as an oracle. A polynomial-time verifier can also obtain the results of polynomial-space computations by interacting with an untrusted prover. For decision problems this gives exactly polynomial space [JJUW11], and the same holds for preparing states [RY22; MY23] and for unitaries in the average case, where the input is one register of a given state that can be prepared in polynomial space [BEM+26, Theorem 7.10]. For size-computable problems, Theorem 1.5 answers the worst-case question posed in [BEM+26, Open Problem 10]: such a problem lies in $\mathsf{unitaryPSPACE}^{\diamond}$ if and only if it lies in $\mathsf{unitaryQIP}^{\diamond}_{1,1/2}$.

The difficulty in the worst case is that the verifier holds the only copy of its input, and a prover that receives this copy could alter it without being detected [BEM+26, Remark 4.2], [Ros23, Section 1.4.2]. Our verifier never sends its input. It runs a circuit of Theorem 6.2 on its own qubits and uses the prover only for the oracle gates, so the inclusion in $\mathsf{unitaryQIP}^{\diamond}_{1,1/2}$ follows from the next theorem, which answers [Ros23, Open Problem 29].

Theorem 7.1. *Let $L' \in \mathsf{PSPACE}$. Every problem in $\mathsf{unitaryBQP}^{\diamond,L'}$ lies in $\mathsf{unitaryQIP}^{\diamond}_{1,1/2}$.*

The oracle gate $O_{L'}$ is a phase unitary up to Hadamard gates, and the phase state of its function can be prepared in polynomial space, so interactive state synthesis lets the verifier obtain this state from the prover. The verifier consumes the state by state injection, which applies the phase unitary to the verifier's own qubits up to a correction (Section 7.1). The remaining problem is that the verifier obtains many such states from one prover, which keeps its memory between the calls, and that acceptance in a single call can have small probability. Because the target states are pure, the subnormalized state on acceptance is close to a product with the prover's memory, however small the acceptance probability, and this lets us compare a whole run with a run in which all phase states are exact (Section 7.2).

7.1 State Injection

For $f\colon \{0,1\}^{\ell} \to \{0,1\}$, recall from Section 5.3 the phase unitary $Z_f|u\rangle = (-1)^{f(u)}|u\rangle$ and the binary phase state $|\varphi_f\rangle = 2^{-\ell/2}\sum_u (-1)^{f(u)}|u\rangle$. To inject $|\varphi_f\rangle$ into an ℓ-qubit register X, we apply a CNOT from each qubit of X onto the corresponding qubit of a register C holding the resource state $|\varphi_f\rangle$, and we measure C [ZLC00; GC99]. The CNOT gates map $|u\rangle|\varphi_f\rangle$ to $2^{-\ell/2}|u\rangle\sum_{u'}(-1)^{f(u')}|u'\oplus u\rangle$, and outcome v selects the term $u' = u \oplus v$. So on every input, and with any further registers, each outcome has probability $2^{-\ell}$, and outcome v applies to X the diagonal unitary with entries $(-1)^{f(u\oplus v)}$, which is

$$Z_f Z_{\partial_v f}, \qquad \text{where} \quad \partial_v f(u) = f(u) \oplus f(u \oplus v).$$

This is Z_f up to a correction that is again a phase unitary. A further injection removes the correction and leaves a new one, and these corrections vanish after $\ell + 1$ injections.

Lemma 7.2. *Let $f_0\colon \{0,1\}^{\ell} \to \{0,1\}$. Inject $|\varphi_{f_0}\rangle, \dots, |\varphi_{f_\ell}\rangle$ into X in this order, where $f_{j+1} = \partial_{v_j} f_j$ and v_j is the outcome of the injection of $|\varphi_{f_j}\rangle$. For all outcomes, these injections apply Z_{f_0} to X.*

Proof. Phase unitaries commute and square to the identity, so the injections apply $\prod_{j=0}^{\ell} Z_{f_j} Z_{f_{j+1}} = Z_{f_0} Z_{f_{\ell+1}}$. By induction on j,

$$f_j(u) = \bigoplus_{J \subseteq \{0,\dots,j-1\}} f_0\Big(u \oplus \bigoplus_{i \in J} v_i\Big). \tag{7.1}$$

For $j = \ell + 1$, the map $J \mapsto \bigoplus_{i \in J} v_i$ is linear from $\mathbb{F}_2^{\ell+1}$ to $\mathbb{F}_2^{\ell}$, so it has a nontrivial kernel and takes each value an even number of times. Hence $f_{\ell+1} = 0$. Alternatively, each derivative lowers the degree of the function as a polynomial over $\mathbb{F}_2$, and f_0 has degree at most ℓ. □

For $L' \in \mathsf{PSPACE}$, let $f_0(u, b) = b \cdot L'(u)$ on $\ell + 1$ bits. Then $O_{L'}$ on ℓ query qubits is Z_{f_0} conjugated by H on the answer qubit. The verifier obtains the states $|\varphi_{f_j}\rangle$ of Lemma 7.2 for this function from the prover by interactive state synthesis, which applies to states that can be prepared in polynomial space.

