Oracle separations of hybrid quantum-classical circuits

Oracle separations of hybrid quantum-classical circuits

Atul Singh Arora,* Alexandru Gheorghiu,† Uttam Singh‡

January 2022

An important theoretical problem in the study of quantum computation, that is also practically relevant in the context of near-term quantum devices, is to understand the computational power of *hybrid models*, that combine polynomial-time classical computation with short-depth quantum computation. Here, we consider two such models: CQ_d which captures the scenario of a polynomial-time classical algorithm that queries a d-depth quantum computer many times; and QC_d which is more analogous to *measurement-based quantum computation* and captures the scenario of a d-depth quantum computer with the ability to change the sequence of gates being applied depending on measurement outcomes processed by a classical computation. Chia, Chung and Lai (STOC 2020) and Coudron and Menda (STOC 2020) showed that these models (with $d = \log^{O(1)}(n)$) are strictly weaker than BQP (the class of problems solvable by polynomial-time quantum computation), relative to an oracle, disproving a conjecture of Jozsa in the relativised world.

In this paper, we show that, despite the similarities between CQ_d and QC_d, the two models are incomparable, i.e. $CQ_d \not\subseteq QC_d$ and $QC_d \not\subseteq CQ_d$ relative to an oracle. In other words, we show that there exist problems that one model can solve but not the other and vice versa. We do this by considering new oracle problems that capture the distinctions between the two models and by introducing the notion of an *intrinsically stochastic oracle*, an oracle whose responses are inherently randomised, which is used for our second result. While we leave showing the second separation relative to a standard oracle as an open problem, we believe the notion of stochastic oracles could be of independent interest for studying complexity classes which have resisted separation in the standard oracle model. Our constructions also yield simpler oracle separations between the hybrid models and BQP, compared to earlier works.

---

*Corresponding author (atul.singh.arora@gmail.com; asarora@caltech.edu). Institute for Quantum Information and Matter, California Institute of Technology; Department of Computing and Mathematical Sciences, California Institute of Technology.

†Institute for Theoretical Studies, ETH Zürich

‡Centre for Theoretical Physics, Polish Academy of Sciences

Contents

1 Introduction

One of the major goals of modern complexity theory is understanding the relations between various models of efficient quantum computation and classical computation. It is widely believed that the set of problems solvable by polynomial-time quantum algorithms, denoted BQP, is strictly larger than that solvable by polynomial-time classical algorithms, denoted BPP [BV97; Sho97; Aru+19]. This, so-called Deutsch-Church-Turing thesis, is motivated by the existence of a number of problems which seem to be classically intractable, but which admit efficient (polynomial-time) quantum algorithms. Notable examples include period finding [Sim97], integer factorisation [Sho97], the discrete-logarithm problem [Sho97], solving Pell's equation [Hal02], approximating the Jones polynomial [AJL06] and others [Jor]. In a seminal paper of Cleve and Watrous [CW00], it was shown that many of these problems (period finding, factoring and discrete-logarithm) can also be solved by quantum circuits of logarithmic-depth together with classical pre- and post-processing. In the language of complexity theory, we say that these problems are contained in the complexity class BPP^{QNC}. This motivated Jozsa to conjecture that $\text{BPP}^{\text{QNC}} = \text{BQP}$ [Joz06]. In other words, it was conjectured that the computational power of general polynomial-size quantum circuits is fully captured by quantum circuits of polylogarithmic depth (the class QNC) together with classical pre- and post-processing.

In fact, Jozsa's conjecture was more broad than simply asserting that $\text{BPP}^{\text{QNC}} = \text{BQP}$. To explain why, one needs to distinguish between two *hybrid models* of classical computation and short-depth quantum computation. These two models capture the idea of interleaving polynomial-time classical computations with d-depth quantum computations. The first model, denoted CQ_d, refers to polynomial-time classical circuits that can call d-depth quantum circuits, as a subroutine, and use their output (see Figure 1c). This is analogous to BPP^{QNC}, except the quantum circuits are of depth d rather than $\log^{O(1)}(n)$ (where n is the size of the input). The second model, denoted QC_d, refers to d-depth quantum circuits that can, at each circuit layer, call polynomial-time classical circuits, as a subroutine and use their output (see Figure 1b). Importantly, the classical circuits *cannot be invoked coherently*—part of the quantum state produced by the circuit is measured and the classical subroutine is invoked on the classical outcome of that measurement.

Though CQ_d and QC_d circuits seem similar, they both capture different aspects of short-depth quantum computation together with polynomial-time classical computation. CQ_d captures the scenario of a classical algorithm that queries a short-depth quantum computer many times, while QC_d is more analogous to *measurement-based quantum computation* in that it captures a short-depth quantum computer with the ability to change the sequence of gates being applied depending on measurement outcomes processed by a classical computation.

Jozsa's conjecture is the assertion that the power of BQP is fully captured by logarithmic-depth quantum computation interspersed with polynomial-depth classical computation. This can be interpreted either as $\text{BQP} = \text{CQ}_\text{d}$ or $\text{BQP} = \text{QC}_\text{d}$ with $d = \log^{O(1)}(n)$. Aaronson later proposed as one of his "ten semi-grand challenges in quantum complexity theory" to find an oracle separation between BQP and the two hybrid models [Aar05]. That is, to provide an oracle O and a problem defined relative to this oracle which is contained in BQP^O but not in $\text{QC}_\text{d}{}^O$ or $\text{CQ}_\text{d}{}^O$. This was resolved in the landmark works of Chia, Chung and Lai [CCL20] and, independently, Coudron and Menda [CM20]. They showed that indeed such an oracle exists and also proved a hierarchy theorem for the two hybrid models showing that $\text{QC}_\text{d}{}^O \subsetneq \text{QC}_{\text{d}'}{}^O$ and $\text{CQ}_\text{d}{}^O \subsetneq \text{CQ}_{\text{d}'}{}^O$, whenever $d' = 2d + 1$. However, both works left it as an open problem whether there exists an oracle separation between QC_d and CQ_d.

1.1 Contributions

In our work, we resolve this question by showing that, indeed, there exist oracles O_1 and O_2 such that $\text{QC}_\text{d}{}^{O_1} \not\supseteq \text{CQ}_\text{d}{}^{O_1}$ and $\text{CQ}_\text{d}{}^{O_2} \not\supseteq \text{QC}_\text{d}{}^{O_2}$. In other words, the two hybrid models are incomparable—there exist problems that one can solve but not the other and vice versa. As corollaries, we also obtain simpler oracle constructions for separating the two hybrid models from BQP. It's important to note that for our results O_1 is a *standard oracle*, i.e. one that performs the mapping $|x\rangle\,|z\rangle \xrightarrow{O_1} |x\rangle\,|z \oplus f(x)\rangle$, for some function $f : X \rightarrow Z$. However, the oracle O_2 is one which we refer to as an *intrinsically stochastic oracle*, which is defined for a function $f : X \times Y \rightarrow Z$ and a probability distribution $\mathbb{F}_Y$ over Y. Classically, for each query $x \in X$, it samples a new $y \sim \mathbb{F}_Y$ and returns $f(x, y)$. Quantumly, for every application, it samples a new $y \sim \mathbb{F}_Y$ and performs the mapping $|x\rangle\,|z\rangle \xrightarrow{O_2} |x\rangle\,|z \oplus f(x, y)\rangle$.

We summarise our main results in Table 1. In particular, we obtain our first result by introducing an oracle problem that we call the *d-Serial Simon's* (d-SeS) Problem, which lets us show the following:

Theorem 1 (informal). *There exists a standard oracle, O, such that for all $d > 0$, $\text{QC}_\text{d}{}^O \not\supseteq \text{CQ}_\text{d}{}^O$.*
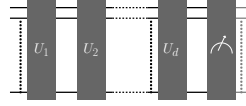
(a) QNC$_d$ scheme; U_i are single depth unitaries; Π is a projector in the computational basis.
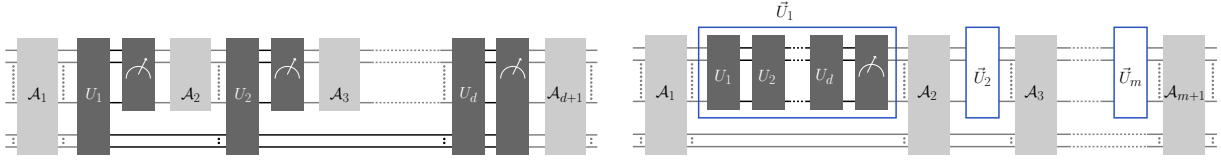
(b) QC$_d$ circuit; U_i are single layered unitaries, $\mathcal{A}_{c,i}$ are classical poly-sized circuits (in the figure, the subscript for $\mathcal{A}_c$ has been dropped) and the measurements are in the computational basis. Dark lines denote qubits.

(c) CQ$_d$ circuit; for clarity, we dropped the indices in $\mathcal{A}_c$ and the second indices in $U_{1,i}, U_{2,i} \ldots U_{d,i}$.

Figure 1: The three circuit models we consider.

	QC$_d$	CQ$_d$	Oracle Model	
d'-SeS	$d' + 1 \le d \le 2d' + 2$	$d \le 1$	Standard	This work
d'-SCS	$d \le 4$	$d' + 1 \le d \le d' + 5$	Stochastic	This work
d'-SSP	$d' + 1 \le d \le 2d' + 2$	$d' + 1 \le d \le 2d' + 1$	Standard	CCL

Table 1: Summary of our results: bounds on the smallest depth, d, needed to solve d'-Serial Simon's (d'-SeS) Problem and d'-Shuffled Collisions-to-Simon's (d'-SCS) Problem in the two hybrid models of computation compared to those of d'-Shuffling Simon's Problem (d'-SSP) introduced by CCL.

Intuitively, since CQ$_d$ can perform polynomially-many calls to a d-depth quantum circuit, we'd like to define a problem which forces one to sequentially solve d Simon's problems.[1] In other words, the period of each Simon's function can be viewed as a key which unlocks in the oracle a different Simon function. The d-SeS problem is to find the period of the d'th function. This problem can be solved in CQ$_1$, since the poly-time algorithm can invoke d separate constant-depth quantum circuits to solve each Simon problem. However, due to the serial nature of the problem, we show that no QC$_d$ algorithm can succeed.

For our second separation, we consider a problem we call the d-Shuffled Collisions-to-Simon's (d-SCS) Problem and show that:

Theorem 2 (informal). *There exists an intrinsically stochastic oracle, O, such that for all $d \ge 4$, CQ$_d^O \not\supseteq$ QC$_d^O$.*

This separation is perhaps more surprising because QC$_d$ has only d quantum layers while CQ$_d$ has polynomially-many quantum layers (as it can invoke d-depth quantum circuits polynomially-many times). Importantly, however, in CQ$_d$ the quantum states prepared when invoking a d-depth quantum circuit are measured entirely. In contrast, in QC$_d$ some qubits are measured and the outcomes are sent to a classical subroutine, while other qubits remain unmeasured and maintain their coherence in between the layers of classical computation. This observation is the basis for defining d-SCS, which can still be viewed as a version of Simon's problem, in which one has to find a secret period s. To explain the idea, we briefly recap the textbook quantum algorithm for Simon's problem. One first queries the oracle to the Simon function in superposition to obtain the state $\sum_x |x\rangle |g(x)\rangle$, where $g : \{0, 1\}^n \to \{0, 1\}^n$ is the Simon function, with period $s \in \{0, 1\}^n, s \ne 0^n$. The second register is then measured, yielding

$$\frac{1}{\sqrt{2}}(|x\rangle + |x \oplus s\rangle) |y\rangle \tag{1}$$

---

[1] Recall that Simon's problem is the following: given oracle access to a 2-to-1 function $f : \{0, 1\}^n \to \{0, 1\}^n$, such that there exists an $s \in \{0, 1\}^n, s \ne 0^n$, for which $f(x) = f(y) \iff x = y \oplus s$, find s. The function f is referred to as a Simon function.

with $y = g(x) = g(x \oplus s)$. Finally, by applying Hadamard gates and measuring the first register, one obtains a string $w \in \{0, 1\}^n$, such that $w \cdot s = 0$. Repeating this in parallel $O(n)$ times yields a system of linear equations from which s can be uniquely recovered.

Note that in Simon's algorithm, the main component of the quantum algorithm is to generate superpositions over colliding pre-images (i.e. pre-images that map to the same image). For d-SCS, a first difference with respect to the standard Simon's problem is that the oracle doesn't provide direct access to the Simon function g. Instead, consider three functions: a random 2-to-1 function, f, a function p which maps colliding pairs of f to those of g and the inverse of p. Since f and g are both 2-to-1, p is a bijection and its inverse is well defined. If one is given access to all three functions, then evidently, the problem is equivalent to that of Simon's: the quantum algorithm would create superpositions over colliding pairs of f and then *coherently* evaluate the bijection together with its inverse, to obtain superpositions over colliding pairs of g. To obtain the desired separation—a problem that QC_d can solve but CQ_d cannot—we only allow restricted access to the bijection p. This restriction, denoted p', may be seen as a form of encrypted access to p. Specifically, to evaluate the bijection (or its inverse) p on a colliding pair (x_0, x_1) of f, i.e. $f(x_0) = f(x_1) = y$, p' additionally takes a "key" $h(y)$ as input. Access to the function h,which produces the key $h(y)$, is also restricted. It is given via a *shuffler* Ξ—an encoding which ensures that at least depth d is required to evaluate h.

So far, given access to f, p' (which encodes p) and Ξ (which encodes h), the goal is to find s (the period of g). It is easy to see that a QC_4 algorithm can solve this problem. The algorithm first creates equal superpositions of colliding pairs for f as in Simon's algorithm. The measured image register, which now contains the classical value $f(x_0) = f(x_1) = y$, can be used to evaluate $h(y)$ via Ξ by expending d classical depth. The QC_4 circuit can perform this computation—it can perform poly depth classical computation while maintaining quantum coherence. With $h(y)$ known, in a second quantum layer, the bijection (and its inverse) p is evaluated, via p', on the superposition of colliding pairs $|x_0\rangle + |x_1\rangle$ to obtain $|p(x_0)\rangle + |p(x_1)\rangle$ (up to normalisation). Recalling that $(p(x_0), p(x_1))$ are pre-images of the Simon function g, it only remains to make Hadamard measurements to obtain the system of linear equations which yield the secret s.

Why can't a CQ_d algorithm also solve this problem? By making the oracle provide access to a generic 2-to-1 function, f, instead of the Simon function, we made it so that superpositions over colliding pairs are no longer related by the period s. Indeed, the only way to obtain any information about s is to query the bijection. But in the case of CQ_d the quantum subroutines must measure their states completely before invoking the classical subroutines. This means that the quantum subroutines essentially obtain no information about s. We also know that only the classical subroutines can obtain access to the bijection as the shuffler can only be invoked by a circuit of depth at least d. In essence, we've made it so that if a CQ_d algorithm could solve the problem with a polynomial number of oracle queries, then Simon's problem could also be solved classically with a polynomial number of queries, which we know is impossible.

Formalising the above intuition turns out to be surprisingly involved. This is due to the following subtlety: the classical subroutines of CQ_d can obtain some information about g by querying the bijection (essentially obtaining evaluations of g) and one would have to show that this information is insufficient for recovering s even when invoking short depth quantum subroutines. We overcome this barrier by using a stochastic oracle in our proof but nevertheless conjecture that the result should also hold in the standard oracle setting. Thus, our final modification to Simon's problem, leading to the definition of d-SCS is to make it so that the oracle does not even provide direct access to f but instead, given a single bit as input, performs the following mappings:

$$0 \xrightarrow{S} (x_0, f(x_0)), \quad 1 \xrightarrow{S} (x_1, f(x_1)) \tag{2}$$

where (x_0, x_1) is a randomly chosen colliding pair of g. In other words, the oracle picks a random colliding pair for g and outputs either the first element in the pair or the second, depending on whether the input was 0 or 1. In the quantum setting, this translates to

$$|0\rangle_\text{B} |0\rangle_\text{X} |0\rangle_\text{Y} \xrightarrow{S} |0\rangle_\text{B} |x_0\rangle_\text{X} |f(x_0)\rangle_\text{Y} \tag{3}$$

$$|1\rangle_\text{B} |0\rangle_\text{X} |0\rangle_\text{Y} \xrightarrow{S} |1\rangle_\text{B} |x_1\rangle_\text{X} |f(x_1)\rangle_\text{Y} \tag{4}$$

We can therefore see that if the input qubit is in an equal superposition of $|0\rangle$and $|1\rangle$ (while all other registers are initialised as $|0\rangle$) we would obtain the state

$$\frac{1}{\sqrt{2}}(|0\rangle_B |x_0\rangle_X + |1\rangle_B |x_1\rangle_X) |f(x_0) = f(x_1)\rangle_Y \tag{5}$$

This is the same type of state as that of Equation 1, except the superposition is over pre-images of f rather than the Simon function g. The oracle will still provide access to the bijection and the shuffler in the manner explained above, so that the stochastic access to f does not change the fact that a QC_4 algorithm is still able to solve the problem (one is still able to map these states to superpositions over colliding pairs of g). But this stochastic access further restricts what a CQ_d algorithm can do, as now even if the classical subroutines are used to obtain evaluations of g, the stochastic nature of the oracle guarantees that these evaluations are uniformly random points. With this restriction in place, one can then show that CQ_d cannot solve the problem unless Simon's problem can be solved classically with polynomially-many queries.

1.2 Overview of the techniques

We begin with sketching the idea behind the hardness of d-SeS for QC_d circuits. To this end, we first describe the d-SeS problem in some more detail.

The d-Serial Simon's Problem (informal). Sample $d + 1$ random Simon's functions $\{f_i\}_{i=0}^d$ with periods $\{s_i\}_{i=0}^d$. The problem is to find the period, s_d, of the last Simon's function. However, only access to f_0 is given directly. Access to f_i, for $i \geq 1$, is given via a function L_{f_i} which outputs $f_i(x)$ if the input is (s_{i-1}, x) and $\perp$ otherwise, i.e. to access the ith Simon's function, one needs the period of the $(i-1)$th Simon's function (see Definition 26 and Definition 27 for details).

The lower bound technique. We now sketch the idea behind the proof that QC_d circuits solve d-SeS with at most negligible probability (see Section 4). Denote the initial set of oracles by $\mathcal{L}$. Let $\mathcal{M}_i$, for $1 \leq i \leq d$, be identical to $\mathcal{L}$, except the oracles $i, \ldots d$ output $\perp$ at all inputs (see Figure 2). Using a hybrid argument, we show that the d-depth quantum circuit $C := U_{d+1} \circ \mathcal{L} \circ U_d \ldots \mathcal{L} \circ U_1$ behaves essentially like $U_{d+1} \circ \mathcal{M}_d \circ U_d \ldots \mathcal{M}_1 U_1$. The intuition is that since all s_i are unknown before the oracles are invoked, the domain at which the oracles respond non-trivially (i.e. $\neq \perp$) is hard to find. After $\mathcal{M}_{i-1}$ is invoked, the "non-trivial domain" of the ith oracle can be learnt. It is therefore exposed in $\mathcal{M}_i$. Evidently, since $\mathcal{M}_1, \ldots \mathcal{M}_d$ do not contain any information about f_d, we conclude the d-depth quantum circuit C cannot solve d-SeS with non-negligible probability. The conclusion continues to hold even when efficient classical computation is allowed between the unitaries. At a high level, this is because one can condition on the classical queries yielding $\perp$ and show that this happens with high probability. Since only polynomially many queries are possible, the aforementioned analysis goes through largely unchanged. This proof is quite straightforward and consequently also serves as a considerable simplification of the proof of the QC_d depth hierarchy theorem proved by CCL.