A family $(|\chi_x\rangle)_x$ of pure states indexed by strings lies in $\mathsf{statePSPACE}$ if a space-uniform family of general quantum circuits of space polynomial in $|x| + r$ prepares, on $(x, 1^r)$, a state within trace distance $1/r$ of $|\chi_x\rangle$ [BEM+26, Definition 2.11 and Remark 2.14]. The states $|\varphi_{f_j}\rangle$, indexed by a string that encodes 1^ℓ and $v_0, \ldots, v_{j-1}$ for $j \le \ell + 1$, together with the state $|0\rangle$ at all other strings, form such a family. Indeed, by Equation (7.1) the value $f_j(u, b)$ can be computed from this string and (u, b) in polynomial space, so a generator can write Hadamard gates on all qubits and then, for each string on which f_j is one, a phase flip on that basis state, built from NOT, Toffoli and T^4 gates with ancillas returned to zero. These circuits prepare the states exactly.

7.2 Resource States from the Prover

Interactive state synthesis provides verifiers of the following kind.

Definition 7.3. A *verifier for a pure state* $|\chi\rangle$ with error ε is a verifier with no input and an output register C such that some prover is accepted with probability one and every prover accepted with probability at least $1/2$ leaves C within trace distance ε of χ, conditioned on acceptance.

Fact 7.4 ([Ros24, Theorem 1.5 and Section 6], [BEM+26, Section 7.1])**.** *Let* $(|\chi_x\rangle)_x \in \mathsf{statePSPACE}$*, where the number of qubits of* $|\chi_x\rangle$ *is computable in polynomial time from* x*. A deterministic polynomial-time machine maps* $(x, 1^{r'})$ *to a verifier for* $|\chi_x\rangle$ *with error* $1/r'$*.*

This guarantee is conditioned on a large acceptance probability and says nothing about the prover's memory. When many such verifiers run as subroutines against the same prover, one of them may accept with small probability on some measurement outcomes. The next lemma bounds the subnormalized state on acceptance, for every acceptance probability and together with the memory.

Lemma 7.5. *Consider a verifier for* $|\chi\rangle$ *with error* ε *and any prover with memory* M*, which we take to include the message register. Let* Ω *be the subnormalized state of* CM *on acceptance, that is, the final state projected onto acceptance and reduced to* CM*, so that* $\operatorname{Tr}\Omega$ *is the acceptance probability. Then* $\rho_{\mathsf{M}} = \langle\chi|\Omega|\chi\rangle$ *satisfies*

$$\|\Omega - \chi \otimes \rho_{\mathsf{M}}\|_1 \le 3\sqrt{\varepsilon}.$$

Proof. Let $\Pi_\chi = \chi \otimes I$, $p_{\mathrm{acc}} = \operatorname{Tr}\Omega$ and $\varepsilon' = \operatorname{Tr}((I - \Pi_\chi)\Omega)$. To bound ε', consider the prover that uses a private coin to follow the given prover or an honest prover, each with probability $1/2$. It is accepted with probability $(1 + p_{\mathrm{acc}})/2 \ge 1/2$, so, conditioned on acceptance, the measurement $\{\chi, I - \chi\}$ on C gives $I - \chi$ with probability at most ε, since χ itself gives this outcome with probability zero. Acceptance with outcome $I - \chi$ has probability at least $\varepsilon'/2$, so $\varepsilon'/2 \le \varepsilon(1 + p_{\mathrm{acc}})/2$ and $\varepsilon' \le 2\varepsilon$.

Since $|\chi\rangle$ is pure, $\Pi_\chi\Omega\Pi_\chi = \chi \otimes \rho_{\mathsf{M}}$, and $\Omega - \Pi_\chi\Omega\Pi_\chi = (I - \Pi_\chi)\Omega + \Pi_\chi\Omega(I - \Pi_\chi)$. Writing $\Omega = \sqrt{\Omega}\sqrt{\Omega}$ and using $\|XX'\|_1 \le \|X\|_F\|X'\|_F$, we get

$$\|\Omega - \Pi_\chi\Omega\Pi_\chi\|_1 \le \sqrt{\varepsilon' p_{\mathrm{acc}}} + \sqrt{(p_{\mathrm{acc}} - \varepsilon')\varepsilon'} \le 2\sqrt{\varepsilon'} \le 2\sqrt{2\varepsilon} \le 3\sqrt{\varepsilon}.$$ □

With this lemma, a verifier can use states obtained from the prover as if they were exact, with an error that adds over the calls.

Lemma 7.6. *Let ψ_0 be the initial state of a private register X of a verifier together with a further register R on which neither party acts. In each of n_{inj} steps, the verifier sends the outcomes v of the earlier steps to the prover and runs, on new qubits and the message register set to zero, a verifier for a pure state $|\chi_v\rangle$ with error ε. It keeps the acceptance qubit of this call, applies to X and the output register C of the call an instrument that depends on v, and appends the outcome to v. It accepts at the end if all acceptance qubits are one.*

Suppose that if each call produced exactly $|\chi_v\rangle$, every complete sequence of outcomes would leave the same state ψ' on XR. Then for every prover whose memory is initially independent of XR and whose acceptance probability p_{acc} is positive, the state σ of XR conditioned on acceptance satisfies

$$D(\sigma, \psi') \leq 3 n_{\mathrm{inj}} \sqrt{\varepsilon} / p_{\mathrm{acc}}.$$

Proof. We keep the prover fixed and compare a run with its ideal execution, in which each accepted resource state is replaced by its target. By Lemma 7.5, this replacement in one step changes the subnormalized state by at most $3\sqrt{\varepsilon}$ in trace norm, however small the acceptance probability of an individual call is. In the ideal execution, XR is in product with the prover's memory on each sequence of outcomes.

Write M for the memory of the prover, including the message register. After a sequence v of outcomes of the earlier steps, step j acts on M by a map $\mathcal{A}_{j,v}$ to CM, which sends v to the prover, runs the call of this step, projects its acceptance qubit onto one and discards this qubit and the other new qubits of the call except C. Its ideal version is $\chi_v \otimes \mathcal{T}_{j,v}$, where χ_v is the target of the call and $\mathcal{T}_{j,v}(\tau) = \langle \chi_v | \mathcal{A}_{j,v}(\tau) | \chi_v \rangle$. For a state ω on M and a further register, which we may include in the memory of the prover of the call, Lemma 7.5 gives

$$\| (\mathcal{A}_{j,v} \otimes \mathrm{id})(\omega) - \chi_v \otimes (\mathcal{T}_{j,v} \otimes \mathrm{id})(\omega) \|_1 \leq 3\sqrt{\varepsilon}.$$

The instrument $\mathcal{B}_j$ of step j appends its outcome to the sequence and removes C. The whole step is $\mathcal{F}_j$, which applies $\mathcal{A}_{j,v}$ on the block of each v and then $\mathcal{B}_j$, and its ideal version $\widetilde{\mathcal{F}}_j$ has $\chi_v \otimes \mathcal{T}_{j,v}$ in place of $\mathcal{A}_{j,v}$. These maps are completely positive and trace nonincreasing, so they do not increase the trace norm of Hermitian matrices. The blocks of different v are orthogonal, so scaling the bound above gives $\| (\mathcal{F}_j - \widetilde{\mathcal{F}}_j)(\omega) \|_1 \leq 3\sqrt{\varepsilon} \operatorname{Tr} \omega$ for every positive ω.

Starting from $\rho_0 = \psi_0 \otimes \tau_0$, with τ_0 the initial state of M, a run and its ideal execution produce $\rho = \mathcal{F}_{n_{\mathrm{inj}}} \cdots \mathcal{F}_1(\rho_0)$ and $\widetilde{\rho} = \widetilde{\mathcal{F}}_{n_{\mathrm{inj}}} \cdots \widetilde{\mathcal{F}}_1(\rho_0)$. Acceptance qubits are not changed after their call, so ρ is the subnormalized final state on acceptance. By the telescoping identity

$$\rho - \widetilde{\rho} = \sum_{j=1}^{n_{\mathrm{inj}}} \mathcal{F}_{n_{\mathrm{inj}}} \cdots \mathcal{F}_{j+1} (\mathcal{F}_j - \widetilde{\mathcal{F}}_j) \widetilde{\mathcal{F}}_{j-1} \cdots \widetilde{\mathcal{F}}_1(\rho_0),$$

we get $\| \rho - \widetilde{\rho} \|_1 \leq \eta$ for $\eta = 3 n_{\mathrm{inj}} \sqrt{\varepsilon}$.

It remains to describe $\widetilde{\rho}$. If v has positive probability when each call produces exactly $|\chi_v\rangle$, let ψ_v be the state of XR on these outcomes in that case. By induction on j,

$$\widetilde{\mathcal{F}}_j \cdots \widetilde{\mathcal{F}}_1(\rho_0) = \sum_v |v\rangle\langle v| \otimes \psi_v \otimes \tau_v, \qquad \tau_v \succeq 0,$$

where v ranges over the sequences of outcomes of the first j steps that have positive probability. Indeed, the maps $\chi_v \otimes \mathcal{T}_{j,v}$ act on M alone and produce χ_v, and on $\psi_v \otimes \chi_v$ the instrument gives

each outcome v' with some probability $c_{v'}$ and, if $c_{v'} > 0$, leaves $\psi_{vv'}$ on XR, so $\tau_{vv'} = c_{v'}\mathcal{T}_{j,v}(\tau_v)$. By hypothesis $\psi_v = \psi'$ for every complete sequence v, so the reduced state of $\widetilde{\rho}$ on XR is $(\operatorname{Tr}\widetilde{\rho})\psi'$. Hence $\|\rho_{\mathsf{XR}} - (\operatorname{Tr}\widetilde{\rho})\psi'\|_1 \le \eta$, and taking the trace gives $|p_{\mathrm{acc}} - \operatorname{Tr}\widetilde{\rho}| \le \eta$. Since $\sigma = \rho_{\mathsf{XR}}/p_{\mathrm{acc}}$,

$$D(\sigma, \psi') = \frac{\|\rho_{\mathsf{XR}} - p_{\mathrm{acc}}\psi'\|_1}{2p_{\mathrm{acc}}} \le \frac{\eta}{p_{\mathrm{acc}}}. \qquad \square$$

7.3 A Verifier for Oracle Circuits

To prove Theorem 7.1, let $L' \in \mathsf{PSPACE}$ and let $U \in \mathsf{unitaryBQP}^{\diamond,L'}$, with a time-uniform family $(\mathcal{C}_{x,r})_{x,r}$ and channel completions $\Phi_{x,r}$ of U_x such that $\|\mathcal{C}_{x,r} - \Phi_{x,r}\|_\diamond \le 1/r$. Fix x and r, and write $\mathcal{C} = \mathcal{C}_{x,r}$. Let $\ell_1, \ldots, \ell_Q$ be the numbers of query qubits of the oracle gates of $\mathcal{C}$, and let $n_{\mathrm{inj}} = \sum_i(\ell_i + 2)$ and $r' = (12(n_{\mathrm{inj}} + 1)r)^2$.