We now turn to the ideas behind the hardness of the d-SCS problem for CQ_d circuits. We describe the d-SCS problem in some detail and to this end, first introduce the d-Shuffler construction[2] which is used for enforcing statements like "k is a function which requires depth larger than d to evaluate".

The d-Shuffler (informal). Intuitively, a d-Shuffler is an oracle which encodes a function, say $f : \{0,1\}^n \rightarrow \{0,1\}^n$, in such a way that one needs to make d sequential calls to it, to access f. Consider d random permutations, $f'_0, f'_1 \ldots f'_{d-1}$, from $\{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$. Define f'_d to be such that $f'_d(f'_{d-1} \ldots f'_0(x)) \ldots) = f(x)$ for $x \in \{0,1\}^n$. Define f_i, for $0 \leq i \leq d$, to be $f'_i(f'_{i-1}(\ldots f'_0(x) \ldots))$ for $x \in \{0,1\}^n$ and $\perp$ otherwise. We say $(f_i)_{i=0}^d$ is the d-Shuffler encoding f. We also use Ξ to abstractly refer to it. Using an argument, similar to the one above, one can show that no d-depth quantum circuit can access f via a d-Shuffler, with non-negligible probability (see Definition 36).

The d-Shuffled Collisions-to-Simon's Problem (informal). Consider the following functions on n-bit functions. Uniformly sample f from all 2-to-1 functions, g from all Simon's functions, and h from all 1-to-1 functions. Let p be some canonical bijection which maps colliding pairs of f to those of g (and p_{inv} be the inverse).

---

[2]This is a close variant of the d-Shuffling technique introduced by CCL.
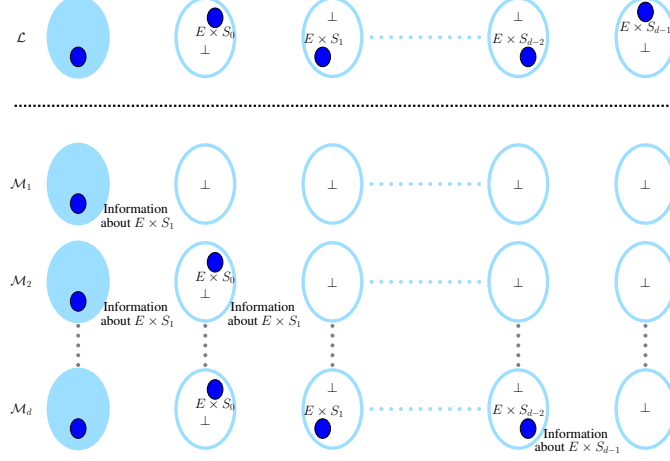
Figure 2: Shadows for QNC_d. The ovals represent the query domains of the various sub-oracles. Shaded regions denote a non-$\perp$ response. Let $\bar{S}_i$ denote the query domain at which $\mathcal{L}$ and $\mathcal{M}_i$ differ. The construction ensures that $\mathcal{M}_1, \ldots \mathcal{M}_{i-1}$ contain no information about $\bar{S}_i$. Convention: The left-most sub-oracles are numbered zero (and they respond with non-$\perp$ on $\{0,1\}^n$).

Let p' be such that $p'(h(f(x)), x) = p(x)$ and $\perp$ if the input is not of that form (p'_{inv} is similarly defined). Let Ξ be a d-Shuffler encoding h. The problem is, given access to $\mathcal{S}$ (a stochastic oracle, encoding f as in Equation (2)), p', p'_{inv} and Ξ, find the period s of g.

The lower bound technique. We briefly outline the broad argument for why CQ_d circuits can solve d-SCS with at most negligible probability (see Section 8). Consider the first classical circuit: It has vanishing probability of learning a collision from $\mathcal{S}$ (since y is chosen stochastically). So, with overwhelming probability, it can learn at most polynomially many values of p, p_{inv}, h and non colliding values of f.

For the subsequent quantum circuit, we expose those values in the shadows of p', p'_{inv} and Ξ (a d-Shuffler which encodes h). Here shadows are the analogues of $\mathcal{M}_i$ introduced above. For $\mathcal{S}$, we condition on never seeing the ys which already appeared (this will happen with overwhelming probability). Without access to h for the new y values (since access to h is via Ξ which requires at least depth d), the circuit cannot distinguish the shadows of p', p'_{inv} from the originals; the shadows contain no information about s (informally, they could contain z or $z \oplus s$ but not both). Finally, since f is a random two-to-one function, and since it is known that finding collisions even when given direct oracle access to f is hard, the quantum output cannot contain collisions (with non-negligible probability).

The next classical circuit, did not learn of any collisions from the quantum output. Further, it did not learn anything about the function, other than those which evaluate to ys which appeared in the previous step. Conditioned on the same ys not appearing (which happens with overwhelming probability), we can repeat the reasoning from the first step, polynomially many times.

In the proof, we first show that one can replace the oracles with their shadows, without changing the outputs of the circuits noticeably. We then show that if no collisions are revealed, the shadows contain no information about s and that collisions are revealed with negligible probability. For the full details of the proofs we refer the reader to Section 8.

1.3 Organisation

We begin with introducing the precise models of computation we consider in Section 2, followed by brief statements of the basic results in query complexity which we use, in Section 3. Our first main result—that d-Serial Simon's Problem can be solved using CQ_1 while it is hard for QC_d—is established in Section 4. The second main result, however, requires more ground work. We first study the simpler aspects of the d-Shuffled Simon's problem in Section 5 where we introduce the d-Shuffler (a close variant of the d-Shuffling technique introduced by CCL). In Section 6 we generalise the so-called "sampling argument" as formulated by CCL, which may be of independent interest. It is a technique used for establishing the continued utility of d-Shufflers despite repeated applications. We

give a new proof of hardness of d-SS for CQ_d circuits in Section 7 and use this as an intermediate step for the proof of our second main result which is delineated in Section 8. This final section begins with formalising intrinsically stochastic oracles, defines the d-SCS problem precisely, gives the algorithms for establishing the upper bounds and concludes with the proof hardness for CQ_d.

1.4 Relation to Prior Work

Four connections with [CCL20] are noteworthy.

First, as was already noted, we recover the depth hierarchy for QC_d with less effort. This was done by circumventing the use of the d-Shuffling and the "Russian nesting doll" technique employed by CCL. Our result is, in fact, slightly stronger too since if we allow Hadamard measurements in the QC_d model, our hierarchy theorem becomes optimal, i.e. it separates d and $d + 1$ depth. This was one of their open questions, as their separation was between d and $2d + 1$, even if we allow Hadamard measurements.[3]

Second, we give a new proof of hardness of d-SS for CQ_d. This proof is arguably simpler in that while it uses ideas related to d-Shuffling, it does not rely on the "Russian nesting doll" technique.[4] Aside from simplicity, the information required to specify the oracles in the proof by CCL scaled exponentially with d while in our proof, there is no dependence on d.

Third, the connection between the "sampling argument", the d-Shuffler and d-Shuffling. We show that the "sampling argument" as used by CCL holds quite generally and in doing so, obtain a proof which relies on very simple properties of the underlying distribution. We also show a composition result which allows us to repeatedly use the argument. We use this generalisation to establish the utility of the d-Shuffler.

Fourth, we also get a hierarchy theorem for CQ_d based on the hardness of the d-SCS problem. While this hierarchy theorem holds in the stochastic oracle setting and is thus not as strong as that obtained using d-SS (or d-Shuffling Simon's problem) which holds in the standard oracle setting, nevertheless, it yields a finer separation—it separates d from $d + 5$.

1.5 Conclusion and Outlook

In addition to resolving the open problems left by the works of Chia, Chung and Lai [CCL20] and Coudron and Menda [CM20], our results provide additional insights into the relations between different models of hybrid classical-quantum computation and sharpen our understanding of these models. Specifically, we find that QC_d and CQ_d solve incomparable sets of problems. Our results relativise and so one should obtain a similar conclusion about a hierarchy of analogous classes ($QCQC..QC_d$, $CQC..CQ_d$ etc). While an important open problem is whether the stochastic oracle in our second construction can be replaced with a standard oracle, we speculate that other classes which have resisted separation in the standard oracle model, may also admit separations via stochastic oracles.

A potentially interesting application that is hinted by our work, assuming the oracles can be instantiated with cryptographic primitives, is that of *tests of coherent quantum control*. In other words, the idea is to design a task (or a test) which can be solved by a quantum device having coherent control (the ability to adapt the gates it will perform based on the measurement results) but which cannot be solved by a device without coherent control (one which performs a measurement of all its qubits when providing a response). This is related to the recent notion of *proofs of quantumness* [Bra+18; Bra+20]. This is a test which can be passed by a BQP machine but not by a BPP one, assuming the intractability of some cryptographic task. In a test of coherent control, the tester (or verifier) has the ability to check for a more fine-grained notion of "quantumness" on the part of the quantum device (prover) and not just its ability to solve, say, BQP problems.

1.6 Acknowledgements

---

[3]Without the Hadamard measurements, we believe their upper bound becomes $2d + 2$.

[4]We say arguably because in proving the basic properties of the d-Shuffler, one needs more notation. The advantage is that, once done, this does not interfere with how it is used.

2 Models of Computation

We begin with defining the computational models we study in this work.

Notation 3. A *single layer unitary*, is defined by a set of one and two-qubit gates which act on disjoint qubits (so that they can all act parallelly in a single step). The number of layers in a circuit defines its *depth*.

Notation 4. A promise problem $\mathcal{P}$ is denoted by a tuple $(\mathcal{P}_0, \mathcal{P}_1)$ where $\mathcal{P}_0$ and $\mathcal{P}_1$ are subsets of $\{0,1\}^*$ satisfying $\mathcal{P}_0 \cap \mathcal{P}_1 = \emptyset$. It is not necessary that $\mathcal{P}_0 \cup \mathcal{P}_1 = \{0,1\}^*$.

Definition 5 (QNC_d circuits and BQNC_d languages)**.** Denote by QNC_d the set of d-depth quantum circuits (see Figure 1a).

Define BQNC_d to be the set of all promise problems $\mathcal{P} = (\mathcal{P}_0, \mathcal{P}_1)$ which satisfy the following: for each problem $\mathcal{P} \in \text{BQNC}_d$, there is a circuit family $\{C_n : C_n \in \text{QNC}_d$ and acts on $\text{poly}(n)$ qubits$\}$ and for all $x \in \mathcal{P}_1$, the circuit $C_{|x|}$ accepts with probability at least 2/3rds and for all $x \in \mathcal{P}_0$, the circuit $C_{|x|}$ accepts with probability at most 1/3rd.

Definition 6 (QC_d circuits and $\text{BQNC}_d^{\text{BPP}}$ languages)**.** Denote by QC_d the set of all circuits which, for each $n \in \mathbb{N}$, act on $\text{poly}(n)$ qubits and bits and can be specified by

- d single layered unitaries, $U_1, U_2 \dots U_d$,

- $\text{poly}(n)$ sized classical circuits $\mathcal{A}_{c,1} \dots \mathcal{A}_{c,d}, \mathcal{A}_{c,d+1}$, and

- d computational basis measurements

that are connected as in Figure 1b.

Define $\text{BQNC}_d^{\text{BPP}}$ to be the set of all promise problems $\mathcal{P} = (\mathcal{P}_0, \mathcal{P}_1)$ which satisfy the following: for each problem $\mathcal{P} \in \text{BQNC}_d^{\text{BPP}}$, there exists a circuit family $\{C_n : C_n \in \text{QC}_d$ and acts on $\text{poly}(n)$ qubits and bits$\}$ and for all $x \in \mathcal{P}_1$, the circuit $C_{|x|}$ accepts with probability at least 2/3rds and for all $x \in \mathcal{P}_0$, the circuit $C_{|x|}$ accepts with probability at most 1/3rd.

Finally, for $d = \text{poly}(\log(n))$, denote the set of languages by BQNC^{BPP}.

Definition 7 (CQ_d circuits and $\text{BPP}^{\text{QNC}_d}$ languages)**.** Denote by CQ_d the set of all circuits which, for each $n \in \mathbb{N}$ and $m = \text{poly}(n)$, act on $\text{poly}(n)$ qubits and bits and can be specified by

- m tuples of d single layered unitaries $(U_{1,i}, U_{2,i} \dots U_{d,i})_{i=1}^m$,

- $m + 1$, $\text{poly}(n)$ sized classical circuits $\mathcal{A}_{c,1} \dots \mathcal{A}_{c,m}, \mathcal{A}_{c,m+1}$, and

- m computational basis measurements

that are connected as in Figure 1c.

Define, as above, $\text{BPP}^{\text{BQNC}_d}$ to be the set of all promise problems $\mathcal{P} = (\mathcal{P}_0, \mathcal{P}_1)$ which satisfy the following: for each problem $\mathcal{P} \in \text{BPP}^{\text{BQNC}_d}$, there exists a circuit family $\{C_n : C_n \in \text{CQ}_d$ and acts on $\text{poly}(n)$ qubits and bits$\}$ and for all $x \in \mathcal{P}_1$, the circuit $C_{|x|}$ accepts with probability at least 2/3rds and for all $x \in \mathcal{P}_0$, the circuit $C_{|x|}$ accepts with probability at most 1/3rd.

Finally, for $d = \text{poly}(\log(n))$, denote the set of languages by BPP^{QNC}.

Remark 8. Connection with the more standard notation: QNC_d has depth d and QNC^m has depth $\log^m(n)$, i.e. $\text{QNC}^m = \text{QNC}_{\log^m(n)}$.

Later, it would be useful to symbolically represent these three circuit models but we mention them here for ease of reference.

Notation 9. We use the following notation for probabilities, QNC_d, QC_d and CQ_d circuits.

- Probability: The probability of an event E occurring, when process P happens, is denoted by $\Pr[E : P]$. In our context, the probability of a random variable X taking the value x when process Y takes place is denoted by $\Pr[x \leftarrow X : Y]$. When the process Y is just a sampling of X, we drop the Y and use $\Pr[x \leftarrow X]$.

- QNC$_d$: We denote a d-depth quantum circuit (see Definition 6 and Figure 1a) by $\mathcal{A} = U_d \circ \cdots \circ U_1$ and (by a slight abuse of notation) the probability that running the algorithm on all zero inputs yields x, by $\Pr[x \leftarrow \mathcal{A}]$ while that on some input state ρ by $\Pr[x \leftarrow \mathcal{A}(\rho)]$.

- QC$_d$: We denote a QC$_d$ circuit (see Definition 6 and Figure 1b) by $\mathcal{B} = \mathcal{A}_{c,d+1} \circ \mathcal{B}_d \circ \mathcal{B}_{d-1} \cdots \circ \mathcal{B}_1$ where $\mathcal{B}_i := \Pi_i \circ U_i \circ \mathcal{A}_{c,i}$ and "$\circ$" implicitly denotes the composition as shown in Figure 1b. As above, the probability of running the circuit $\mathcal{A}$ on all zero inputs and obtaining output x is denoted by $\Pr[x \leftarrow \mathcal{B}]$ while that on some input state ρ by $\Pr[x \leftarrow \mathcal{B}(\rho)]$.

- CQ$_d$: We denote a CQ$_d$ circuit (see Definition 7 and Figure 1c) by $C = \mathcal{A}_{c,m+1} \circ C_m \circ \cdots \circ C_1$ where $C_i := \Pi_i \circ U_{d,i} \circ \cdots \circ U_{1,i} \circ \mathcal{A}_{c,i}$ and "$\circ$" implicitly denotes the composition as shown in Figure 1c. Again, the probability of running the circuit C on all zero inputs and obtaining output x is denoted by $\Pr[x \leftarrow C]$ while that on some input state ρ by $\Pr[x \leftarrow C(\rho)]$.

2.1 The Oracle Versions

We consider the standard Oracle/query model corresponding to functions—the oracle returns the value of the function when invoked classically and its action is extended by linearity when it is accessed quantumly. The following also extends to stochastic oracles which are introduced later in Subsection 8.1 when we need them.

Notation 10. An oracle O_f corresponding to a function f is given by its action on "query" and "response" registers as $O_f |x\rangle_Q |a\rangle_R = |x\rangle_Q |a \oplus f(x)\rangle_R$. An oracle $O_{(f_i)_{i=1}^k}$ corresponding to multiple functions $f_1, f_2 \ldots f_k$ is given by $O_{(f_i)_{i=1}^k} |x_1, x_2 \ldots x_k\rangle_Q |a_1, a_2 \ldots a_k\rangle_R = |x_1, x_2 \ldots x_k\rangle_Q |a_1 \oplus f_1(x_1), a_2 \oplus f_2(x_2), \ldots a_k \oplus f_k(x_k)\rangle_R$.
We overload the notation; when O_f is accessed classically, we use $O_f(x)$ to mean it returns $f(x)$.

Remark 11 (QNC$_d^O$, QC$_d^O$, CQ$_d^O$). The oracle versions of QNC$_d$, QC$_d$ and CQ$_d$ circuits are as shown in Figure 3a, Figure 3b and Figure 3c. We allow (polynomially many) parallel uses of the oracle even though in the figures we represent these using single oracles. We do make minor changes to the circuit models, following [CCL20] when we consider QNC$_d$ circuits and CQ$_d$ circuits—an extra single layered unitary is allowed to process the final oracle call.

We end by explicitly augmenting Notation 9 to include oracles.

Notation 12. When oracles are introduced, we use the following notation.

- QNC$_d^O$: $\mathcal{A}^O = U_{d+1} \circ O \circ U_d \circ \ldots O \circ U_1$ (see Figure 3a)

- QC$_d^O$: $\mathcal{B}^O = \mathcal{A}_{c,d+1}^O \circ \mathcal{B}_d^O \circ \ldots \mathcal{B}_1^O$ where $\mathcal{B}_i^O = \Pi_i \circ O \circ U_i \circ \mathcal{A}_{c,i}^O$ and $\mathcal{A}_{c,i}^O$ can access O classically (see Figure 3b).

- CQ$_d^O$: $C^O = \mathcal{A}_{m+1}^O \circ C_m^O \circ \ldots C_1^O$ where $C_i^O := \Pi_i \circ U_{d+1,i} \circ O \circ U_{d,i} \circ \cdots \circ O \circ U_{1,i} \circ \mathcal{A}_{c,i}^O$ where $\mathcal{A}_{c,i}^O$ can access O classically (see Figure 3c).