The verifier $V_{x,r}$ runs $\mathcal{C}$ on its ancilla register, with the input of $\mathcal{C}$ on the qubits that hold the verifier's input, and it performs the non-oracle operations of $\mathcal{C}$ itself. It replaces an oracle gate on ℓ query qubits by $\ell + 2$ steps between two Hadamard gates on the answer qubit. In step j, for $j = 0, \ldots, \ell + 1$, let $v_0, \ldots, v_{j-1}$ be the outcomes of the earlier steps for this gate. The verifier sends all injection outcomes so far to the prover, sets the message register to zero, and runs the verifier of Fact 7.4 for the family of Section 7.1, with target $|\varphi_{f_j}\rangle$ and precision r', keeping the acceptance qubit of this call. Then it injects the output register C of the call into the query and answer qubits, which gives the outcome v_j. At the end, $V_{x,r}$ moves the output wires of $\mathcal{C}$ to B_x and accepts if the acceptance qubits of all calls are one.

The call in a step depends on the outcomes, while the circuits of $V_{x,r}$ are fixed before the interaction. So $V_{x,r}$ simulates the machine of Fact 7.4 reversibly with NOT and Toffoli gates [Ben73; Tof80], computing the description of the required verifier on ancillas from the outcome wires, and it pads this description so that the numbers of wires, message qubits and rounds do not depend on the outcomes. For each position in this description and each possible operation, $V_{x,r}$ applies the operation controlled on the description naming it at this position, using the exact controlled gates of Lemma 4.3. A measurement becomes a CNOT onto a new qubit, an initialization takes a new qubit at zero, and a discarded qubit is left untouched. The new qubits of measurements and the untouched discarded qubits stay private until the call ends and are then discarded, so on each sequence of outcomes the accepted state of C and the prover's memory is that of the verifier of Fact 7.4. The extra message qubits and rounds are idle, so a prover for the padded verifier is a prover for this verifier, with these qubits in its memory. The generator of $\mathcal{C}$ and the machine of Fact 7.4 run in polynomial time, so $V_{x,r}$ can be written in time polynomial in $|x| + r$.

Proof of Theorem 7.1. An honest prover reads the outcomes and acts as the honest prover of each call, so it is accepted with probability one on every input. For soundness, let ψ be a pure state on $\mathsf{A}_x\mathsf{R}$, and consider a prover with acceptance probability $p_{\mathrm{acc}} \ge 1/2$. The verifier $V_{x,r}$ has the form of Lemma 7.6 with $\varepsilon = 1/r'$, where X consists of the qubits of $\mathcal{C}$ present at each point, ψ_0 is the state of XR after the operations on X that precede the first call, and the instrument of a step is its injection followed by the operations on X that precede the next call, so that finally X is B_x. If each call produced its target exactly, Lemma 7.2 would give every oracle gate exactly on all outcomes, so every complete sequence of outcomes would leave $\psi' = (\mathcal{C} \otimes \mathrm{id})(\psi)$ on $\mathsf{B}_x\mathsf{R}$. Thus the accepted output σ satisfies

$$D(\sigma, \psi') \le \frac{3n_{\mathrm{inj}}}{\sqrt{r'}\, p_{\mathrm{acc}}} \le \frac{6n_{\mathrm{inj}}}{\sqrt{r'}} \le \frac{1}{2r}.$$

Since $\|\mathcal{C} - \Phi_{x,r}\|_\diamond \le 1/r$, also $D(\psi', (\Phi_{x,r} \otimes \mathrm{id})(\psi)) \le 1/(2r)$, and hence $D(\sigma, (\Phi_{x,r} \otimes \mathrm{id})(\psi)) \le 1/r$, with the same channel completion for all ψ and all provers. $\square$

7.4 The Converse

Theorem 7.7. *A size-computable problem in* $\mathsf{unitaryQIP}^{\diamond}_{1,1/2}$ *lies in* $\mathsf{unitaryPSPACE}^{\diamond}$.

A polynomial-space circuit can run a verifier against a prover if the prover's actions are computable in polynomial space. For a verifier that starts from zero, a near-optimal prover has this property.

Fact 7.8 ([MY23, Theorem 7.1], [BEM+26, Theorem 7.9])**.** *Let* $(V_x)_x$ *be a time-uniform family of polynomial-time verifiers with polynomially many rounds that start with all registers at zero, let* $p_{\mathrm{opt}}(V_x)$ *be the supremum of the acceptance probabilities of* V_x *over all provers, and let* q *be a positive polynomial. There is a space-uniform family of general quantum circuits of polynomial space that implement provers* $\mathcal{P}_x$ *such that* $\mathcal{P}_x$ *is accepted by* V_x *with probability at least* $p_{\mathrm{opt}}(V_x) - 1/q(|x|)$.

The verifier of a synthesis problem has a quantum input, and the circuit must use one prover for all inputs. We optimize the prover on the all-zero input, and soundness forces the acceptance probability to be nearly the same on all inputs. Indeed, if a prover were accepted with very different probabilities on two inputs, then acceptance on a superposition of them, entangled with a purifying qubit, would change the distribution of this qubit, which no channel on the input can do.

Lemma 7.9. *Fix a verifier that satisfies the soundness condition of Definition 2.10 with error* ε *in place of* $1/r$*, and fix a prover. Let* $p(\psi)$ *be the acceptance probability on a pure input* $|\psi\rangle$ *on* A_x*. If* $p(\psi) \geq 3/4$*, then* $p(\psi') \geq p(\psi) - \frac{9}{2}\varepsilon$ *for every pure input* $|\psi'\rangle$.