The classes $\left(\mathrm{BQNC}_d^{\mathrm{BPP}}\right)^O$ and $\left(\mathrm{BPP}^{\mathrm{BQNC}_d}\right)^O$ are implicitly defined to be the query analogues of $\mathrm{BQNC}_d^{\mathrm{BPP}}$ and $\mathrm{BPP}^{\mathrm{BQNC}_d}$ (resp.), i.e. class of promise problems solved by QC$_d^O$ and CQ$_d^O$ circuits (resp.).

3 (Known) Technical Results I

The basic result we use, following Chia, Chung and Lai [CCL20], is a simplified version of the so-called "one-way to hiding", O2H lemma, introduced by Ambainis, Hamburg and Unruh [AHU18]. Informally, the lemma says the following: suppose there are two oracles O and Q which behave identically on all inputs except some subset S of their input domain. Let $\mathcal{A}^O$ and $\mathcal{A}^Q$ be identical quantum algorithms, except for their oracle access, which is to O and Q respectively. Then, the probability that the result of $\mathcal{A}^O$ and $\mathcal{A}^Q$ will be distinct, is bounded by the probability of finding the set S. We suppress the details of the general finding procedure and only focus on the case of interest for us here.
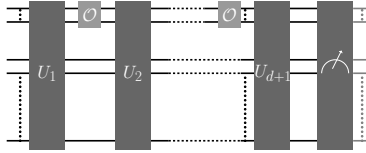
(a) A QNC$_d$ circuit with access to oracle O. Following [CCL20], in the oracle version of QNC$_d$, we allow it to perform one extra single layered unitary to process the output.
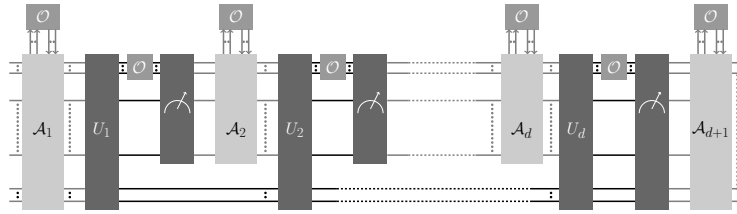
(b) A QC$_d$ circuit with access to an oracle O. There is no "extra" single layered unitary in this model.
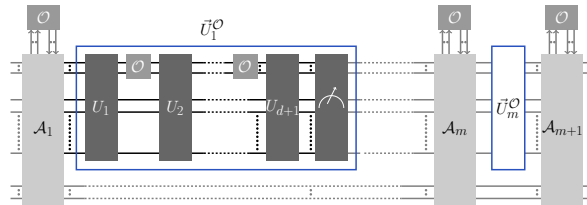
(c) A CQ$_d$ circuit with access to an oracle O. Again, following [CCL20], we allow an extra single layer unitary to process the result of the last oracle call.

Figure 3: The same three circuit models, but with oracle access.

3.1 Standard notions of distances

We quickly recall some notions of distances that appear here.

Definition 13. Let ρ, ρ' be two mixed states. Then we define

- Fidelity: $F(\rho, \rho') := \text{tr}(\sqrt{\sqrt{\rho}\rho'\sqrt{\rho}})$

- Trace Distance: $\text{TD}(\rho, \rho') := \frac{1}{2}\text{tr}\,|\rho - \rho'|$ and

- Bures Distance: $B(\rho, \rho') := \sqrt{2 - 2F(\rho, \rho')}$.

Fact 14. *For any string s, any two mixed states, ρ and ρ', and any quantum algorithm $\mathcal{A}$, we have*

$$|\Pr[s \leftarrow \mathcal{A}(\rho)] - \Pr[s \leftarrow \mathcal{A}(\rho')]| \leq B(\rho, \rho').$$

3.2 The O2H lemma

Notation 15. In the following, we treat W as the workspace register for our algorithm which is left untouched by the Oracle and recall that Q represents the query register and R represents the response register.

In fact, we wish to allow parallel access to $\boldsymbol{Q}$ and we represent this by allowing queries to a tuple of inputs, $\boldsymbol{q} = (q_1, q_2 \ldots q_{\text{poly}(n)})$ simultaneously. We use boldface to represent such tuples. The query register in this case is denoted by $\boldsymbol{Q}$.

Definition 16 ($U^{\mathcal{L}\backslash S}$). Suppose U acts on QRW, $\mathcal{L}$ is an oracle that acts on $\boldsymbol{QR}$ and S is a subset of the query domain of $\mathcal{L}$. We define

$$U^{\mathcal{L}\backslash S}|\psi\rangle_{QRW}|0\rangle_B := \mathcal{L}U_S U|\psi\rangle_{QRW}|0\rangle_B$$

where B is a qubit register, and U_S flips qubit B if any query is made inside the set S, i.e.

$$U_S|\boldsymbol{q}\rangle_{\boldsymbol{Q}}|b\rangle_B := \begin{cases} U_S|\boldsymbol{q}\rangle_{\boldsymbol{Q}}|b\rangle_B & \text{if } \boldsymbol{q} \cap S = \emptyset \\ U_S|\boldsymbol{q}\rangle_{\boldsymbol{Q}}|b \oplus 1\rangle_B & \text{otherwise.} \end{cases}$$

Here we treat $\boldsymbol{q}$ as a set when we write $\boldsymbol{q} \cap S$.

For notational simplicity, in the following, we drop the boldface for the query and response registers as they do not play an active role in the discussion.

Definition 17 ($\Pr[\text{find} : U^{\mathcal{L}\backslash S}, \rho]$). Let $U^{\mathcal{L}\backslash S}$ be as above and suppose $\rho \in \text{D}(QRWB)$. We define

$$\Pr[\text{find} : U^{\mathcal{L}\backslash S}, \rho] := \text{tr}[\mathbb{I}_{QRW} \otimes |1\rangle\langle 1|_B\, U^{\mathcal{L}\backslash S} \circ \rho].$$

This will depend on $\mathcal{L}$ and S. When $\mathcal{L}$ and S are random variables, we additionally take expectation over them.

Remark 18. Let $U^{\mathcal{L}\backslash S}$ be as in Definition 16 and let $|\psi\rangle \in QRW$. Note that we can always write

$$\mathcal{L}U|\psi\rangle_{QRW} = |\phi_0\rangle_{QRW} + |\phi_1\rangle_{QRW}$$

where $|\phi_0\rangle$ and $|\phi_1\rangle$ contains queries outside S and inside S respectively, i.e. $\langle\phi_0|\phi_1\rangle = 0$. Further, we can write

$$U^{\mathcal{L}\backslash S}|\psi\rangle_{QRW}|0\rangle_B = |\phi_0\rangle_{QRW}|0\rangle_B + |\phi_1\rangle_{QRW}|1\rangle_B.$$

Lemma 19 ([CCL20; AHU18] O2H)**.** *Let*

- $\mathcal{L}$ *be an oracle which acts on QR and S be a subset of the query domain of $\mathcal{L}$,*

- $\mathcal{G}$ *be a shadow of $\mathcal{L}$ with respect to S, i.e. $\mathcal{G}$ and $\mathcal{L}$ behave identically for all queries outside S,*

- *further, suppose that within S, $\mathcal{G}$ responds with $\perp$ while (again within S), $\mathcal{L}$ does not respond with $\perp$. Finally, let Π_t be a measurement in the computational basis, corresponding to the string t.*

Then

$$|\text{tr}[\Pi_t \mathcal{L} \circ U \circ \rho] - \text{tr}[\Pi_t \mathcal{G} \circ U \circ \rho]| \le B(\mathcal{L} \circ U \circ \rho, \mathcal{G} \circ U \circ \rho)$$

$$\le \sqrt{2 \Pr[\text{find} : U^{\mathcal{L} \backslash S}, \rho]}.$$

If $\mathcal{L}$ and S are random variables with a joint distribution, we take the expectation over them in the RHS (see Definition 17).

Proof. We begin by assuming that $\mathcal{L}$ and S are fixed (and so is $\mathcal{G}$). In that case, we can assume ρ is pure. If not, we can purify it and absorb it in the work register. (The general case should follow from concavity). From Remark 18, we have

$$|\psi_L\rangle := \mathcal{L}U |\psi\rangle_{Q'} \overset{18}{=} |\phi_0\rangle_{Q'} + |\phi_1\rangle_{Q'}.$$
$$\mathcal{L}U_S U |\psi\rangle_{Q'} |0\rangle_B = |\phi_0\rangle_{Q'} |0\rangle_B + |\phi_1\rangle_{Q'} |1\rangle_B$$

where Q' is a shorthand for QRW. Similarly let

$$|\psi_G\rangle := \mathcal{G}U |\psi\rangle_{Q'} = |\phi_0\rangle_{Q'} + |\phi_1^\perp\rangle_{Q'}$$

where note that

$$\langle \phi_1 | \phi_1^\perp \rangle_{QRW} = 0 \tag{6}$$

because $|\phi_1\rangle$ and $|\phi_1^\perp\rangle$ are the states where the queries were made on S, and on S $\mathcal{G}$ responds with $\perp$ while $\mathcal{L}$ does not. Further, we analogously have

$$\mathcal{G}U_S U |\psi\rangle_{Q'} |0\rangle_B = |\phi_0\rangle_{Q'} |0\rangle_B + |\phi_1^\perp\rangle_{Q'} |1\rangle_B.$$

We show that the difference between $|\psi_L\rangle$ and $|\psi_G\rangle$ is bounded by $P_{\text{find}}(\mathcal{L}, S) := \Pr[\text{find} : U^{\mathcal{L} \backslash S}, \rho]$, which in turn can be used to bound the quantity in the statement of the lemma.

$$\begin{aligned} \||\psi_L\rangle - |\psi_G\rangle\|^2 &= \||\phi_1\rangle - |\phi_1^\perp\rangle\|^2 \\ &\overset{(6)}{=} \||\phi_1\rangle\|^2 + \||\phi_1^\perp\rangle\|^2 \\ &= 2\||\phi_1\rangle\|^2 &&\because \||\phi_1\rangle\|^2 = \||\phi_1^\perp\rangle\|^2 = 1 - \||\phi_0\rangle\|^2 \\ &= 2P_{\text{find}}(\mathcal{L}, S). \end{aligned}$$

If $\mathcal{L}$ and S are random variables drawn from a (possibly) joint distribution $\Pr(\mathcal{L}, S)$, the analysis can be generalised as follows. Let

$$\rho_L := \sum_{\mathcal{L}, S} \Pr(\mathcal{L}, S) |\psi_L\rangle \langle\psi_L|$$
$$\rho_G := \sum_{\mathcal{L}, S} \Pr(\mathcal{L}, S) |\psi_G\rangle \langle\psi_G|$$

where $|\psi_G\rangle$ is fixed by $\mathcal{L}$ and S because G itself is fixed once $\mathcal{L}$ and S is fixed (by assumption). One can then use monotonicity of fidelity to obtain

$$\begin{aligned} F(\rho_L, \rho_G) &\ge \sum_{L,S} \Pr(\mathcal{L}, S) F(|\psi_L\rangle, |\psi_G\rangle) \\ &\ge 1 - \frac{1}{2} \cdot \sum_{L,S} \Pr(\mathcal{L}, S) \||\psi_L\rangle - |\psi_G\rangle\|^2 &&\because 1 - \frac{1}{2}F(|a\rangle, |b\rangle) \ge \||a\rangle - |b\rangle\|^2 \\ &\ge 1 - \frac{1}{2} \sum_{L,S} \Pr(\mathcal{L}, S) 2P_{\text{find}}(\mathcal{L}, S) \\ &= 1 - P_{\text{find}} \end{aligned}$$

where P_{find} is the expectation of $P_{\text{find}}(\mathcal{L}, S)$ over $\mathcal{L}$ and S. It is known that the trace distance bounds the LHS of the Lemma and the trace distance itself is bounded by $\sqrt{2 - 2F}$.

$\square$

In the following, we resume the use of boldface for the query and response registers as they do play an active role in the discussion.

Lemma 20 ([CCL20; AHU18] Bounding $\Pr[\text{find} : U^{\mathcal{L}\backslash S}, \rho]$). *Suppose S is a random variable and $\Pr[x \in S] \leq p$ for some p. Further, assume that U and ρ are uncorrelated[5] to S. Then, (see Definition 16)*

$$\Pr[\text{find} : U^{\mathcal{L}\backslash S}, \rho] \leq \bar{q} \cdot p$$

where $\bar{q}$ is the total number of queries U makes to $\mathcal{L}$.

Proof. Let us begin with the case where the oracle is applied only once, i.e. $\boldsymbol{Q}$ is a single query register Q. Since the RW registers don't play any significant role, we denote it by L. Let

$$U \ket{\psi} = \sum_{q,l} \psi(q,l) \ket{q,l,0}_{QLS}$$

$$\implies U_S U \ket{\psi} = \sum_{q \notin S} \left(\sum_{r,l} \psi(q,l) \ket{q,l}_{QL} \right) \ket{0}_S + \sum_{q \in S} \left(\sum_{r,l} \psi(q,l) \ket{q,l}_{QL} \right) \ket{1}_S .$$

Since $\mathcal{L}$ leaves registers QS unchanged,

$$\text{tr}[\mathbb{I}_{QL} \otimes \ket{1}\bra{1}_B (\mathcal{L} \circ U_S \circ U \circ \ket{\psi}\bra{\psi})] = \text{tr}[\mathbb{I}_{QL} \otimes \ket{1}\bra{1}_B (U_S \circ U \circ \ket{\psi}\bra{\psi})]$$
$$= \sum_q \psi^2(q) \chi_S(q)$$

where $\psi(q) = \sum_l \psi(q,l)$ and χ_S is the characteristic function for S, i.e.

$$\chi_S(q) = \begin{cases} 1 & q \in S \\ 0 & q \notin S. \end{cases}$$

We are yet to average over the random variable S. Clearly, $\mathbb{E}(\chi_S(q)) = \Pr[x \in S] \leq p$, yielding

$$\Pr[\text{find} : U^{\mathcal{L}\backslash S}, \rho] \leq p.$$

In the general case, everything goes through unchanged except the string q is now a set of strings $\boldsymbol{q}$ and

$$\chi_S(\boldsymbol{q}) = \begin{cases} 1 & \boldsymbol{q} \cap S \neq \emptyset \\ 0 & \boldsymbol{q} \cap S = \emptyset. \end{cases}$$

Consequently, one evaluates $\mathbb{E}(\chi_S(\boldsymbol{q})) = \Pr[\boldsymbol{q} \cap S \neq \emptyset] \leq |\boldsymbol{q}| \cdot p = \bar{q} \cdot p$, by the union bound, yielding

$$\Pr[\text{find} : U^{\mathcal{L}\backslash S}, \rho] \leq \bar{q} \cdot p.$$

$\square$

We now generalise the statement slightly to facilitate the use of conditional random variables. These become useful for the proof of hardness for QC_d circuits.

Corollary 21. *Let D be the query domain of $\mathcal{L}$. Suppose a set $S \subseteq D$, a quantum state ρ and a unitary U are drawn from a joint distribution (which may be correlated with $\mathcal{L}$). Let the set $T \subseteq D$ be another random variable (again, possibly arbitrarily correlated) and F be the event that[6] $S \cap T = \emptyset$. Define the random variables $\mathcal{N} := \mathcal{L}|F$, $R := S|F$, $\sigma := \rho|F$ and $V := U|F$ and assume that R, σ and V are uncorrelated.[7] Suppose for all $x \in D$, $\Pr[x \in R] \leq p$ for some p. Then, (see Definition 16)*

$$\Pr[\text{find} : V^{\mathcal{N}\backslash R}, \sigma] \leq \bar{q} \cdot p$$

where $\bar{q}$ is the total number of queries V makes to $\mathcal{L}$.

---

[5]i.e. the distribution from which S is sampled is uncorrelated to the distribution from which U and ρ are sampled,

[6]One could take F be to a general event as well but for our purposes, this suffices.

[7]i.e. the joint probability distribution of $S|T$, $\rho|T$ and $U|T$ is a product of their individual probability distributions.

4 d-Serial Simon's Problem « Main Result 1

In this section, we introduce the d-Serial Simon's Problem. To get some familiarity, we first state the upper bounds—we see that it can be solved by a QC_{d+1} circuit and also by a CQ_1 circuit. As for the lower bound, we begin by showing that no QNC_d circuit can solve the problem. We then extend this proof to show that no QC_d circuit can solve the problem either.

4.1 Oracles and Distributions | Simon's and c-Serial

We begin with Simon's problem. As will be the case for all problems we discuss, we consider both search and decision variants. The latter will usually take the form of distinguishing, say, a Simon's oracle from a one-to-one oracle. We thus, first define the two oracles, and then use them to define both variants of the problem.

Definition 22 (Distribution for one-to-one functions). Let $F := \{f : \{0,1\}^n \to \{0,1\}^n | f \text{ is one-to-one}\}$ be the set of all one-to-one functions acting on n-bit strings, and let $O_F := \{O_f : f \in F\}$ be the set of all oracles associated with these functions. Define the *distribution over one-to-one functions*, $\mathbb{F}_R(n)$, to be the uniform distribution over F and the *distribution for one-to-one function oracles*, $\mathbb{O}_R(n)$, to be the uniform distribution over O_F.

Definition 23 (Distribution for Simon's function). Let

$$F := \{(f,s)|f : \{0,1\}^n \to \{0,1\}^n \text{ is two-to-one and } f(x \oplus s) = f(x)\}$$

be the set of all Simon functions acting on n-bit strings and let $O_F := \{(O_f, s) : (f,s) \in F\}$ be the set of all oracles associated with these functions. Define the *distribution over Simon's functions*, $\mathbb{F}_S(n)$, to be the uniform distribution over F and the *distribution over Simon's Oracles*, $\mathbb{O}_S(n)$ to be the uniform distribution over O_F.

Definition 24 (Simon's Problem). Fix an integer $n > 0$ to denote the problem size. Let $\mathcal{R} \sim \mathbb{O}_R(n)$ be an oracle for a randomly chosen one-to-one function and $(\mathcal{S}, s) \sim \mathbb{O}_S(n)$ be an oracle for a randomly chosen Simon's function which has period s.

- Search version: Given $\mathcal{S}$, find s.

- Decision version: Given $O \in_R \{\mathcal{R}, \mathcal{S}\}$, i.e. one of the oracles with equal probability, determine which oracle was given.

While we focus on d-Serial Simon's Problem, we define the problem more generally as a d-Serial Generic Oracle Problem with respect a "generic oracle problem". To this end, we briefly formalise the latter.

Definition 25 (Generic Oracle Problem). Let $(O, r) \sim \mathbb{O}(n)$ where $\mathbb{O}$ is some fixed distribution over oracles and the corresponding expected answers, and n is the problem size. Let $\mathcal{R} \sim \varnothing(n)$ where suppose that $\varnothing(n)$ is the distribution over oracles against which the decision version is defined. The generic oracle problem is:

- Search version: Given O, find r.

- Decision version: Given $Q \in_R \{O, \mathcal{R}\}$, determine which oracle was given.