Proof. Apply soundness to the input $\sqrt{2/3}|\psi\rangle|0\rangle + \sqrt{1/3}|\psi'\rangle|1\rangle$, where the second factor is one purifying qubit. Neither the verifier nor the prover acts on this qubit, so measuring it first does not change its joint distribution with acceptance. Thus the acceptance probability is $p_{\mathrm{acc}} = (2p(\psi)+p(\psi'))/3 \geq 1/2$, and acceptance with outcome 0 has probability $\frac{2}{3}p(\psi)$. A channel completion preserves the trace and does not act on this qubit, so its output has outcome 0 with probability $2/3$. Soundness therefore gives

$$\varepsilon \geq \frac{2p(\psi)}{3p_{\mathrm{acc}}} - \frac{2}{3} = \frac{2(p(\psi) - p(\psi'))}{9p_{\mathrm{acc}}},$$

and $p_{\mathrm{acc}} \leq 1$ gives the claim. □

High acceptance and soundness will show that on each input the circuit is close to some channel completion of U_x, which may depend on the input. The class $\mathsf{unitaryPSPACE}^{\diamond}$ requires one channel completion for all inputs, and a minimax argument provides it.

Lemma 7.10. *Let* U *be a partial isometry from a register* A *to a register* B*, and let* $\mathcal{C}$ *be a channel from* A *to* B*. Suppose that for every pure state* ψ *on* AR *there is a channel completion* Φ *of* U *with* $D((\mathcal{C}\otimes\mathrm{id})(\psi), (\Phi\otimes\mathrm{id})(\psi)) \leq \varepsilon$*. Then some channel completion* Φ *of* U *satisfies* $\|\mathcal{C} - \Phi\|_{\diamond} \leq 2\varepsilon$.

Proof. For a state ρ on A with a purification ψ_ρ, let

$$F(\rho, \Phi) = D((\mathcal{C}\otimes\mathrm{id})(\psi_\rho), (\Phi\otimes\mathrm{id})(\psi_\rho)).$$

Purifications of ρ differ by an isometry on the purifying register, so F does not depend on the purification, and the purification $(\sqrt{\rho}\otimes I)\sum_u |u,u\rangle$ shows that F is continuous. It is convex in Φ. It is concave in ρ: if $\rho = \lambda\rho_1 + (1-\lambda)\rho_2$, then $\sqrt{\lambda}|\psi_{\rho_1}\rangle|0\rangle + \sqrt{1-\lambda}|\psi_{\rho_2}\rangle|1\rangle$ purifies ρ, and measuring the last qubit does not increase the trace distance, which gives $F(\rho,\Phi) \geq \lambda F(\rho_1,\Phi) + (1-\lambda)F(\rho_2,\Phi)$. The states on A and the channel completions of U form compact convex sets, so the minimax theorem [Sio58] gives

$$\tfrac{1}{2}\min_{\Phi}\|\mathcal{C} - \Phi\|_{\diamond} = \min_{\Phi}\max_{\rho} F(\rho,\Phi) = \max_{\rho}\min_{\Phi} F(\rho,\Phi) \leq \varepsilon.$$ □

Proof of Theorem 7.7. Let U be size-computable with verifiers $V_{x,r}$. For $y = 1^{r'}0x$ with $r' \geq 1$, let V_y be $V_{x,r'}$ on the all-zero input, so that all its registers start at zero, and at other strings y let V_y be a fixed verifier. This family is time-uniform, so Fact 7.8 with $q(s) = s + 1$ gives provers $\mathcal{P}_y$. Now fix x and r, let $r' = 32r$, $\varepsilon = 1/r'$ and $y = 1^{r'}0x$, so that $\mathcal{P}_y$ has acceptance probability at least $p_{\mathrm{opt}}(V_y) - \varepsilon$ on the all-zero input. Let $\mathcal{C}_{x,r}$ be the general quantum circuit that runs $V_{x,r'}$ against $\mathcal{P}_y$ and discards everything except B_x. Its input wires are the first $n_{\mathrm{in}}(x)$ qubits of the ancilla register of $V_{x,r'}$, which the generator can list because U is size-computable. The family $(\mathcal{C}_{x,r})_{x,r}$ is space-uniform, of space polynomial in $|x| + r$.

If $\operatorname{supp} U_x = \{0\}$, every channel is a channel completion of U_x. Otherwise, an honest prover is accepted with probability one on a unit vector in $\operatorname{supp} U_x$, so by Lemma 7.9 it is accepted with probability at least $1 - \frac{9}{2}\varepsilon$ on the all-zero input. Hence $p_{\mathrm{opt}}(V_y) \geq 1 - \frac{9}{2}\varepsilon$, and $\mathcal{P}_y$ is accepted on the all-zero input with probability at least $1 - \frac{11}{2}\varepsilon > 3/4$. Applying Lemma 7.9 to $\mathcal{P}_y$ shows that it is accepted with probability at least $1 - 10\varepsilon$ on every pure input. The same holds on every input with a purifying register, since the reduced state of the input is a mixture of pure states.