The d-Serial Generic Oracle problem is based on the following idea: there is a sequence of $d + 1$ oracles (indexed $0, 1 \ldots d$), of which the first d encode a Simon's problems. The zeroth oracle can be accessed directly. To access the first oracle, however, a secret key is needed. This secret is the period of the zeroth Simon's function. The first oracle, once unlocked, behaves as a Simon's oracle whose period unlocks the second oracle and so on. The Generic oracle is unlocked by the period of the $(d-1)$th Simon's problem. The problem is to solve the "Generic Oracle problem" using the aforementioned $d + 1$ oracles.

The intuition is quite simple. Consider the d-Serial Simon's Oracle Problem, i.e. where the generic oracle problem is a Simon's problem. Then, observe that, naïvely, a QC_d scheme would have to use all its d depth to solve the d Simon's problems to access the last oracle and have no quantum depth left for solving the last Simon's problem. However, with one more depth, i.e. with QC_{d+1}, the problem can be solved. Further, note that a CQ_1 scheme too can solve the problem. We will revisit these statements shortly but first, we formally define the problem.

Definition 26 (*c*-Serial Generic Oracle). Suppose the Generic Oracle is sampled from the distributions $\mathbb{O}(n)$ and $\varnothing(n)$ as in Definition 25 above where the oracles' domain is assumed to be $\{0, 1\}^n$. We define the *c-Serial Generic Function distribution* $\mathbb{F}_{\text{Serial}}(c, n, \mathbb{O}, \varnothing)$ and the *c-Serial Generic Oracle distribution* $\mathbb{O}_{\text{Serial}}(c, n, \mathbb{O}, \varnothing)$ by specifying its sampling procedure.

- Sampling step:

    – For $i \in \{0, 1, 2, \ldots c - 1\}$, $(f_i, s_i) \sim \mathbb{F}_S(n)$, i.e. sample c Simon's functions from $\mathbb{F}_S(n)$.

    – Sample $(O, r) \sim \mathbb{O}(n)$ and $\mathcal{R} \sim \varnothing(n)$. For the search version, $Q := O$ while for the decision version, $Q \in_R \{O, \mathcal{R}\}$.

- Let $L_{f_i} : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}^n \cup \{\perp\}$ for $i \in \{0, 1, 2, \ldots c\}$ be defined as

    – $L_{f_0}(x, z) = f_0(x)$ when $i = 0$ and

    – $L_{f_i}(x, z) := \begin{cases} f_i(x) & z = s_{i-1} \\ \perp & z \neq s_{i-1} \end{cases}$ when $i \in \{1, 2, 3 \ldots c - 1\}$.

    – $L_{f_c}(x, z) := \begin{cases} Q(x) & z = s_{c-1} \\ \perp & z \neq s_{c-1}. \end{cases}$

- Let $\mathcal{L}$ be the oracle associated with $(L_{f_i})_{i=0}^c$.

- Returns

    – $\mathbb{O}_{\text{Serial}}$

        * Search: When $\mathbb{O}_{\text{Serial}}(c, n, \mathbb{O})$ is sampled, consider the search version of the sampling step above and return $(\mathcal{L}, r)$.

        * Decision: When $\mathbb{O}_{\text{Serial}}(c, n, \mathbb{O}, \varnothing)$ is sampled, consider the decision version of the sampling step above and return $(\mathcal{L}, l)$ where $l = 0$ if $Q = O$ and $l = 1$ if $Q = \mathcal{R}$.

    – $\mathbb{F}_{\text{Serial}}$

        * When $\mathbb{F}_{\text{Serial}}(c, n, \mathbb{O})$ is sampled, consider the search version of the sampling step above and return $(L_{f_0}, s_0, L_{f_1}, s_1 \ldots L_{f_{c-1}}, s_{c-1}, L_{f_c}, r)$, where recall that (O, r) were sampled from $\mathbb{O}(n)$ and $Q = O$ (for the search version of the sampling step).

Definition 27 (*c*−Serial Generic Oracle Problem). As before, we define two variants of the *c*-Serial Generic Oracle Problem:

- Search Variant: Let $(\mathcal{L}, r) \sim \mathbb{O}_{\text{Serial}}(c, n, \mathbb{O})$. Given $\mathcal{L}$, find r.

- Decision Variant: Let $(\mathcal{L}, l) \sim \mathbb{O}_{\text{Serial}}(c, n, \mathbb{O}, \varnothing)$. Given $\mathcal{L}$, determine whether $l = 0$ or 1.

4.2 Depth upper bounds for *d*-Serial Simon's Problem

One can use Simon's algorithm for easily obtaining the following depth upper bounds for solving *d*-Serial Simon's Problem. Note that while we only need $d + 1$ queries to the oracle, we need to apply Hadamards before and after, for Simon's algorithm to work. Thus, for QC circuits (which by definition allow only one layer of unitary before an oracle call, not after), we need depth $2d + 2$ while for CQ circuits (which allow a layer of unitary before and after), we only need depth 1.

Proposition 28. *The decision variant of the d-Serial Simon's Problem is in* $\text{BQNC}_{2d+2}^{\text{BPP}}$ *and the search version can be solved using* QC_{2d+2} *circuits.*

Proposition 29. *The decision variant of the d-Serial Simon's Problem is in* $\text{BPP}^{\text{BQNC}_1}$ *and the search version can be solved using* CQ_1 *circuits.*

4.3 Depth lower bounds for c-Serial Simon's problem

Following Chia, Chung and Lai [CCL20], it would be useful to define "shadows" of oracles for establishing lower bounds. The idea is simple. Given an oracle and a subset S of the query domain thereof, the shadow behaves exactly like the oracle when queried outside S and outputs $\perp$ when queried inside S. Since there are multiple functions involved, for notational ease, we formalise this notion for a sequence of sets.

Definition 30 (c–Serial Generic Shadow Oracle). Given

- $(L_{f_i}, s_i)_{i=0}^c$ from the sample space of $\mathbb{F}_{\text{Serial}}(c, n, \mathbb{O})$, and

- a tuple of subsets $\bar{S} = (S_1, S_2, \ldots S_c)$ where $S_i \subseteq \{0, 1\}^n \times \{0, 1\}^n$,

let $\mathcal{L}$ be the oracle associated with $(L_{f_i})_{i=1}^c$. Then, the c-Serial Generic Shadow Oracle (or simply the shadow) $\mathcal{H}$ for the oracle $\mathcal{L}$ with respect to $\bar{S}$ is defined to be the oracle associated with $(L_{f_0}, L'_{f_1}, L'_{f_2} \ldots L'_{f_c})$ where

$$L'_{f_i} := \begin{cases} L_{f_i}(x, z) & (x, z) \in \{0, 1\}^{2n} \backslash S_i \\ \perp & (x, z) \in S_i \end{cases}$$

for $i \in \{1, 2 \ldots c\}$.

Remark 31. For the moment, we exclusively consider the case where the "Generic Oracle" (drawn from $\mathbb{O}$) is also a Simon's oracle (which are drawn from $\mathbb{O}_S$) and define the *c-Serial Simon's Oracle/Problem/Shadow Oracle* implicitly.

As will become evident shortly, it would be useful to have shadows $\{\mathcal{M}_i\}_{i=1}^d$ such that $\mathcal{M}_j$ behaves like the d-Serial Simon's oracle $\mathcal{L}$ at all sub-oracles from 0 to $j - 1$ but outputs $\perp$ for all subsequent sub-oracles (see Figure 1a).

Algorithm 32 ($\bar{S}_j$ for QNC_d exclusion).
Input:

- $1 \leq j \leq d$ and

- $(L_{f_i}, s_i)_{i=0}^d$ from the sample space of $\mathbb{F}_{\text{Serial}}(c, n, \mathbb{O}_S)$,

Output: $\bar{S}_j$, a tuple of d subsets defined as

$$\bar{S}_j := \begin{cases} (\emptyset, \emptyset, \ldots \emptyset, E \times s_{j-1}, E \times s_j \ldots, E \times s_{d-1}) & \text{for } j > 1 \\ (E \times s_0, E \times s_1 \ldots, E \times s_{d-1}) & \text{for } j = 1 \end{cases}$$

where $E = \{0, 1\}^n$ and $\times$ is the Cartesian product.

4.3.1 Warm up | d-Serial Simon's Problem is hard for QNC_d

Theorem 33. *Let $(\mathcal{L}, s) \sim \mathbb{O}_{\text{Serial}}(d, n, \mathbb{O}_S)$, i.e. let $\mathcal{L}$ be an oracle for a random d-Serial Simon's Problem of size n and period s. Let $\mathcal{A}^{\mathcal{L}}$ be any d depth quantum circuit (see Definition 5 and Remark 11) acting on $O(n)$ qubits, with query access to $\mathcal{L}$. Then $\Pr[s \leftarrow \mathcal{A}^{\mathcal{L}}] \leq \text{negl}(n)$, i.e. the probability that the algorithm finds the period is exponentially small.*

Proof. Suppose[8] $(L_{f_i}, s_i)_{i=0}^d \sim \mathbb{F}_{\text{Serial}}(d, n, \mathbb{O}_S)$ (see Definition 26) and let $\mathcal{L}$ be the oracle associated with $(L_{f_i})_{i=0}^d$. For notational consistency, let $s = s_d$. Denote an arbitrary $\text{QNC}_d^{\mathcal{L}}$ circuit, $\mathcal{A}^{\mathcal{L}}$, by

$$\mathcal{A}^{\mathcal{L}} := \Pi \circ U_{d+1} \circ \mathcal{L} \circ U_d \ldots \mathcal{L} \circ U_2 \circ \mathcal{L} \circ U_1$$

and suppose Π corresponds to the algorithm outputting the string s. For each $i \in \{1, \ldots d\}$, construct the tuples $\bar{S}_i$ using Algorithm 32. Let $\mathcal{M}_i$ be the shadow of $\mathcal{L}$ with respect to $\bar{S}_i$ (see Definition 30). Define

$$\mathcal{A}^{\mathcal{M}} := \Pi \circ U_{d+1} \circ \mathcal{M}_d \circ U_d \ldots \mathcal{M}_2 \circ U_2 \circ \mathcal{M}_1 \circ U_1.$$

---

[8]This proof closely follows techniques from Chia, Chung and Lai [CCL20] but establishes a new result.

Note that $\Pr[s \leftarrow \mathcal{A}^{\mathcal{M}}] \leq \frac{1}{2^n}$ because no $\mathcal{M}_i$ contains any information about f_d, the last Simon's function, whose period, s, is the required solution. Thus, no algorithm can do better than making a random guess. We now show that the output distributions of $\mathcal{A}^{\mathcal{L}}$ and $\mathcal{A}^{\mathcal{M}}$ cannot be noticeably different using the O2H lemma (see Lemma 19).

To apply the lemma, one can use the hybrid method as follows (we drop the $\circ$ symbol for brevity):

$$\left| \Pr[s \leftarrow \mathcal{A}^{\mathcal{L}}] - \Pr[s \leftarrow \mathcal{A}^{\mathcal{M}}] \right|$$

$$= |\mathrm{tr}[\Pi U_{d+1} \mathcal{L} U_d \ldots \mathcal{L} U_2 \mathcal{L} U_1 \rho_0 - \Pi U_{d+1} \mathcal{M}_d U_d \ldots \mathcal{M}_2 U_2 \mathcal{M}_1 U_1 \rho_0]|$$

$$\leq \left| \mathrm{tr}[\Pi U_{d+1} \mathcal{L} U_d \ldots \mathcal{L} U_2 \underbrace{\mathcal{L} U_1 \rho_0}_{} - \Pi U_{d+1} \mathcal{L} U_d \ldots \mathcal{L} U_2 \underbrace{\mathcal{M}_1 U_1 \rho_0}_{}] \right| +$$

$$\left| \mathrm{tr}[\Pi U_{d+1} \mathcal{L} U_d \ldots U_3 \underbrace{\mathcal{L} U_2 \mathcal{M}_1 U_1 \rho_0}_{} - \Pi U_{d+1} \mathcal{L} U_d \ldots \mathcal{L} U_3 \underbrace{\mathcal{M}_2 U_2 \mathcal{M}_1 U_1 \rho_0}_{}] \right| +$$

$$\vdots$$

$$\left| \mathrm{tr}[\Pi U_{d+1} \underbrace{\mathcal{L} U_d \mathcal{M}_{d-1} U_{d-1} \ldots U_3 \mathcal{M}_2 U_2 \mathcal{M}_1 U_1 \rho_0}_{} - \Pi U_{d+1} \underbrace{\mathcal{M}_d U_d \mathcal{M}_{d-1} \ldots U_3 \mathcal{M}_2 U_2 \mathcal{M}_1 U_1 \rho_0}_{}] \right|$$

$$\leq \mathrm{B}(\mathcal{L} \circ U_1(\rho_0), \mathcal{M}_1 \circ U_1(\rho_0)) +$$
$$\mathrm{B}(\mathcal{L} \circ U_2(\rho_1), \mathcal{M}_2 \circ U_2(\rho_1)) +$$
$$\vdots$$
$$\mathrm{B}(\mathcal{L} \circ U_d(\rho_{d-1}), \mathcal{M}_d \circ U_d(\rho_{d-1}))$$

$$\leq \sum_{i=1}^{d} \sqrt{2 \Pr[\mathrm{find} : U_i^{\mathcal{L} \setminus \bar{S}_i}, \rho_{i-1}]}$$

where $\rho_0 = |0 \ldots 0\rangle \langle 0 \ldots 0|$ and $\rho_i = \mathcal{M}_i \circ U_i \circ \ldots \mathcal{M}_1 \circ U_1(\rho_0)$ for $i > 0$. To bound the last expression, we apply Lemma 20. To apply the lemma, however, we must ensure that the subset of queries at which $\mathcal{L}$ and $\mathcal{M}_i$ differ, i.e. $\bar{S}_i = (\emptyset, \ldots \emptyset, E \times s_{i-1}, E \times s_i \ldots E \times s_{d-1})$, (recall $E = \{0, 1\}^n$) is uncorrelated to U_i and ρ_{i-1}. Observe that ρ_{i-1} is completely uncorrelated to $s_{i-1}, s_i, s_{i+1} \ldots s_d$. At a high level, this is because ρ_{i-1} can at most access $\mathcal{M}_1, \ldots \mathcal{M}_{i-1}$ and these in turn contain no information about $\bar{S}_i$. To see this, note that even though to define $\mathcal{M}_1$, we used $\bar{S}_1 = (E \times s_0, E \times s_1 \ldots, E \times s_{d-1})$, still $\mathcal{M}_1$ contains no information about $s_1, \ldots s_d$ because (other than the zeroth sub-oracle which can reveal s_0) it always outputs $\perp$. Similarly, $\mathcal{M}_2$ contains information about s_0, s_1 but not about $s_2, s_3 \ldots s_d$. Analogously for $\mathcal{M}_3$ and so on. Since the definition of ρ_{i-1} only involves $\mathcal{M}_1, \ldots \mathcal{M}_{i-1}$, it contains no information about $s_{i-1}, s_i, s_{i+1} \ldots s_d$. As for U_i, that is uncorrelated to all s_is by construction.

Finally, to apply Lemma 20, we need to bound the probability that a fixed query, x, lands in the set $\bar{S}_i$. To this end, observe that $\Pr[x_i \in E \times s_i] = \frac{1}{2^n}$ when $s_i \in_R \{0, 1\}^n$ and that the union bound readily bounds the desired probability, i.e. $\Pr[x \in \bar{S}_i] \leq d \cdot 2^{-n}$. Since U acts on $\mathrm{poly}(n)$ many qubits, q in the lemma can be set to $\mathrm{poly}(n)$. Thus, we can bound the last inequality by $d \cdot \mathrm{poly}(n) / 2^n$. Using the triangle inequality, we get

$$\Pr[s \leftarrow \mathcal{A}^{\mathcal{L}}] \leq \frac{\mathrm{poly}(n)}{2^n}.$$

$\square$

4.3.2 d-Serial Simon's Problem is hard for QC_d

To prove our first main result—the same statement for QC_d—we need to account for the possibility that the classical algorithm can make $\mathrm{poly}(n)$ many queries and process these before applying the next quantum layer. The high-level intuition is quite simple. We follow essentially the same strategy as in the QNC_d case, except that we successively condition the distribution over $\mathcal{L}$ to exclude the cases where the classical algorithm obtains a non-$\perp$ output.

To be slightly more precise, fix a particular $\mathcal{L}$. When the classical algorithm queries locations T, it could either get all $\perp$ responses or not get all $\perp$ responses (that is some responses may be non-$\perp$). We treat the latter case as

though the classical algorithm solved the problem. For the former case, we conclude that the classical algorithm ruled out certain values of s_is. We condition on this event and proceed similarly with the remaining analysis.

Since $\mathcal{L}$ is actually a random variable, notice that the probability that $\mathcal{L}$ responds with non-$\perp$ for a given T, is at most $O(\text{poly}(n) \cdot 2^{-n})$. Thus, the probability that $\mathcal{L}$ responds with $\perp$ for all queries in T is essentially 1. Since we want an upper bound on the winning probability, we treat this conditional probability as 1 and in the subsequent analysis, use $\mathcal{L}|T$ where T is s.t. $\mathcal{L}(T) = \perp$. Since the shadows were defined using $\mathcal{L}$, they also get conditioned and the remaining analysis, essentially goes through unchanged. The only difference is that when $\Pr[\text{find} : U^{\mathcal{L}\backslash S}, \rho]$ is evaluated, because of the conditioning, the probabilities change by polynomial factors but these we have anyway been absorbing so the result remains unchanged.

Theorem 34. *Let $(\mathcal{L}, s) \sim \mathbb{O}_{\text{Serial}}(d, n, \mathbb{O}_S)$, i.e. let $\mathcal{L}$ be an oracle for a random d-Serial Simon's Problem of size n and period s. Let $\mathcal{B}^{\mathcal{L}}$ be any QC_d circuit (see Definition 6 and Remark 11) with query access to $\mathcal{L}$. Then, $\Pr[s \leftarrow \mathcal{B}^{\mathcal{L}}] \leq \text{negl}(n)$, i.e. the probability that the algorithm finds the period is exponentially small.*

Proof. The initial part of the proof is almost identical to that of the QNC_d case. Suppose $(L_{f_i}, s_i)_{i=0}^{d} \sim \mathbb{F}_{\text{Serial}}(d, n, \mathbb{O}_S)$ (see Definition 26) and let $\mathcal{L}$ be the oracle associated with $(L_{f_i})_{i=0}^{d}$. For notational consistency, let $s = s_d$. Recall that we denoted an arbitrary $\text{QC}_d^{\mathcal{L}}$ circuit (see Notation 9) with oracle access to $\mathcal{L}$,

$$\mathcal{B}^{\mathcal{L}} := \Pi \circ \mathcal{A}_{c,d+1}^{\mathcal{L}} \circ \mathcal{B}_d^{\mathcal{L}} \circ \mathcal{B}_{d-1}^{\mathcal{L}} \cdots \circ \mathcal{B}_1^{\mathcal{L}} \circ \rho_0$$

where $\mathcal{B}_i^{\mathcal{L}} := \Pi_i \circ \mathcal{L} \circ U_i \circ \mathcal{A}_{c,i}^{\mathcal{L}}$, $\rho_0 = |0 \ldots 0\rangle \langle 0 \ldots 0|$ and Π corresponds to the algorithm outputting s. For each $i \in \{1, \ldots d\}$, construct the tuples $\bar{S}_i$ using Algorithm 32. Let $\mathcal{M}_i$ be the shadow of $\mathcal{L}$ with respect to $\bar{S}_i$ (see Definition 30). Define
$$\mathcal{B}^{\mathcal{M}} := \Pi \circ \mathcal{A}_{c,d+1}^{\mathcal{L}} \circ \mathcal{B}_d^{\mathcal{M}} \circ \mathcal{B}_{d-1}^{\mathcal{M}} \cdots \circ \mathcal{B}_1^{\mathcal{M}}.$$

where $\mathcal{B}_i^{\mathcal{M}} := \Pi_i \circ \mathcal{M}_i \circ U_i \circ \mathcal{A}_{c,i}^{\mathcal{L}}$. We have

$$\left| \Pr[s \leftarrow \mathcal{B}^{\mathcal{L}}] - \Pr[s \leftarrow \mathcal{B}^{\mathcal{M}}] \right|$$
$$= \left| \text{tr}[\Pi \mathcal{A}_{c,d+1}^{\mathcal{L}} \mathcal{B}_d^{\mathcal{L}} \mathcal{B}_{d-1}^{\mathcal{L}} \ldots \mathcal{B}_1^{\mathcal{L}} \rho_0] - \text{tr}[\Pi \mathcal{A}_{c,d+1}^{\mathcal{L}} \mathcal{B}_d^{\mathcal{M}} \mathcal{B}_{d-1}^{\mathcal{M}} \ldots \mathcal{B}_1^{\mathcal{M}} \rho_0] \right| \qquad \text{we dropped } \circ \text{ for brevity}$$
$$\leq B(\mathcal{B}_1^{\mathcal{L}}(\rho_0), \mathcal{B}_1^{\mathcal{M}}(\rho_0)) +$$
$$\quad B(\mathcal{B}_2^{\mathcal{L}}(\rho_1), \mathcal{B}_2^{\mathcal{M}}(\rho_1)) +$$
$$\quad \vdots$$
$$\quad B(\mathcal{B}_d^{\mathcal{L}}(\rho_{d-1}), \mathcal{B}_2^{\mathcal{M}}(\rho_{d-1}))$$
$$= \sum_{i=1}^{d} \sqrt{2 \Pr[\text{find} : U_i^{\mathcal{L}\backslash \bar{S}_i}, \mathcal{A}_{c,i}^{\mathcal{L}} \circ \rho_{i-1}]}.$$

where for $i \in \{1, 2 \ldots d-1\}$, $\rho_i := \mathcal{B}_i^{\mathcal{M}} \circ \cdots \circ \mathcal{B}_1^{\mathcal{M}} \circ \rho_0$.

So far, everything was essentially the same as in the QNC_d case. The difference arises because of the classical algorithm. We begin with the first term, $\Pr[\text{find} : U_1^{\mathcal{L}\backslash \bar{S}_1}, \mathcal{A}_{c,1}^{\mathcal{L}} \circ \rho_0]$ and denote by $\bar{T}_1 = (T_{1,1}, T_{1,2} \ldots T_{1,d})$ the tuple of subsets queried by $\mathcal{A}_{c,1}$. There are two possibilities: either all queries in $\bar{T}_1$ result in $\perp$ or at least one query yields a non-$\perp$ result. We treat the second event as though the algorithm was able to "find" the solution. Denote the first event by $F_1 := \bar{S}_1 \cap \bar{T}_1 = \emptyset$ and the second event as $\neg F_1 := \bar{S}_1 \cap \bar{T}_1 \neq \emptyset$ where the intersection is component-wise. Note that the random variable of interest here is $\mathcal{L}$ and those derived using it, i.e. $\mathcal{M}_i$s and $\bar{S}_i$s. While the notation is cumbersome, we can use $\Pr[E] = \Pr[E|F]\Pr[F] + \Pr[E|\neg F]\Pr[\neg F] \leq \Pr[E|F] + \Pr[\neg F]$, to write

$$\Pr[\text{find} : U_1^{\mathcal{L}\backslash \bar{S}_1}, \mathcal{A}_{c,1}^{\mathcal{L}} \circ \rho_0] \leq \Pr[\text{find} : V_1^{\mathcal{N}_1 \backslash \bar{R}_1}, \sigma_0] + \Pr[\neg F_1] \qquad (7)$$

where $\sigma_0 := \mathcal{A}_{c,1}^{\mathcal{L}} \circ \rho_0 | F_1$, $\bar{R}_1 := \bar{S}_1 | F_1$, $\mathcal{N}_1 := \mathcal{L}|F_1$ and $V_1 := U_1|F_1$. First, it is clear that $\Pr[\neg F_1] \leq O(\text{poly}(n)2^{-n})$ by the union bound as $\Pr[x \in E \times s_i] \leq 2^{-n}$ for all $i \in \{1, 2 \ldots d\}$. Second, $\mathcal{N}_1$ is uncorrelated to σ_0, and V_1 because we have restricted the sample space to the cases where the correlation is absent by conditioning on F_1 (i.e. $\bar{T}_1$ has

been effectively removed from this part of the analysis). In more detail, the algorithm $\mathcal{A}_{c,1}$ (conditioned on event F_1), ruled out a polynomial number of locations where the various $E \times s_i$s are not. In the remaining query domain, $E \times s_i$s are not restricted, i.e. $\mathcal{N}_1$ is uncorrelated with σ_0. This also means that $\bar{R}_1$ is uncorrelated with σ_0. Finally, note that $\Pr[x \in \bar{R}_1] \leq O\left(\frac{1}{2^n - \text{poly}(n)}\right) \leq O(\text{poly}(n)2^{-n})$. We can therefore apply Corollary 21 to obtain the bound the first term in Equation (7) with $O(\text{poly}(n)2^{-n})$. This yields

$$\Pr[\text{find} : U_1^{\mathcal{L}\backslash \bar{S}_1}, \mathcal{A}_{c,1}^{\mathcal{L}} \circ \rho_0] \leq O(\text{poly}(n)2^{-n})$$

We now apply this reasoning to the second term,[9]

$$\Pr[\text{find} : U_2^{\mathcal{L}\backslash \bar{S}_2}, \mathcal{A}_{c,2}^{\mathcal{L}} \circ \underbrace{\Pi_1 \circ \mathcal{M}_1 \circ U_1 \circ \mathcal{A}_{c,1}^{\mathcal{L}} \circ \rho_0}_{\rho_1}].$$

It is clear,[10] that ρ_1 can contain information about s_0 (as it has access to $\mathcal{M}_1$) and therefore $\bar{S}_2$ was chosen so that $\mathcal{M}_2$ and $\mathcal{L}$ behave identically when their zeroth sub-oracle is queried (whose "access requires s_0 and secret is s_1", see Figure 2). Additionally, $\mathcal{A}_{c,1}^{\mathcal{L}}$ may have partially learnt something about the remaining s_is as well and this would prevent us from applying Lemma 20. We condition as in the case above to obtain

$$\Pr[\text{find} : U_2^{\mathcal{L}\backslash \bar{S}_2}, \mathcal{A}_{c,2}^{\mathcal{L}} \circ \rho_1] \leq \Pr[\text{find} : V_2^{\mathcal{N}_2\backslash \bar{R}_2}, \mathcal{A}_{c,2}^{\mathcal{N}_2} \circ \sigma_1] + \Pr[\neg F_1] \tag{8}$$

where $\sigma_1 := \rho_1|F_1$, $\mathcal{N}_2 := \mathcal{L}|F_1$, $V_2 := U_2|F_1$ and $\bar{R}_2 := \bar{S}_2|F_1$. The first term restricts the sample space (of $\mathcal{L}$) such that σ_1 is uncorrelated with $\bar{R}_2$.[11] We are therefore essentially in the same situation as the starting point of the case above (see Equation (7)). Let $\bar{T}_2$ be the set where $\mathcal{A}_{c,2}^{\mathcal{N}_2}$ queries and define the event $F_2 := \bar{R}_2 \cap \bar{T}_2 = \emptyset$. Conditioning on F_2, we can bound the first term as

$$\Pr[\text{find} : V_2^{\mathcal{N}_2\backslash \bar{R}_2}, \mathcal{A}_{c,2}^{\mathcal{N}_2} \circ \sigma_1] \leq \Pr[\text{find} : W_2^{\mathcal{O}_2\backslash \bar{Q}_2}, \tau_2] + \Pr[\neg F_2] \tag{9}$$

where[12] $\tau_1 := \mathcal{A}_{c,2}^{\mathcal{N}_2} \circ \sigma_1|F_2$, $\mathcal{O}_2 := \mathcal{N}_2|F_1$, $W_2 := V_2|F_2$ and $\bar{Q}_2 := \bar{R}_2|F_2$. Now, $\bar{Q}_2$ is uncorrelated to τ_1 because we restricted to the part of the sample space where the effect $\mathcal{A}_{c,2}^{\mathcal{N}_2}$ has been accounted for. To see this, observe that after the conditioning, the reasoning is analogous to why ρ_1 is uncorrelated to $\bar{S}_2$. We can now apply Corollary 21 with $\Pr[x \in \bar{Q}_2] \leq O(\text{poly}(n)2^{-n})$, $\Pr[\neg F_2] \leq O(\text{poly}(n)2^{-n})$ in Equation (9) and combine it with Equation (8) to obtain

$$\Pr[\text{find} : U_2^{\mathcal{L}\backslash \bar{S}_2}, \mathcal{A}_{c,2}^{\mathcal{L}} \circ \rho_1] \leq O(\text{poly}(n)2^{-n}).$$

Proceeding similarly, one obtains the final bound. □

5 Warm-up | d-Shuffled Simon's Problem (d-SS)

In the previous section, we established the first main result of this work—for each d, we saw that d-Serial Simon's problem is easy for a CQ_1 circuit but hard for QC_d. Our second main goal is to show that for each d, there is a problem which is easy for QC_c (for some constant $c > 0$, wrt d and n) but hard for CQ_d. However, showing depth lower bounds for CQ_d circuits can get quite involved. We, therefore, first re-derive a result due to Chia, Chung and Lai [CCL20]—for each d, there is a problem which is easy for CQ_{2d+1} but hard for CQ_d. While we use essentially the same problem (more on this momentarily), our proof is different and simpler. We build upon these techniques for proving our second main result in Section 8.

The problem we consider in this section is called d-Shuffl*ed* Simon's Problem which is essentially the same as the d-Shuffl*ing* Simon's Problem introduced by Chia, Chung and Lai. The difference arises in the way we construct the Shuffler and consequently in its analysis.

---

[9]It may help to explicitly state the convention: as we condition, we introduce more letters, leaving the indices unchanged.

[10]It may help to recall that, by construction, $\mathcal{M}_1 \ldots \mathcal{M}_{i-1}$ contain no information about $\bar{S}_i$, i.e. the query domain on which $\mathcal{L}$ and $\mathcal{M}_i$ differ.

[11]Note that σ_1 is potentially correlated with $\mathcal{N}_2$ (because it had access to $\mathcal{M}_1$ (see see Figure 2) which contains information about s_0 and note that $\mathcal{N}_2$ also has dependence on s_0; conditioning only excludes those $\mathcal{L}$s where $\bar{T}_1$ is not assigned $\perp$)

[12]The index of the state, i, is incremented to reflect that $\mathcal{M}_1 \ldots \mathcal{M}_i$ have been queried by the state (and not $\mathcal{M}_{i+1} \ldots \mathcal{M}_d$).

5.1 Oracles and Distributions

5.1.1 d-Shuffler

Notation 35. We identify binary strings with their associated integer values implicitly. E.g. $f : \{0,1\}^{2n} \to \{0,1\}^{2n}$ then we may use $f(2^n - 1)$ to denote $f(\underbrace{00\ldots0\,11\ldots1}_{n \text{ many}})$.

We begin with defining a d-Shuffler for a function f. We give two equivalent definitions of a d-Shuffler. The first is more intuitive but the second is easier to analyse.

A d-Shuffler for $f : \{0,1\}^n \to \{0,1\}^n$ is simply a sequence of d permutations $f_0, f_1 \ldots f_{d-1}$ on a larger space, which uses $2n$ length strings and a function f_d such that $f_d \circ f_{d-1} \ldots f_0(x) = f(x)$ for all x in the domain of f and for all points untouched by these "paths" originating from the domain of x, the permutations are modified to output $\perp$.

This definition makes counting arguments slightly convoluted. We therefore observe that a sequence of permutations may equivalently be specified by a sequence of tuples.[13] For instance two permutations over four elements may be expressed as $[0,1,2,3]^T \mapsto [1,3,0,2]^T \mapsto [2,0,3,1]^T$ but the advantage here is that restricting the permutation to a subset (as we did above by defining the permutations to be $\perp$ outside the "paths"), corresponds to simply dropping some elements from the tuple: $[0,1]^T \mapsto [1,3]^T \mapsto [2,0]^T$. See Figure 4.

Definition 36 (Uniform d-Shuffler for f). A *uniform d-Shuffler* for a function $f : \{0,1\}^n \to \{0,1\}^n$ is a sequence of random functions $(f_0, f_1, \ldots f_d)$, where $f_0, \ldots f_d : \{0,1\}^{2n} \cup \{\perp\} \to \{0,1\}^{2n} \cup \{\perp\}$ sampled from a distribution $\mathbb{F}_{\text{shuff}}(d, n, f)$ (to be defined) such that $f_d \circ f_{d-1} \cdots \circ f_0(x) = f(x)$ for all $x \in Z \times \{0,1\}^n$ where $Z := \{(0,\ldots 0)\}$ (containing an n-bit zero string). Let $\mathcal{L}$ be the oracle associated with $(f_i)_{i=0}^d$ and define $\mathbb{O}_{\text{shuff}}(d, n, f)$ to be the corresponding distribution.

The sampling process is defined in two equivalent ways.

First definition:

- Sample $f_0', \ldots f_{d-1}' : \{0,1\}^{2n} \to \{0,1\}^{2n}$ from a uniform distribution of permutation functions acting on strings of length $2n$.

- Define f_d to be such that $f_d \circ f_{d-1}' \circ \cdots \circ f_0'(x) = f(x)$ for all $x \in Z \times \{0,1\}^n$ and $f_d \circ f_{d-1}' \circ \ldots f_0'(x) = \perp$ otherwise.

- For each $i \in \{0, 1 \ldots d-1\}$,

    - define $X_{i+1} := f_i' \circ \ldots f_0'(X_0)$ where $X_0 = Z \times \{0,1\}^n$ and
    - define
    $$f_i(x) := \begin{cases} f_i'(x) & \forall x \in X_{i-1} \\ \perp & \forall x \notin X_{i-1}. \end{cases}$$

Second equivalent definition:

- Let $t_0, t_1 \ldots t_{d-1}$ each be a tuple $(x_1, x_2 \ldots x_N)$ of size $N := 2^n$, sampled uniformly from the collection of all size N tuples containing distinct elements $x_i \in \{0,1\}^{2n}$. Let $t_{-1} := (0, 1, \ldots N)$ and $t_d := (f(0), f(1) \ldots f(N-1))$ (see Notation 35).
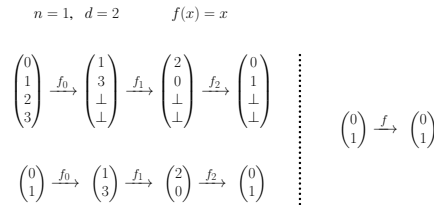
Figure 4: Sequence of tuples instead of restricted permutation functions.

---

[13]To prevent confusion with the standard notation for permutations, we denote these tuples by columns.

- $f_0, \ldots f_d$ | For each $i \in \{0, \ldots, d-1\}$, define f_i as follows.

  - If $x \notin t_{i-1}$, then define $f_i(x) = \perp$.
  - Otherwise, suppose x is the jth element in t_{i-1} and y is the jth element in t_i. Then, define $f_i(x) = y$.

Given the second definition, a d-Shuffler may equivalently be defined as a sequence $(t_0, \ldots t_{d-1}, t_d)$ of tuples sampled as described above. We overload the notation and let $\mathbb{F}_{\text{shuff}}(d, n, f)$ return $(t_i)_{i=0}^{d}$ and similarly let $\mathbb{O}_{\text{shuff}}(d, n, f)$ return the oracle associated with $(t_i)_{i=0}^{d}$.

Finally, if f is omitted when $\mathbb{F}_{\text{shuff}}$ or $\mathbb{O}_{\text{shuff}}$ are invoked, then it is assumed that $f \sim \mathbb{F}_R(n)$.

Notation 37. In this section, we drop the word "uniform" for conciseness.

5.1.2 The d-SS Problem

Definition 38 (*d-Shuffled Simon's Distribution and Oracle*). We define the d-Shuffled Simon's Function distribution, $\mathbb{F}_{\text{SS}}(d, n)$ and the corresponding d-Shuffled Simon's Oracle distribution $\mathbb{O}_{\text{SS}}(d, n)$ by specifying its sampling procedure.

- Sample a random Simon's function $(f, s) \sim \mathbb{F}_S(n)$ (see Definition 23).

- Sample a d-Shuffler $(f_i)_{i=0}^{d} \sim \mathbb{F}_{\text{Shuff}}(d, n, f)$ where f_i are functions (see Definition 36).

Return $((f_i)_{i=0}^{d}, s)$ when $\mathbb{F}_{\text{SS}}(d, n)$ is sampled and $(\mathcal{F}, s)$ when $\mathbb{O}_{\text{SS}}(d, n)$ is sampled where $\mathcal{F}$ is the oracle associated with $(f_i)_{i=0}^{d}$.

Definition 39 (*d-Shuffled Simon's Problem*). Let $(\mathcal{F}, s) \sim \mathbb{O}_{\text{SS}}(d, n)$ (see Definition 38) be sampled from the d-Shuffled Simon's Oracle distribution and $Q \sim \mathbb{O}_{\text{Shuff}}(d, n)$ be a random d-Shuffler (see Definition 36). The d-Shuffled Simon's Problem is,

- Search version: Given $\mathcal{F}$, find s.

- Decision version: Given either $\mathcal{F}$ or Q with equal probability, output 1 if $\mathcal{F}$ was given and 0 otherwise.

5.2 Shadow Boilerplate

For the analysis, we need to consider shadow oracles associated with the d-Shuffled Simon's Oracle. The definition is somewhat redundant—it is the direct analogue of Definition 30. See Figure 5.