Let ψ be a pure state on $\mathsf{A}_x\mathsf{R}$. Its acceptance probability p_{acc} against $\mathcal{P}_y$ is at least $1 - 10\varepsilon > 1/2$, so soundness gives a channel completion Φ of U_x with $D(\sigma, (\Phi \otimes \mathrm{id})(\psi)) \leq \varepsilon$ for the accepted output σ. The output of $\mathcal{C}_{x,r}$ is σ with probability p_{acc} and some other state otherwise, so

$$D\big((\mathcal{C}_{x,r} \otimes \mathrm{id})(\psi), (\Phi \otimes \mathrm{id})(\psi)\big) \leq p_{\mathrm{acc}}\varepsilon + 1 - p_{\mathrm{acc}} \leq 11\varepsilon.$$

By Lemma 7.10, one channel completion $\Phi_{x,r}$ of U_x satisfies

$$\|\mathcal{C}_{x,r} - \Phi_{x,r}\|_\diamond \leq 22\varepsilon < \frac{1}{r}. \qquad \square$$

Remark 7.11. The condition on the register sizes in Theorem 1.5 cannot be removed from the converse, and it cannot be removed from the inclusion in $\mathsf{unitaryQIP}^\diamond_{1,1/2}$ unless $\mathsf{P} = \mathsf{PSPACE}$. The description of a verifier does not fix its input size, because the input is placed on the first qubits of an ancilla register that otherwise starts at zero [BEM+26, Section 2.4]. Let L be a language outside PSPACE, and let U_x be the identity on two qubits if $x \in L$ and $|\psi\rangle \mapsto |\psi\rangle|0\rangle$ from one qubit to two qubits otherwise. One verifier for all x, which moves the first two qubits of its ancilla register to B_x and accepts, implements U_x with error zero. If U were in $\mathsf{unitaryPSPACE}^\diamond$, counting the input wires written by the generator would decide L in polynomial space.

The description of a verifier does fix its output register, so every problem in $\mathsf{unitaryQIP}^\diamond_{1,1/2}$ has n_{out} computable in polynomial time. Thus the problem of Example 6.1 lies in $\mathsf{unitaryQIP}^\diamond_{1,1/2}$ only if $L' \in \mathsf{P}$, so if every problem in $\mathsf{unitaryPSPACE}^\diamond$ lies in $\mathsf{unitaryQIP}^\diamond_{1,1/2}$ then $\mathsf{P} = \mathsf{PSPACE}$. Conversely, if $\mathsf{P} = \mathsf{PSPACE}$, every problem in $\mathsf{unitaryPSPACE}^\diamond$ is size-computable, as in the proof of Theorem 6.5(ii).

Together, Theorems 6.2, 7.1 and 7.7 prove Theorem 1.5.

AI Methodology Statement

The author used ChatGPT and Codex from OpenAI and Claude Code from Anthropic to help organize and polish the exposition of this paper, based on the author's own drafts. The same tools assisted in surveying the related literature, in understanding the current state of research in this area, in flagging possible errors in draft proofs, and in suggesting simpler proofs of some statements. The author checked, revised, and finalized every statement, proof, and reference in the paper. The Lean formalization was written almost entirely by these tools. Lean checks its proofs, and the author checked that its definitions and theorem statements match those in the paper. The author takes full responsibility for the content of the submission.

References

[Aar16] Scott Aaronson. The complexity of quantum states and transformations: From quantum money to black holes. Lecture notes, 28th McGill Invitational Workshop on Computational Complexity, 2016. arXiv:1607.05256.

[Aar21] Scott Aaronson. Open problems related to quantum query complexity. *ACM Transactions on Quantum Computing*, 2(4):14:1–14:9, 2021. arXiv:2109.06917, doi:10.1145/3488559.

[AFCB14] Mateus Araújo, Adrien Feix, Fabio Costa, and Časlav Brukner. Quantum circuits cannot control unknown operations. *New Journal of Physics*, 16(9):093026, 2014. arXiv:1309.7976, doi:10.1088/1367-2630/16/9/093026.

[AGL25] Prabhanjan Ananth, Aditya Gulati, and Yao-Ting Lin. Cryptography in the common Haar state model: Feasibility results and separations. In *Theory of Cryptography (TCC 2024)*, volume 15365 of *Lecture Notes in Computer Science*, pages 94–125. Springer, 2025. arXiv:2407.07908, doi:10.1007/978-3-031-78017-2_4.

[AK07] Scott Aaronson and Greg Kuperberg. Quantum versus classical proofs and advice. *Theory of Computing*, 3(7):129–157, 2007. arXiv:quant-ph/0604056, doi:10.4086/toc.2007.v003a007.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997. arXiv:quant-ph/9701001, doi:10.1137/S0097539796300933.

[BCC+14] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. Exponential improvement in precision for simulating sparse Hamiltonians. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 283–292. ACM, 2014. Numbered references are to arXiv version 2. arXiv:1312.1414v2, doi:10.1145/2591796.2591854.

[BCN25] John Bostanci, Boyang Chen, and Barak Nehoran. Oracle separation between quantum commitments and quantum one-wayness. In *Advances in Cryptology – EUROCRYPT 2025*, volume 15607 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2025. arXiv:2410.03358, doi:10.1007/978-3-031-91098-2_1.

[BCQ23] Zvika Brakerski, Ran Canetti, and Luowen Qian. On the computational hardness needed for quantum cryptography. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. arXiv:2209.04101, doi:10.4230/LIPIcs.ITCS.2023.24.