Definition 40 (*d-Shuffled Simon's Shadow Oracle*). Given

- $((f_i)_{i=0}^{d}, s)$ from the sample space of $\mathbb{F}_{\text{SS}}(d, n)$ and

- a tuple of subsets $\bar{S} = (S_1, S_2 \ldots S_d)$ where each $S_i \subseteq \{0, 1\}^{2n}$,

let $\mathcal{F}$ be the oracle associated with $(f_i)_{i=0}^{d}$. Then, the d-Shuffled Simon's Shadow Oracle (or simply the shadow) $\mathcal{G}$ for the oracle $\mathcal{F}$ with respect to $\bar{S}$ is defined to be the oracle associated with $(f_0, f_1' \ldots f_d')$ where

$$f_i' := \begin{cases} f_i(x) & x \in \{0, 1\}^{2n} \setminus S_i \\ \perp & x \in S_i. \end{cases}$$

for all $i \in \{1, 2 \ldots d\}$.

The analysis here is a straightforward adaptation of the QNC_d analysis for d-Serial Simon's. Thus, we use the analogue of Algorithm 41. The only difference is in the numbering convention. The domain of the ith oracle was determined by s_{i-1} in Algorithm 41 while here, the domain of the ith oracle is determined by X_i.

Algorithm 41 ($\bar{S}_j$ for QNC_d exclusion using d-SS).
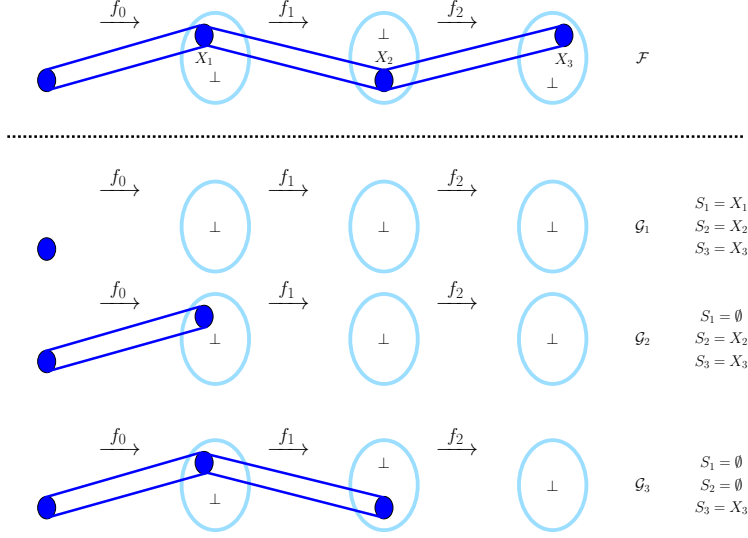*Input:*

- $1 \leq j \leq d$ *and*

Figure 5: Shadows for d-Shuffled Simon's Oracle. Ignore the arrows below the function labels $f_0, \ldots f_2$ at first. Suppose $d = 3$. Consider the row for $\mathcal{F}$. As before, the ovals represent the query domains of functions f_0, f_1, f_2 and f_3. Similarly for the shadows $\mathcal{G}_1, \ldots \mathcal{G}_3$. The shaded regions denote a non-$\perp$ response. Let $\bar{S}_i$ be the query domain at which $\mathcal{F}$ and $\mathcal{G}_i$ differ. The construction, as before, ensures that $\mathcal{G}_1 \ldots \mathcal{G}_{i-1}$ do not contain any information about $\bar{S}_i$.

- $((f_i)_{i=0}^d, s)$ *from the sample space of* $\mathbb{F}_{\mathrm{SS}}(d, n)$.

Output: $\bar{S}_j$, *a tuple of d subsets, defined as*

$$\bar{S}_j := \begin{cases} (\emptyset, \ldots \emptyset, X_j, \ldots X_d) & j > 1 \\ (X_1, \ldots X_d) & j = 1 \end{cases}$$

where for each $i \in \{1, \ldots d\}$, $X_i = f_{i-1}(X_{i-1})$ *with* $X_0 = \{0, 1\}^n$.

Proposition 42. *For all* $2 \le i \le d$, *let* $\mathcal{G}_1 \ldots \mathcal{G}_{i-1}$ *denote the shadows of* $(\mathcal{F}, s) \in \mathbb{O}_{\mathrm{SS}}(d, n)$ *(see Definition 40) with respect to* $\bar{S}_1 \ldots \bar{S}_{i-1}$ *(constructed using Algorithm 41 with the index and* $\mathcal{F}$ *as inputs). Then,* $\mathcal{G}_1 \ldots \mathcal{G}_{i-1}$ *contain no information about* $\bar{S}_i$.

Proof. To see this, note that even though to define $\mathcal{G}_1$, we used $\bar{S}_1 = (X_1, X_2 \ldots X_d)$, still $\mathcal{G}_1$ contains no information about $X_2, \ldots X_d$ (and so no information about $\bar{S}_2$ either) because it always outputs $\perp$ (except for the zeroth oracle which can reveal X_1). Similarly, $\mathcal{G}_2$ contains information about X_1, X_2 but not about $X_3 \ldots X_d$ (thus $\bar{S}_3$). Analogously for $\mathcal{G}_3$ and so on. □

Proposition 43. *Let* $\bar{S}_i$ *be the output of Algorithm 41 with i and* $(\mathcal{F}, s)$ *as inputs where* $(\mathcal{F}, s) \sim \mathbb{O}_{\mathrm{SS}}(d, n)$ *(see Definition 40). Let x be some fixed query (in the query domain of* $\mathcal{F}$*). Then,* $\Pr[x \in \bar{S}_i] \le O(d \cdot 2^{-n})$.

Proof. For any fixed $x_i \in \{0, 1\}^{2n}$, $\Pr[x_i \in X_i] = \frac{1}{2^n}$ (see Remark 82 using $N = 2^n$ and $M = 2^{2n}$). The union bound, then, readily bounds the desired probability. □

5.3 Depth Lower bounds for d-Shuffled Simon's Problem (d-SS)

Again, as in the analysis for d-Serial Simon's problem, we begin with briefly demonstrating the depth lower bound for QNC_d (with classical post-processing) and then generalise it to CQ_d.

5.3.1 d-Shuffled Simon's Problem is hard for QNC_d

Theorem 44 (d-SS is hard for QNC_d). *Let $(\mathcal{F}, s) \sim \mathbb{O}_{\mathrm{SS}}(d, n)$, i.e. let $\mathcal{F}$ be an oracle for a random d-Shuffled Simon's problem of size n and period s. Let $\mathcal{A}^{\mathcal{F}}$ be any d depth quantum circuit (see Definition 5 and Remark 11) acting on $O(n)$ qubits, with query access to $\mathcal{F}$. Then $\Pr[s \leftarrow \mathcal{A}^{\mathcal{F}}] \leq \mathrm{negl}(n)$, i.e. the probability that the algorithm finds the period is exponentially small.*

This follows from the proof of Theorem 33 with no essential change. The main difference is in the details of how the shadow is constructed—earlier it was comprised of $E \times s_i$s, now X_is comprise it. However, that doesn't change the relevant properties (see Proposition 42 and Proposition 43) for the main argument (which is given explicitly in Subsection B.1 for completeness).

We expect the QNC_d hardness of d-SS to readily generalise to hardness for QC_d (following the d-Serial Simon's hardness proof) but we do not prove it here.

5.3.2 d-Shuffled Simon's Problem is hard for CQ_d | idea

The generalisation to CQ_d takes some work. We first describe the basic idea behind our approach and then formalise these. These draw from the insights of Chia, Chung and Lai [CCL20] but differ substantially in their implementation. Let $((f_i)_{i=0}^d, s) \sim \mathbb{F}_{\mathrm{SS}}(d, n)$. For our discussion, we require three key concepts.

1. The first, already introduced in the proof of Theorem 34, was the notion of conditioning the oracle distribution (and the quantities that depend on it) based on a "transcript" as the algorithm proceeds and analysing the conditioned cases. In the proof of Theorem 34, the locations queried by the classical algorithm constituted this transcript but it could, and as we shall see it will, more generally contain other correlated variables.

2. The second, is the notion of an almost uniform d-Shuffler. Neglect, for the moment, the last function f_d which is used to define the d-Shuffler, $(f_i)_{i=0}^d$. Then, informally, suppose that an algorithm (which can even be computationally unbounded) is given access to a uniform d-Shuffler and it produces a $\mathrm{poly}(n)$ sized advice (a string that is correlated with the Shuffler). One can show that for advice strings which appear with non-vanishing probability, the d-Shuffler conditioned on the advice string, continues to stay almost uniform.

3. The third, is basically the bootstrapping of the proof of Theorem 44, by letting it now play the role of the algorithm mentioned above. This ties the loose end left above, the question of correlation with f_d. A generic QNC_d circuit may reveal some information about $(f_0, \ldots f_{d-1})$ and f_d in the poly-length string it outputs, however, the proof of Theorem 44 guarantees that the behaviour of this algorithm cannot be very different from one which has no access to f_d.

In the next section, we formalise step two and in the one that follows, we stitch everything together to establish d-SS is hard for CQ_d. We later use parts of this proof for establishing our second main result.

6 Technical Results II

The description in Subsection 5.3.2 above, used the notion of an almost uniform d-Shuffler. In this section, we make this notion precise.

6.1 Sampling argument for Uniformly Distributed Permutations

The basic idea used in this section is called a "pre-sampling argument" which we have adapted from the work of Coretti et al. [Cor+17] and Chia, Chung and Lai [CCL20]. It was considered earlier in cryptographic contexts by Unruh [Unr07] and for communication complexity by Göös et al. [Göö+15] and Kothari, Meka and Raghavendra [KMR17]. For our purposes, we need to generalise their result.

We first describe the idea in its simplest form (that considered by Coretti et al. [Cor+17]). Let $N = 2^n$ and fix some small $\delta > 0$. Suppose there is an N-bit random string X which is uniformly distributed. Suppose an arbitrary function of X, $f(X)$ is known. Then, given we focus on values of $f(X)$ which occur with probability above a threshold, say γ, i.e. for r such that $\Pr[f(X) = r] \geq \gamma = 2^{-m}$, the main result, informally, is that $X|(f(X) = r)$ may be viewed as a "convex combination of random variables" which are "δ far from" X's with a small number of bits (scaling as

$-\log \gamma/\delta = m/\delta$) fixed. We justify and reify the phrases in quotes shortly. In our setting, the random variable X is replaced by the d-Shuffler and the function f would encode the advice generated by a QNC_d circuit gives after acting upon the d-Shuffler.[14]

While Chia, Chung and Lai [CCL20] already generalised this argument to the case of a sequence of permutations over N elements for their analysis, we show here that the idea itself can be applied quite generally—first, one needn't restrict to uniform distributions, second, a rather limited structure on the random variable suffices, i.e. one needn't restrict to strings or permutations. Using these, one can also show a composition result, where repeated advice are given. This is pivotal to the analysis of CQ_d circuits where QNC_d circuits repeatedly give advice.

In the following, for clarity, we present our results for a single uniformly distributed permutation but do not use that fact in our derivation. The results thus directly lift to the d-Shuffler with minor tweaks to the notation.

6.1.1 Convex Combination of Random Variables

We first make the notion of "convex combination of random variables" precise. Consider a function f which acts on a random permutation, say t, to produce an output, i.e. $f(t) = r$ where r is an element in the range of f.[15] This range can be arbitrary. We say a *convex combination* $\sum_i p_i t_i$ of random variables t_i is *equivalent* to t if for all functions f, and all outputs s in its range, $\sum_i p_i \Pr[f(t_i) = s] = \Pr[f(t) = s]$. This relation is denoted by $\sum_i p_i t_i \equiv t$.

6.1.2 The "parts" notation

While permutations are readily defined as an ordered set of distinct elements, it would nonetheless be useful to introduce what we call the "parts" notation which allows one to specify parts of the permutation.

Notation 45. Consider a permutation t over N elements, labelled $\{0, 1 \ldots N-1\}$.

- *Parts:* Let $S = \{(x_i, y_i)\}_{i=1}^M$ denote the mapping of $M \le N$ elements under some permutation, i.e. there is some permutation t, such that $t(x_i) = y_i$. Call any such set S a "part" and its constituents "paths".

    - Denote by $\Omega_{\text{parts}}(N)$ the set of all such "parts".
    - Call two parts $S = \{(x_i, y_i)\}_i$ and $S' = \{(x'_{i'}, y'_{i'})\}_{i'}$ *distinct* if for all i, i' (a) $x_i \ne x_{i'}$, and (b) there is a permutation t such that $t(x_i) = y_i$ and $t(x_{i'}) = y_{i'}$.
    - Denote by $\Omega_{\text{parts}}(N, S)$ the set of all parts $S' \in \Omega_{\text{parts}}(N)$ such that S' is distinct from S.

- *Parts in t:* The probability that t maps the elements as described by S may be expressed as $\Pr[\wedge_{i=1}^M (t(x_i) = y_i)] = \Pr[S \subseteq \text{parts}(t)]$ where $\text{parts}(t) := \{(x, t(x))\}_{x=0}^{N-1}$.

- *Conditioning t based on parts:* Finally, use the notation t_S to denote the random variable t conditioned on $S \subseteq \text{parts}(t)$.

To clarify the notation, consider the following simple example.

Example 46. Let $N = 2$. Then $\Omega_{\text{parts}}(N) = \{\{(0,0)\}, \{(1,1)\}, \{(0,0),(1,1)\}, \{(0,1)\}, \{(1,0)\}, \{(0,1),(1,0)\}\}$ and there are only two permutations, $t(x) = x$ and $t'(x) = x \oplus 1$ for all $x \in \{0, 1\}$. An example of a part S is $S = \{(0,0)\}$. A part (in fact the only part) distinct from S is $(1,1)$, i.e. $\Omega_{\text{parts}}(N, S) = \{(1,1)\}$.

6.1.3 δ non-uniform distributions

Using the "parts" notation, we define uniform distributions over permutations and a notion of being δ non-uniform—distributions which are at most δ "far from" being being uniform.[16]

---

[14]The term "pre-sampling argument" arose from the cryptographic application in the context of random oracles. There, the adversary is allowed to arbitrarily interact with the random oracle (or pre-sample it) before initiating the protocol but only allowed to keep a poly sized advice from that interaction.

[15]The function will later be interpreted as an algorithm and the random permutation accessed via an oracle.

[16]Clarification to a possible conflict in terms: We use the word uniform in the sense of probabilities—a uniformly distributed random variable—and not quite in the complexity theoretic sense—produced by some Turing Machine without advice.

Definition 47 (uniform and δ non-uniform distributions). Consider the set, $\Omega(N)$, of all possible permutations of N objects labelled $\{0, 1, 2 \ldots N - 1\}$. Let $\mathbb{F}$ be a distribution over Ω. Call $\mathbb{F}$ a *uniform distribution* if for $u \sim \mathbb{F}$, $\Pr[S \subseteq \mathrm{parts}(u)] = \frac{(N-M)!}{N!}$ for all parts $S \in \Omega_{\mathrm{parts}}(N)$.

An arbitrary distribution $\mathbb{F}^\delta$ over Ω is δ non-uniform if it satisfies for $t \sim \mathbb{F}^\delta$

$$\Pr[S \subseteq \mathrm{parts}(t)] \leq 2^{\delta|S|} \cdot \Pr[S \subseteq \mathrm{parts}(u)]$$

for all parts $S \in \Omega_{\mathrm{parts}}(N)$.

Finally, $\mathbb{F}^{p,\delta}$ over Ω is (p, δ) non-uniform if there is a subset of parts S of size $|S| \leq p$ such that the distribution conditioned on S specifying a part of the permutation, becomes δ non-uniform over parts distinct from S. Formally, let $t' \sim \mathbb{F}^{p,\delta}$. Then t' is (p, δ) non-uniformly distributed if t'_S is δ non-uniformly distributed over all $S' \in \Omega_{\mathrm{parts}}(N, S)$ (see Notation 45), i.e.

$$\Pr[S' \subseteq \mathrm{parts}(t')|S \subseteq \mathrm{parts}(t')] \leq 2^{\delta|S'|} \cdot \Pr[S' \subseteq \mathrm{parts}(u)|S \subseteq \mathrm{parts}(u)]. \tag{10}$$

In Equation (10), we are conditioning a uniform distribution using the "parts" notation which may be confusing. The following should serve as a clarification.

Note 48. Let $u \sim \mathbb{F}$ as above. Then, we have $\Pr[S' \subseteq \mathrm{parts}(u)|S \subseteq \mathrm{parts}(u)] = \frac{(N-|S|-|S'|)!}{(N-|S|)!}$ where $S' \in \Omega_{\mathrm{parts}}(N, S)$ and $S \in \Omega_{\mathrm{parts}}(N)$. Let $S = \{(x_i, y_i)\}_{i=1}^{|S|}$. Then, the conditioning essentially specifies that the $|S|$ elements in $X = (x_i)_{i=1}^{|S|}$ must be mapped to $Y = (y_i)_{i=1}^{|S|}$ by u, i.e. $u(x_i) = y_i$, but the remaining elements $\{0, 1 \ldots N - 1\} \backslash X$ are mapped uniformly at random to $\{0, 1 \ldots N - 1\} \backslash Y$.

Clearly, for $\delta = 0$, the δ non-uniform distribution becomes a uniform distribution. However, this can be achieved by relaxing the uniformity condition in many ways. The δ non-uniform distribution is defined the way it is to have the following property. Notice that $|S|$ appears in a form such that the product of two probabilities, $\Pr[S_1 \subseteq \mathrm{parts}(t)]$ and $\Pr[S_2 \subseteq \mathrm{parts}(t)]$ yields $|S_1| + |S_2|$, e.g. $(1 + \delta)^{|S|}$ instead of $2^{\delta|S|}$ would also have worked.[17] This property plays a key role in establishing that in the main decomposition (as described informally in Subsection 6.1), the number of "paths" (in the informal discussion it was bits) fixed is small. We chose the prefactor $2^{|S|}$ for convenience—unlikely events in our analysis are those which are exponentially suppressed, and we therefore take the threshold parameter to be $\gamma = 2^{-m}$. These choices result in a simple relation between $|S|$ and m.

Notation 49. To avoid double negation, we use the phrase "t is more than δ non-uniform" to mean that t is not δ non-uniform. Similarly, we use the phrase "t is at most δ non-uniform" to mean that t is δ non-uniform.

As shall become evident, the only property of a uniform distribution we use in proving the main proposition of this section, is the following. It not only holds for all distributions over permutations, but also for d-Shuffler. We revisit this later.

Note 50. Let t be a permutation sampled from an arbitrary distribution $\mathbb{F}'$ over $\Omega(N)$. Let $S, S' \subseteq \Omega_{\mathrm{parts}}(N)$ be *distinct* parts (see Notation 45). Then,

$$\Pr[S \subseteq \mathrm{parts}(t) \wedge S' \subseteq \mathrm{parts}(t)] = \Pr[S \cup S' \subseteq \mathrm{parts}(t)].$$

If the parts are not distinct, then both expressions vanish.

6.1.4 Advice on uniform yields δ non-uniform

We are now ready to state and prove the simplest variant of the main proposition of this section.

Proposition 51 ($\mathbb{F}|r' \equiv \mathrm{conv}(\mathbb{F}^{p,\delta})$). *Premise:*

- *Let $u \sim \mathbb{F}$ where $\mathbb{F}$ is a uniform distribution over all permutations, Ω, on $\{0, 1 \ldots N - 1\}$, as in Definition 47 with $N = 2^n$.*

- *Let r be a random variable which is arbitrarily correlated to u, i.e. let $r = g(u)$ where g is an arbitrary function.*

- *Fix any $\delta > 0$, $\gamma = 2^{-m} > 0$ (m may be a function of n) and some string r'.*

---

[17] The former was chosen by Chia, Chung and Lai [CCL20] while the latter by Coretti et al. [Cor+17] and possibly others.

- *Suppose*

$$\Pr[r = r'] \geq \gamma. \tag{11}$$

- *Let t denote the variable u conditioned on $r = r'$, i.e. let $t = u|(g(u) = r')$.*

Then, t is "γ-close" to a convex combination of finitely many (p, δ) non-uniform distributions, i.e.

$$t \equiv \sum_i \alpha_i t_i + \gamma' t'$$

where $t_i \sim \mathbb{F}_i^{p,\delta}$ and $\mathbb{F}_i^{p,\delta}$ is (p, δ) non-uniform with $p = \frac{2m}{\delta}$. The permutation t' is sampled from an arbitrary (but normalised) distribution over Ω and $\gamma' \leq \gamma$.

Proof. Suppose that t is more than δ non-uniformly distributed (see Definition 47 and Notation 49), otherwise then there is nothing to prove (set α_1 to 1, and t_i to t, remaining α_is and γ' to zero). Recall $\Omega_{\mathrm{parts}}(N)$ is the set of all parts (see Notation 45). Let the subset $S \in \Omega_{\mathrm{parts}}(N)$ be the maximal subset of parts (i.e. subset with the largest size) such that

$$\Pr[S \subseteq \mathrm{parts}(t)] > 2^{\delta \cdot |S|} \cdot \Pr[S \subseteq \mathrm{parts}(u)]. \tag{12}$$

Claim 52. Let S and t be as described above. The random variable t conditioned on being consistent with the paths in $S \in \Omega_{\mathrm{parts}}(N)$, i.e. t_S, is δ non-uniformly distributed over $S' \subseteq \Omega_{\mathrm{parts}}(N, S)$, is δ non-uniformly distributed.

We prove Claim 52 by contradiction. Suppose that t_S is "more than" δ non-uniform. Then, there exists some $S' \in \Omega_{\mathrm{parts}}(N, S)$ such that

$$\Pr[S' \subseteq \mathrm{parts}(t_S)] = \Pr[S' \subseteq \mathrm{parts}(t)|S \subseteq \mathrm{parts}(t)] > 2^{\delta \cdot |S'|} \cdot \Pr[S' \subseteq \mathrm{parts}(u)|S \subseteq \mathrm{parts}(u)]. \tag{13}$$

Since S' violates the δ non-uniformity condition for t_S, the idea is to see if the union $S \cup S'$ violates the δ non-uniformity condition for t. If it does, we have a contradiction because S was by assumption the maximal subset satisfying this property. Indeed,

$$\begin{aligned}
\Pr[S \cup S' \subseteq \mathrm{parts}(t)] &= \Pr[S \subseteq \mathrm{parts}(t) \wedge S' \subseteq \mathrm{parts}(t)] && \because S \text{ and } S' \text{ are distinct} \\
&= \Pr[S \subseteq \mathrm{parts}(t)] \Pr[S' \subseteq \mathrm{parts}(t)|S \subseteq \mathrm{parts}(t)] && \text{conditional probability} \\
&> 2^{\delta \cdot (|S|+|S'|)} \cdot \Pr[S \subseteq \mathrm{parts}(u)] \Pr[S' \subseteq \mathrm{parts}(u)|S \subseteq \mathrm{parts}(u)] && \text{using (12) and (13)} \\
&= 2^{\delta \cdot |S \cup S'|} \cdot \Pr[S \cup S' \subseteq \mathrm{parts}(u)] && \because S \text{ and } S' \text{ are disjoint}
\end{aligned}$$

which completes the proof.

Claim 52 shows how to construct a δ non-uniform distribution after conditioning but we must also bound $|S|$. This is related to how likely is the r' we are conditioning upon, i.e. the probability of $g(u)$ being r'.

Claim 53. One has

$$|S| < \frac{m}{\delta}.$$

While Equation (12) lower bounds $\Pr[S \subseteq \mathrm{parts}(t)]$, the upper bound is given by

$$\begin{aligned}
\Pr[S \subseteq \mathrm{parts}(t)] &= \Pr[S \subseteq \mathrm{parts}(u)|(g(u) = r')] \\
&= \Pr[S \subseteq \mathrm{parts}(u) \wedge g(u) = r']/\Pr[g(u) = r'] \\
&\leq \Pr[S \subseteq \mathrm{parts}(u) \wedge g(u) = r'] \cdot \gamma^{-1} \\
&\leq \Pr[S \subseteq \mathrm{parts}(u)] \cdot \gamma^{-1}. \tag{14}
\end{aligned}$$

Combining these, we have $2^{\delta \cdot |S|} < 2^m$, i.e., $|S| < \frac{m}{\delta}$.

Using Bayes rule on the event that $S \subseteq \mathrm{parts}(t)$ we conclude that

$$t \equiv \alpha_1 t_1 + \alpha'_1 t'_1$$

where $\alpha_1 = \Pr[S \subseteq \mathrm{parts}(t)]$, $t_1 = t_S$, i.e. t conditioned on $S \subseteq \mathrm{parts}(t)$, $\alpha'_1 = 1 - \alpha_1$ and t'_1 is t conditioned on $S \not\subseteq \mathrm{parts}(t)$. Further, while t_1 is (p, δ) non-uniform (from Claim 52 and Claim 53), t'_1 may not be. Proceeding as we

did for t, if t'_1 is itself δ non-uniform, there is nothing left to prove (we set $\alpha_2 = \alpha'_1$ and $t_2 = t'_1$ and the remaining α_is and γ' to zero). Also assume that $\alpha'_1 > \gamma$ because otherwise, again, there is nothing to prove.

Therefore, suppose that t'_1 is not δ non-uniform. Note that the proof of Claim 52 goes through for any permutation which is not δ non-uniform. Thus, the claim also applies to t'_1 where we denote the maximal set of parts by S_1. Let t_2 be t'_1 conditioned on $S_1 \subseteq \text{parts}(t'_1)$ and t'_2 be t'_1 conditioned on $S_1 \nsubseteq \text{parts}(t'_1)$. Using Bayes rule as before, we have

$$t \equiv \alpha_1 t_1 + \alpha_2 t_2 + \alpha'_2 t'_2.$$

Adapting the statement of Claim 52 (with t'_1 playing the role of t and S_1 playing the role of S) to this case, we conclude that t_2 is δ non-uniform but we still need to show that $|S_1| \leq p$. We need the analogue of Claim 53 which we assert is essentially unchanged.

Claim 54. One has

$$|S_i| < \frac{2m}{\delta}. \tag{15}$$

The proof is deferred to Subsection B.2. The factor of two appears because for the general case, we use both $\alpha'_i > \gamma$ and $\Pr[g(u) = r'] > \gamma$. One can iterate the argument above. Suppose

$$t \equiv \alpha_1 t_1 + \dots \alpha_j t_j + \alpha'_j t'_j \tag{16}$$

where $t_1, \dots t_j$ are (p, δ) non uniformly distributed while t'_j is not and $\alpha'_j := \Pr[S \nsubseteq \text{parts}(t) \wedge \dots \wedge S_{j-1} \nsubseteq \text{parts}(t)] > \gamma$ (else one need not iterate). Let S_j be the maximal set such that $t_{j+1} := t'_j | S_j \subseteq \text{parts}(t'_j)$ is δ non-uniform (which must exist from Claim 52) and let $t'_{j+1} := t'_j | S_j \nsubseteq \text{parts}(t'_j)$. Let $\alpha'_{j+1} := \Pr[S_j \nsubseteq \text{parts}(t'_j)]$ which equals $\Pr[S \nsubseteq \text{parts}(t) \wedge \dots \wedge S_j \nsubseteq \text{parts}(t)]$. From Claim 54, $|S_j| < 2m/\delta \leq p$ therefore t_{j+1} is (p, δ) non-uniform.

We now argue that the sum in Claim 54 contains finitely many terms. At every iteration, α'_i strictly decreases because at each step, more constraints are added; $S_i \neq S_j$ for all $i \neq j$ (otherwise conditioning on S_j (if $j \geq i$) as in Claim 52 could not have any effect). Since $\Omega_{\text{parts}}(N)$ is finite, the decreasing sequence $\alpha'_1 \dots \alpha'_i$ must, for some integer i, satisfy $\alpha_i \leq \gamma$ after finitely many iterations. □

6.1.5 Iterating advice and conditioning on uniform distributions | δ non-β-uniform distributions

Once generalised to the d-Shuffler (which, as we shall, see is surprisingly simple), recall that the way we intend to use the above result is to repeatedly get advice from a quantum circuit, a role played by g in the previous discussion. However, the way it is currently stated, one starts with a uniformly distributed permutation u for which some advice $g(u)$ is given but one ends up with (p, δ) non-uniform distributions. We want the result to apply even when we start with a (p, δ) non-uniform distribution.

As should become evident shortly, the right generalisation of Proposition 51 for our purposes is as follows. Assume that the advice being conditioned occurs with probability at least $\gamma = 2^{-m}$ and think of m as being polynomial in n; $\delta > 0$ is some constant and $p = 2m/\delta$.

- Step 1: Let $t \sim \mathbb{F}^{\delta'}$ be δ' non-uniform[18] and $s \sim \mathbb{F}^{\delta'}|r$ be $t|(g(t) = r)$. Then it is straightforward to show that $s \equiv \sum_i \alpha_i s_i$ where s_i are $(p, \delta + \delta')$ non-uniform, which we succinctly write as

$$\mathbb{F}^{\delta'}|r \equiv \text{conv}(\mathbb{F}^{p, \delta + \delta'}).$$

Observation: If $t \sim \mathbb{F}^{p, \delta}$ is (p, δ) non-uniform, then there is some S of size at most p such that $t \sim \mathbb{F}^{\delta|\beta}$ is δ non-β-uniform where[19] $\beta := (S)$. A β-uniform distribution is simply a uniform distribution conditioned on having S as parts. This amounts to basically making the conditioning explicit. Having this control will be of benefit later.

- Step 2: It is not hard to show that Step 1 goes through unchanged if non-uniform is replaced with non-β-uniform for an arbitrary β.

---

[18]Notation: When I say t is δ non-uniform, it is implied that t is sampled from a δ non-uniform distribution.

[19]The conditioning is in superscript because it is non-standard; standard would be $S \subseteq \text{parts}(t)$ which is too long.

These combine to yield the following. Let $t \sim \mathbb{F}^{\delta'|\beta}$ be a δ' non-β-uniform distribution and $s \sim \mathbb{F}^{\delta'|\beta}|r$ be $t|(g(t) = r)$. Then $t \equiv \sum_i \alpha_i s_i$ where $s_i \sim \mathbb{F}^{p,\delta+\delta'|\beta}$ are $(p, \delta + \delta')$ non-β-uniform,[20] which we briefly express as

$$\mathbb{F}^{\delta'|\beta}|r \equiv \mathrm{conv}(\mathbb{F}^{p,\delta+\delta'|\beta}).$$

Observe that this composes well,

$$\mathbb{F}^{p,\delta+\delta'|\beta}|r \equiv \mathrm{conv}(\mathbb{F}^{2p,2\delta+\delta'|\beta}). \tag{17}$$

To see this, consider the following:

- For some S_i, s_i (as defined in the statement above) is $\delta'' := \delta + \delta'$ non-β'-uniform where $\beta' := (S \cup S_i)$ if $\beta = (S)$.

- With t set to s_i, β set to β', one can apply the above to get $s_i|(h(s_i) = r') \equiv \sum_i \alpha_i' q_i$ where q_i are $(p, \delta + \delta'')$ non-β'-uniform.

- Note that q_i are also $(2p, 2\delta + \delta')$ non-β-uniform; which we succinctly denoted as $\mathbb{F}^{2p,2\delta+\delta'|\beta}$.

Clearly, if this procedure is repeated $\tilde{n} \leq \mathrm{poly}(n)$ times, starting from $\delta = 0$ and $\beta = (\emptyset)$, then the final convex combination would be over $\mathbb{F}^{\tilde{n}p,\tilde{n}\delta}$. As we shall see, for our use, it suffices to ensure that $\tilde{n}\delta$ is a small constant and that $\tilde{n}p = \frac{\tilde{n}m}{\delta} \leq \mathrm{poly}(n)$. Choosing $\delta = \Delta/\tilde{n}$ for some small fixed $\Delta > 0$ yields $\tilde{n}\delta = \Delta$ and $\tilde{n}p = \frac{\tilde{n}^2 m}{\Delta}$ which is indeed bounded by $\mathrm{poly}(n)$ (recall m and $\tilde{n}$ are bounded by $\mathrm{poly}(n)$).

One can define a notion of closeness to any arbitrary distribution, as we did for closeness to uniform. To this end, first consider the following.

Definition 55 (δ non-$\mathbb{G}$ distributions—$\mathbb{G}^\delta$). Let s be sampled from an arbitrary distribution, $\mathbb{G}$, over the set of all permutations $\Omega(N)$ of N objects and fix any $\delta > 0$.

Then, a distribution $\mathbb{G}^\delta$ is δ non-$\mathbb{G}$ if $s' \sim \mathbb{G}^\delta$ satisfies

$$\Pr[S \subseteq \mathrm{parts}(s')] \leq 2^{\delta|S|} \cdot \Pr[S \subseteq \mathrm{parts}(s)]$$

for all $S \in \Omega_{\mathrm{parts}}(N)$.

Similarly, a distribution $\mathbb{G}^{p,\delta}$ is (p, δ) non-$\mathbb{G}$ if there is a subset $S' \in \Omega_{\mathrm{parts}}(N)$ of size at most $|S'| \leq p$ such that conditioned on $S' \subseteq \mathrm{parts}(s)$, $s'' \sim \mathbb{G}^{p,\delta}$ satisfies

$$\Pr[S \subseteq \mathrm{parts}(s'')|S' \subseteq \mathrm{parts}(s'')] \leq 2^{\delta|S'|} \cdot [S \subseteq \mathrm{parts}(s)|S' \subseteq \mathrm{parts}(s)]$$

for all $S \in \Omega_{\mathrm{parts}}(N, S')$, i.e. conditioned on S' is a part of both s and s'', s'' is δ non-$\mathbb{G}$.

We now define β-uniform as motivated above and using the previous definition, define δ non-β-uniform.

Definition 56 (β-uniform and δ non-β-uniform distributions—$\mathbb{F}^{|\beta}$ and $\mathbb{F}^{\delta|\beta}$). Let $u \sim \mathbb{F}(N)$ be sampled from a uniform distribution over all permutations, $\Omega(N)$, of $\{0, 1 \ldots N - 1\}$ as in Notation 49. A permutation $s \sim \mathbb{F}^{|\beta}(N)$ sampled from a β-uniform distribution is $s = u|(S \subseteq \mathrm{parts}(u))$ where[21] $\beta =: (S)$ and $S \in \Omega_{\mathrm{parts}}(N)$.

A distribution $\mathbb{F}^{\delta|\beta}$ is δ non-β-uniform if it is δ non-$\mathbb{G}$ with $\mathbb{G}$ set to a β-uniform distribution (see Definition 55, above). Similarly, a distribution $\mathbb{F}^{p,\delta|\beta}$ is (p, δ) non-β-uniform if it is (p, δ) non-$\mathbb{G}$ with $\mathbb{G}$, again, set to a β-uniform distribution.

We now state the general version of Proposition 51.

Proposition 57 ($\mathbb{F}^{\delta'|\beta}|r' = \mathrm{conv}(\mathbb{F}^{(p,\delta+\delta')|\beta})$). Let $t \sim \mathbb{F}^{\delta'|\beta}(N)$ be sampled from a δ' non-β-uniform distribution with $N = 2^n$. Fix any $\delta > 0$ and let $\gamma = 2^{-m}$ be some function of n. Let $s \sim \mathbb{F}^{\delta'|\beta}|r$, i.e. $s = t|(h(t) = r)$ and suppose $\Pr[h(t) = r] \geq \gamma$ where h is an arbitrary function and r some string in its range. Then s is "γ-close" to a convex combination of finitely many $(p, \delta + \delta')$ non-β-uniform distributions, i.e.

$$s \equiv \sum_i \alpha_i s_i + \gamma' s'$$

where $s_i \sim \mathbb{F}_i^{p,\delta+\delta'|\beta}$ with $p = 2m/\delta$. The permutation s' may have an arbitrary distribution (over $\Omega(2^n)$) but $\gamma' \leq \gamma$.

The proof follows from minor modifications to that of Proposition 51 and is thus deferred to Subsection B.2.

---

[20] The last term with $\alpha_k < \gamma$ is suppressed for clarity in this informal discussion.

[21] As alluded to earlier, we define β to be a redundant-looking "one-tuple" (S) here but this is because later when we generalise to d-Shufflers, we set $\beta = (S, T)$ where T encodes paths not in u.

6.2 Sampling argument for the d-Shuffler

Our objective in this section is to state the analogue of Proposition 57 for the d-Shuffler. To this end, we first define a more abstract notation for the d-Shuffler which builds upon Definition 36.

Notation 58 (Abstract notation for d-Shufflers). Represent a uniform d-Shuffler sampled from $\mathbb{F}_{\text{shuff}}(d, n, f)$ (see Definition 36) abstractly as Ξ. Denote by

- $\text{func}_i(\Xi)$ the function f_i, for $i \in \{0, 1 \dots d\}$, which correspond to the first definition,

- $\text{tup}(\Xi)$ the tuples $(t_0, \dots t_d)$, which correspond to the second definition,

  - $\text{dom}_i(\Xi)$ the unordered set corresponding to the tuple t_{i-1}; $\text{dom}(\Xi) := (\text{dom}_i(\Xi))_{i=0}^{d}$ and $\text{dom}(\Xi^*) := (\text{dom}_i(\Xi))_{i=0}^{d-1}$,

- $\text{mat}(\Xi)$ the tuples $(t_i)_{i=0}^{d}$ stacked as columns in a matrix,

$$
\begin{bmatrix}
0 & t_0[0] & t_1[0] & \dots & t_d[0] \\
1 & t_0[1] & t_1[1] & \dots & t_d[1] \\
\vdots & \vdots & \vdots & & \vdots \\
N-1 & t_0[N-1] & t_1[N-1] & \dots & t_d[N-1]
\end{bmatrix}
=
\begin{bmatrix}
0 & f_0(0) & f_1 f_0(0) & \dots & f_d f_{d-1} \dots f_0[0] \\
1 & f_0(1) & f_1 f_0(1) & \dots & f_d f_{d-1} \dots f_0[1] \\
\vdots & \vdots & \vdots & & \vdots \\
N-1 & f_0(N-1) & f_1 f_0(N-1) & \dots & f_d f_{d-1} \dots f_0[N-1]
\end{bmatrix},
$$

  of size $N \times (d+2)$ where $N = 2^n$, $\text{mat}(\Xi^*)$ the same matrix with the last column removed,

- $\text{paths}(\Xi)$ the set of rows of $\text{mat}(\Xi)$, i.e. $\text{paths}(\Xi) := \{\text{mat}(\Xi)[i]\}_{i=0}^{N-1}$, and similarly $\text{paths}(\Xi^*)$ the set of rows of $\text{mat}(\Xi^*)$, i.e. $\text{paths}(\Xi^*) := \{\text{mat}(\Xi^*)[i]\}_{i=0}^{N-1}$,

- $\text{parts}(\Xi)$ the power set of rows of $\text{mat}(\Xi)$, i.e. $\text{parts}(\Xi) := \mathscr{P}[\text{paths}(\Xi)] = \{\text{mat}(\Xi)[i]\}_{i=0}^{N-1}$, and similarly $\text{parts}(\Xi^*) := \mathscr{P}[\text{paths}(\Xi^*)] = \mathscr{P}\{\text{mat}(\Xi^*)[i]\}_{i=0}^{N-1}$,

- $\beta(\Xi)$ the ordered set $(\text{parts}(\Xi^*), \text{dom}(\Xi^*), \text{parts}(\Xi), \text{dom}(\Xi))$.