[BEM+26] John Bostanci, Yuval Efron, Tony Metger, Alexander Poremba, Luowen Qian, and Henry Yuen. Unitary complexity and the Uhlmann transformation problem. In *17th Innovations in Theoretical Computer Science Conference (ITCS 2026)*, volume 362 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. Numbered references are to arXiv version 3. arXiv:2306.13073v3, doi:10.4230/LIPIcs.ITCS.2026.24.

[Ben73] Charles H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, 1973. doi:10.1147/rd.176.0525.

[Bre76] Richard P. Brent. Fast multiple-precision evaluation of elementary functions. *Journal of the ACM*, 23(2):242–251, 1976. doi:10.1145/321941.321944.

[BVMS05] Ville Bergholm, Juha J. Vartiainen, Mikko Möttönen, and Martti M. Salomaa. Quantum circuits with uniformly controlled one-qubit gates. *Physical Review A*, 71(5):052330, 2005. arXiv:quant-ph/0410066, doi:10.1103/PhysRevA.71.052330.

[BY26] Zvika Brakerski and Henry Yuen. On scalable pseudorandom unitaries and the unitary synthesis problem. In *Advances in Cryptology – CRYPTO 2026*, volume 16804 of *Lecture Notes in Computer Science*, pages 372–401. Springer, 2026. arXiv:2605.09957, doi:10.1007/978-3-032-35409-9_13.

[CCH+26] Nai-Hui Chia, Kai-Min Chung, Tzu-Hsiang Huang, Chuhan Lu, and Jhih-Wei Shih. Complexity theory for quantum promise problems, 2026. arXiv:2411.03716v3.

[CCS25] Boyang Chen, Andrea Coladangelo, and Or Sattath. The power of a single Haar random state: Constructing and separating quantum pseudorandomness. In *Advances in Cryptology – EUROCRYPT 2025*, volume 15607 of *Lecture Notes in Computer Science*, pages 108–137. Springer, 2025. arXiv:2404.03295, doi:10.1007/978-3-031-91098-2_5.

[Cho75] Man-Duen Choi. Completely positive linear maps on complex matrices. *Linear Algebra and its Applications*, 10(3):285–290, 1975. doi:10.1016/0024-3795(75)90075-0.

[CM26] Ezekiel Cochran and Atul Mantri. Black hole radiation decoding in the Haar random oracle model, 2026. arXiv:2610.07124.

[CMY26] Boyang Chen, Tomoyuki Morimae, and Takashi Yamakawa. Quantum Pessiland, 2026. arXiv:2608.29493.

[DLM26] Fangqi Dong, Alex Lombardi, and Fermi Ma. Explicit separations for one-query unitary synthesis, 2026. arXiv:2607.26478.

[DN06] Christopher M. Dawson and Michael A. Nielsen. The Solovay–Kitaev algorithm. *Quantum Information and Computation*, 6(1):81–95, 2006. arXiv:quant-ph/0505030, doi:10.26421/QIC6.1-6.

[FR21] Bill Fefferman and Zachary Remscrim. Eliminating intermediate measurements in space-bounded quantum computation. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1343–1356. ACM, 2021. arXiv:2006.03530, doi:10.1145/3406325.3451051.

[GC99] Daniel Gottesman and Isaac L. Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402:390–393, 1999. arXiv:quant-ph/9908010, doi:10.1038/46503.

[GR02] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions, 2002. arXiv:quant-ph/0208112.

[GR22] Uma Girish and Ran Raz. Eliminating intermediate measurements using pseudorandom generators. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 76:1–76:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. arXiv:2106.11877, doi:10.4230/LIPIcs.ITCS.2022.76.

[GSLW19] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 193–204. ACM, 2019. Numbered references are to the arXiv version. arXiv:1806.01838, doi:10.1145/3313276.3316366.

[Hel76] Carl W. Helstrom. *Quantum Detection and Estimation Theory*, volume 123 of *Mathematics in Science and Engineering*. Academic Press, 1976.

[HRW26] Andrew Huang, Akshar Ramkumar, and John Wright. Random-oracle unitary synthesis is impossible, 2026. arXiv:2610.06700.

[Hua25] Ting Jia Huang. A 1.5-query lower bound for the unitary synthesis problem, 2025. arXiv:2508.13215.

[JJUW11] Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay, and John Watrous. QIP = PSPACE. *Journal of the ACM*, 58(6):30:1–30:27, 2011. arXiv:0907.4737, doi:10.1145/2049697.2049704.

[JLS18] Zhengfeng Ji, Yi-Kai Liu, and Fang Song. Pseudorandom quantum states. In *Advances in Cryptology – CRYPTO 2018*, volume 10993 of *Lecture Notes in Computer Science*, pages 126–152. Springer, 2018. arXiv:1711.00385, doi:10.1007/978-3-319-96878-0_5.

[KLL+17] Shelby Kimmel, Cedric Yen-Yu Lin, Guang Hao Low, Maris Ozols, and Theodore J. Yoder. Hamiltonian simulation with optimal sample complexity. *npj Quantum Information*, 3(1):13, 2017. arXiv:1608.00281, doi:10.1038/s41534-017-0013-7.

[Kni95] Emanuel Knill. Approximation by quantum circuits. Technical Report LAUR-95-2225, Los Alamos National Laboratory, 1995. arXiv:quant-ph/9508006.

[KQST23] William Kretschmer, Luowen Qian, Makrand Sinha, and Avishay Tal. Quantum cryptography in Algorithmica. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1589–1602. ACM, 2023. arXiv:2212.00879, doi:10.1145/3564246.3585225.

[KQT25] William Kretschmer, Luowen Qian, and Avishay Tal. Quantum-computable one-way functions without one-way functions. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing (STOC)*, pages 189–200. ACM, 2025. arXiv:2411.02554, doi:10.1145/3717823.3718144.

[Kre21] William Kretschmer. Quantum pseudorandomness and classical complexity. In *16th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2021)*, volume 197 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. Numbered references are to arXiv version 5, which corrects the conference version. arXiv:2103.09320v5, doi:10.4230/LIPIcs.TQC.2021.2.

[KT26] William Kretschmer and Ewin Tang. Unitary complexity in polynomial space, 2026. arXiv:2610.03705v2.

[LMR14] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10(9):631–633, 2014. arXiv:1307.0401, doi:10.1038/nphys3029.

[LMW24] Alex Lombardi, Fermi Ma, and John Wright. A one-query lower bound for unitary synthesis and breaking quantum cryptography. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC)*, pages 979–990. ACM, 2024. Numbered references are to the arXiv version. arXiv:2310.08870, doi:10.1145/3618260.3649650.

[Mac95] Ian G. Macdonald. *Symmetric Functions and Hall Polynomials*. Oxford Mathematical Monographs. Oxford University Press, second edition, 1995.

[MY23] Tony Metger and Henry Yuen. stateQIP = statePSPACE. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1349–1356. IEEE, 2023. Numbered references are to arXiv version 2. arXiv:2301.07730v2, doi:10.1109/FOCS57990.2023.00082.

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition, 2010. doi:10.1017/CBO9780511976667.

[Ope26] OpenAI. Polynomial-time unitary synthesis from a Boolean oracle. OpenAI Math Release preprint, October 5, 2026. URL: https://github.com/openai/math/tree/fd4aeeb2ee4fc729c18d98444fed42fd0529eeeb/preprints/Polynomial-Time-Unitary-Synthesis-from-a-Boolean-Oracle-October-5-2026.

[Ros23] Gregory Rosenthal. *Quantum State and Unitary Complexity*. PhD thesis, University of Toronto, 2023. URL: https://hdl.handle.net/1807/130326.

[Ros24] Gregory Rosenthal. Efficient quantum state synthesis with one query. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2508–2534. SIAM, 2024. Numbered references are to arXiv version 4. arXiv:2306.01723v4, doi:10.1137/1.9781611977912.89.

[Ros26] Gregory Rosenthal. Query and depth upper bounds for quantum unitaries via Grover search. *Quantum*, 10:2144, 2026. arXiv:2111.07992, doi:10.22331/q-2026-06-30-2144.

[RY22] Gregory Rosenthal and Henry Yuen. Interactive proofs for synthesizing quantum states and unitaries. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 112:1–112:4. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. Numbered references are to arXiv version 2. arXiv:2108.07192v2, doi:10.4230/LIPIcs.ITCS.2022.112.

[Sav70] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. doi:10.1016/S0022-0000(70)80006-X.

[SBM06] Vivek V. Shende, Stephen S. Bullock, and Igor L. Markov. Synthesis of quantum-logic circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(6):1000–1010, 2006. arXiv:quant-ph/0406176, doi:10.1109/TCAD.2005.855930.

[Sch27] Otto Schreier. Die Untergruppen der freien Gruppen. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 5(1):161–183, 1927. doi:10.1007/BF02952517.

[Sio58] Maurice Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171–176, 1958. doi:10.2140/pjm.1958.8.171.

[SM73] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time (preliminary report). In *Proceedings of the 5th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1–9. ACM, 1973. doi:10.1145/800125.804029.

[Tof80] Tommaso Toffoli. Reversible computing. In *Automata, Languages and Programming (ICALP 1980)*, volume 85 of *Lecture Notes in Computer Science*, pages 632–644. Springer, 1980. doi:10.1007/3-540-10003-2_104.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

[Wat99] John Watrous. Space-bounded quantum complexity. *Journal of Computer and System Sciences*, 59(2):281–326, 1999. doi:10.1006/jcss.1999.1655.

[Wat03] John Watrous. On the complexity of simulating space-bounded quantum computations. *Computational Complexity*, 12(1–2):48–84, 2003. doi:10.1007/s00037-003-0177-8.

[Yan22] Jun Yan. General properties of quantum bit commitments (extended abstract). In *Advances in Cryptology – ASIACRYPT 2022*, volume 13794 of *Lecture Notes in Computer Science*, pages 628–657. Springer, 2022. Cryptology ePrint Archive, Paper 2020/1488. doi:10.1007/978-3-031-22972-5_22.

[Zha24] Mark Zhandry. The space-time cost of purifying quantum computations. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, volume 287 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 102:1–102:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. arXiv:2401.07974, doi:10.4230/LIPIcs.ITCS.2024.102.

[Zha25] Mark Zhandry. How to model unitary oracles. In *Advances in Cryptology – CRYPTO 2025*, volume 16001 of *Lecture Notes in Computer Science*, pages 237–268. Springer, 2025. Cryptology ePrint Archive, Paper 2025/1072. doi:10.1007/978-3-032-01878-6_8.

[ZLC00] Xinlan Zhou, Debbie W. Leung, and Isaac L. Chuang. Methodology for quantum logic gate construction. *Physical Review A*, 62(5):052316, 2000. arXiv:quant-ph/0002039, doi:10.1103/PhysRevA.62.052316.