Call Ξ^* an empty d-Shuffler because it doesn't contain any information about f (which is contained in f_d).

We can now define the notion of paths and parts for an empty d-Shuffler, the analogue of Notation 45. To keep the notation simple, we overload the symbol Ω_{parts}. We use $\Omega_{\text{parts}}(d, n)$.

Notation 59. Further, denote by $\boldsymbol{x} = (x_{-1}, x_0, \dots x_{d-1})$ a tuple of $d+1$ elements. Use $\boldsymbol{x}[i]$ to denote the ith element of the tuple, i.e. x_i.

- *Parts*: Consider a set $S = \{\boldsymbol{x}_j\}_{j=0}^{M}$ with $M \leq N$ containing mappings of $\{\boldsymbol{x}_j[-1]\}_j$ specified by some empty d-Shuffler, i.e. there is some d-Shuffler Ξ such that $S \subseteq \text{parts}(\Xi^*)$. Call any such set S a "part" and its constituents "paths".

  - Denote by $\Omega_{\text{parts}}(d, n)$ the set of all such sets S, i.e. the set of all "parts".
  - Call two parts S and S' *distinct* if $S \cap S' = \emptyset$ and there is some d-Shuffler Ξ such that $S \cup S' \subseteq \text{parts}(\Xi^*)$.
  - Denote by $\Omega_{\text{parts}}(d, n, S)$ the set of all parts $S' \in \Omega_{\text{parts}}(d, n)$ such that S' is distinct from S.

- *Parts in Ξ^**: The probability that Ξ^* maps the elements as described by S may be expressed as $\Pr[\wedge_j \boldsymbol{x}_j \in \text{parts}(\Xi^*)] = \Pr[S \subseteq \text{parts}(\Xi^*)]$.

Example 60. Let $N = 2^n = 2$ and $d = 2$. Then an empty d-Shuffler Ξ^* can take the following values

$$
\text{mat}(\Xi^*) \in \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 2 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 2 \\ 1 & 2 & 0 \end{bmatrix} \dots \right\}.
$$

Let $(t_0, t_1) := \text{tup}(\Xi^*)$. Then the first of these corresponds to $t_0 = t_1 = \mathbb{I}$ (identity permutation) and $f_0 = f_1 = \mathbb{I}$, the second to $t_0 = \mathbb{I}$, $t_1 = \mathbb{X}$ ("swap") and $f_0 = \mathbb{I}, f_1 = \mathbb{X}$, the third to $t_0 = \mathbb{X}$, $t_1 = \mathbb{I}$ and $f_0 = \mathbb{X}$, $f_1 = \mathbb{X}$, the fourth to $t_0 = \mathbb{X}$, $t_1 = \mathbb{X}$ and $f_0 = \mathbb{X}$, $f_1 = \mathbb{I}$, and so on. The set of parts is $\Omega_{\text{parts}}(d, n) = \{\mathscr{P}\{(0, 0, 0), (1, 1, 1)\}, \mathscr{P}\{(0, 0, 1), (1, 1, 0)\}, \mathscr{P}\{(0, 1, 0), (1, 0, 1)\}, \mathscr{P}\{(0, 1, 1), (1, 0, 0)\}, \mathscr{P}\{(0, 0, 0), (1, 2, 2)\}, \dots \}$. An example of a part is $S = \{(0, 0, 0)\}$. Parts distinct from S, i.e. elements of $\Omega_{\text{parts}}(n, d, S)$, are $\{(1, 1, 1)\}, \{(1, 2, 1)\}, \{(1, 1, 2)\} \dots$.

We can now define β-uniform distributions as uniform distributions over d-Shufflers, given some information about the d-Shuffler. Unlike the simpler case of permutations where β was simply specifying parts, we would now like to condition on both, certain parts being present in parts(Ξ) and certain locations yielding $\bot$, i.e. queries outside dom(Ξ). While this may look involved, it is simply conditioning on certain aspects of the uniform distribution over d-Shufflers.

The notion of δ non β-uniform distribution and (p, δ) non β-uniform distribution is defined analogous to the simpler case of permutations. The only subtlety is that the conditioning is only over the parts of the empty d-Shuffler, i.e. the information about the function f is not included. Why do we do this?

Suppose $d = 0$ and f is a Simon's function. Then having a (p, δ) non β-uniform distribution is pointless. The advice could just be the period of f and conditioning on even one path with collision, would mean we restrict to distributions over f with that same period (for more details, see Subsection C.1).

Thus, the notion of (p, δ) non β-uniform distributions only makes sense when the underlying distribution is not severely constrained upon some paths being revealed. This allows us to show that the output distribution given the advice and the (p, δ) non β-uniform distributions is not very different from that produced by being given access to only a β-uniform distribution.

More concretely, we show that the "finding" probability even with Ξ^*, where $\Xi \sim \mathbb{R}_{\text{shuff}}^{p,\delta|\beta}$, is exponentially suppressed so long as p and the conditions in β are poly sized. We return to this after stating the main result.

Definition 61 (β-uniform and (p, δ) non β-uniform distributions for d-Shufflers—$\mathbb{F}_{\text{shuff}}^{|\beta}$ and $\mathbb{R}_{\text{shuff}}^{p,\delta|\beta}$). Let $\Xi \sim \mathbb{F}_{\text{shuff}}(d, n, f)$ be a d-Shuffler sampled from the uniform distribution over d-Shufflers. Let $\beta = (\beta_{\text{incl}}, \beta_{\text{excl}})$ where β_{incl} specifies elements in $\beta(\Xi)$ and β_{excl} specifies elements not in $\beta(\Xi)$ (see Notation 58). A d-Shuffler $\Xi' \sim \mathbb{F}_{\text{shuff}}^{|\beta}(d, n, f)$ sampled from a β-uniform distribution over d-Shufflers is $\Xi' = \Xi|(\beta_{\text{incl}} \in \beta(\Xi) \wedge \beta_{\text{excl}} \notin \beta(\Xi))$.

A distribution $\mathbb{F}_{\text{shuff}}^{\delta|\beta}$ is δ non-β-uniform over d-Shufflers if it is δ non-$\mathbb{G}$ with $\mathbb{G}$ set to a β-uniform distribution over d-Shufflers (see Definition 55) and the conditioning is over the empty d-Shuffler's parts, i.e. parts(Ξ^*) for $\Xi \sim \mathbb{F}_{\text{shuff}}^{\delta|\beta}$. Similarly, a distribution $\mathbb{F}_{\text{shuff}}^{p,\delta|\beta}$ is (p, δ) non-β-uniform over d-Shufflers if it is (p, δ) non-$\mathbb{G}$ with $\mathbb{G}$, again, set to a β-uniform distribution over d-Shufflers (and the conditioning is over the empty d-Shuffler's parts).

As we already noted, the proof of Definition 56 does not depend on the underlying distribution $\mathbb{F}$ and the conditioning β. We can therefore lift that result directly for the case of d-Shufflers.

Proposition 62 ($\mathbb{F}_{\text{shuff}}^{\delta'|\beta}|r' = \text{conv}(\mathbb{F}_{\text{shuff}}^{(p,\delta+\delta')|\beta})$). Let $\Xi^t \sim \mathbb{F}_{\text{shuff}}^{\delta'|\beta}(d, n, f)$ be sampled from a δ' non β-uniform distribution over all d-Shufflers. Let $N = 2^n$. Fix any $\delta > 0$ and let $\gamma = 2^{-m}$ be some function of n. Let $\Xi^s \sim \mathbb{F}^{\delta'|\beta}|r$, i.e. $\Xi^s := \Xi^t|(h(\Xi^t) = r)$ and suppose that $\Pr[h(\Xi^t) = r] \geq \gamma$ where h is an arbitrary function and r some string in its range. Then Ξ^s is "γ-close" to a convex combination of finitely many $(p, \delta + \delta')$ non-β-uniform distributions over d-Shufflers, i.e.

$$\Xi^s \equiv \sum_i \alpha_i \Xi_i^s + \gamma' \Xi^{s\prime} \tag{18}$$

where $\Xi_i^s \sim \mathbb{F}_{\text{shuff},i}^{(p,\delta+\delta')|\beta}$ with $p = 2m/\delta$ (the i in $\mathbb{F}_{\text{shuff},i}^{(p,\delta+\delta')|\beta}$, indicates that each Ξ_i^s can come from a different distribution which is still $(p, \delta + \delta')$ non-β-uniform; e.g. they may be fixing different paths but their count is bounded by p). The d-Shufflers $\Xi^{s\prime}$ may have arisen from an arbitrary distribution over d-Shufflers but $\gamma' \leq \gamma$.

To show that the "finding" probability remains small even with Ξ^* when $\Xi \sim \mathbb{R}_{\text{Shuff}}^{(p,\delta)}$, the key ingredient is the following lemma. To make the β notation easier to use, we introduce the following.

Notation 63. We say $\beta = (\beta_{\text{incl}}, \beta_{\text{excl}})$ as introduced in Definition 61 is *proper* if the following holds. Let $\beta_{\text{incl}} :=: (\{x_i\}_i, (H_j)_{j=1}^{d-1}, \{y_i\}_i, (I_j)_{j=1}^d)$ (recall recall $\beta(\Xi) = (\text{parts}(\Xi^*), \text{dom}(\Xi^*), \text{parts}(\Xi), \text{dom}(\Xi))$). We require $x_i[j] \in H_j$ and $y_i[j] \in I_j$, i.e. for each path x_i (required to be in paths(Ξ^*) when $\beta \in \beta(\Xi)$), the constituent points are required to be in H_j (viz. to be in the domain dom$_i(\Xi^*)$ when $\beta \in \beta(\Xi)$) and similarly for y_i and I_j.

This requirement is obviously redundant, i.e. if β is not proper and β' is made proper by including the necessary elements in H_j and I_j, then $\Xi|\beta \in \beta(\Xi)$ is identical to $\Xi|\beta' \in \beta(\Xi)$. However, it serves to simplifying the notation. We use it below.

Lemma 64. *Let*[22] $\Xi \sim \mathbb{F}^{\delta|\beta}_{\text{Shuff}}(n,d,f)$ *(see Definition 61) and suppose that* $\beta = (\beta_{\text{incl}}, \beta_{\text{excl}})$ *is proper (as in Notation 63 above) and that it only specifies* $\text{poly}(n)$ *many paths and answers to queries, i.e. let* $\beta_{\text{incl}} =:$ $(\{x_i\}_{i=1}^K, (H_j)_{j=1}^d, \{y_i\}_{i=1}^L, (I_j)_{j=1}^d)$ *then* $K, L, |H_j|, |I_j| \leq \text{poly}(n)$ *(see Notation 58) and similarly for* β_{excl}*. Fix any* x *which is neither specified by* β_{incl}*, i.e.* $x \notin \cup_j (H_j \cup I_j)$*, nor by* β_{excl}*. Then, for each* $i \in \{1, 2 \ldots d\}$*,*

$$\Pr[x \in \text{dom}_i(\Xi)] = \Pr[\text{func}_i(\Xi)(x) \neq \perp] \leq 2^\delta \cdot \text{poly}(n) \cdot 2^{-n}.$$

Proof. Let $\Xi^u \sim \mathbb{F}_{\text{Shuff}}(n,d,f)$, $M = 2^{2n}$, $N = 2^n$. Then, it is clear that $\Pr[x \in \text{dom}_i(\Xi^u)] = N/M = 2^{-n}$ by looking at the permutation $t_{i-1} =: \text{tup}_i(\Xi^u)$. Consider points with the following domains $y_0 \in \{0, \ldots N-1\}, y_1, \ldots y_{d-1} \in \{0, \ldots M-1\}$. We express the aforementioned probability in terms of individual paths as

$$\Pr[x \in \text{dom}_i(\Xi^u)] = \Pr[\vee_{\{y_j : j \neq i\}} \{(y_0, y_1 \ldots y_{i-2}, x, y_i, \ldots y_{d-1})\} \in \text{parts}(\Xi^{u*})]$$

$$= \sum_{\{y_j : j \neq i\}} \Pr[\{(y_0, y_1 \ldots y_{i-2}, x, y_i, \ldots y_{d-1})\} \in \text{parts}(\Xi^{u*})]$$

because $\Pr[\{(y_0 \ldots, x, \ldots y_{d-1})\} \in \text{parts}(\Xi^{u*}) \wedge \{(y'_0, \ldots x, \ldots y'_{d-1})\} \in \text{parts}(\Xi^{u*})] = 0$ if for any k, s.t. $y_k \neq y'_k$. The logical AND and the sum are over the domains of y_i. Suppose $\Xi' \sim \mathbb{F}^\delta_{\text{Shuff}}(n,d,f)$. Then, using the same reasoning, one has

$$\Pr[x \in \text{dom}_i(\Xi')] = \sum_{\{y_j : j \neq i\}} \Pr[\{(y_0, y_1 \ldots y_{i-2}, x, y_i, \ldots y_{d-1})\} \in \text{parts}(\Xi'^*)]$$

$$\leq 2^\delta \sum_{\{y_j : j \neq i\}} \Pr[\{(y_0, y_1 \ldots y_{i-2}, x, y_i, \ldots y_{d-1})\} \in \text{parts}(\Xi^{u*})]$$

$$= 2^\delta \Pr[x \in \text{dom}_i(\Xi^u)] = 2^\delta \cdot 2^{-n}.$$

To obtain the main result, it suffices to observe[23] that $\Pr[x \in \text{dom}_i(\Xi^{\beta'})] \leq \text{poly}(n) \cdot 2^{-n}$ where $\Xi^{\beta'} \sim \mathbb{F}^{|\beta'}_{\text{Shuff}}(n,d,f)$ and β' only specifies polynomially many constraints. $\square$

Final remark for this section, the distribution over oracles $\mathbb{O}^{(p,\delta)|\beta}_{\text{Shuff}}$ is implicitly defined from $\mathbb{F}^{(p,\delta)|\beta}_{\text{Shuff}}$ and similarly for others.

7   *d*-Shuffled Simon's Problem (cont.)

With the sampling argument for the *d*-Shuffler in place, we are almost ready to return to establishing that *d*-SS is hard for CQ_d circuits; it remains to introduce (and mildly modify) some *d*-Shuffler notation to facilitate its use in the context of *d*-SS.

Definition 65 (Extension of Definition 38)**.** Let $\mathbb{F}^{\delta|\beta}_{\text{SS}}(d,n)$ be exactly the same as $\mathbb{F}_{\text{SS}}(d,n)$ except that $\mathbb{F}^{\delta|\beta}_{\text{Shuff}}$ is used instead of $\mathbb{F}_{\text{Shuff}}$.

Notation 66. In the previous section (see Notation 59) we used Ω_{parts} to refer to the set of $\text{parts}(\Xi^*)$ for all Ξ *d*-Shufflers in the sample space of $\mathbb{F}_{\text{Shuff}}(d,n,f)$.

- $\Omega_{\text{parts}}, \Omega_{\text{parts}^*}$.
  Now, instead we use Ω_{parts^*} for the aforementioned and Ω_{parts} for the set of $\text{parts}(\Xi)$ for all Ξ.

- Convention for paths and paths*; y and x.
  A part of Ξ, $Y \in \Omega_{\text{parts}}(d,n)$, is denoted as $Y = \{y_k\}_{k=1}^{|Y|}$ where each path y_k is a tuple of $d+2$ elements indexed from -1 to d. Similarly a part of Ξ^*, $X \in \Omega_{\text{parts}^*}(d,n)$ is denoted as $X = \{x_k\}_{k=1}^{|X|}$ where each path x_k is a tuple of $d+1$ elements indexed from -1 to $d-1$.

---

[22]If one wishes to use $\mathbb{F}^{(p,\delta)|\beta}_{\text{Shuff}}$, then one must first explicitly absorb the size p paths fixed into β'. Otherwise, one could always pick x among those paths and obtain $\Pr[x \in \text{dom}_i(\Xi)] = 1$.

[23]One has $\Pr[x \in \text{dom}_i(\Xi^{\beta'})] = N'/M' \leq N/(M - \text{poly}(n)) = 2^{-n}(1 - \text{poly}(n) \cdot 2^{-2n})^{-1} \leq 2^{-n}(1 + \text{poly}(n) \cdot 2^{-2n}) \leq \text{poly}(n) \cdot 2^{-n}$ where N' is N minus some polynomial and M' is M minus some polynomial.

7.1 Shadow Boilerplate (cont.)

We will need the following generalisation of Algorithm 41 to prove our result.

Algorithm 67 ($\bar{S}_j$ for CQ_d exclusion using d-SS). *Fix d and n.*
Input:

- $1 \leq j \leq d$,

- *a part of a d-Shuffler (i.e. a set of paths) $Y = \{\boldsymbol{y}_k\}_{k=1}^{|Y|} \in \Omega_{\text{parts}}(d, n)$ to "expose" (could be implicitly specified using β as in Notation 63 and Notation 58)*

- $((f_i)_{i=0}^d, s)$ *from the sample space $\mathbb{F}_{\text{SS}}(d, n)$.*

Output: $\bar{S}_j$, a tuple of d subsets, defined as

$$
\bar{S}_j := \begin{cases} (\emptyset, \ldots \emptyset, X_j \backslash Y_j, \ldots X_d \backslash Y_d) & j > 1 \\ (X_1 \backslash Y_1, \ldots X_d \backslash Y_d) & j = 1 \end{cases}
$$

where for each $i \in \{1, \ldots d\}$, $X_i = f_{i-1}(X_{i-1})$ with $X_0 = \{0, 1\}^n$ and $Y_i = \{\boldsymbol{y}_k[i]\}_{k=1}^{|Y|}$.

Proposition 68. *Let*

- $((f_i)_i, s) \in \mathbb{F}_{SS}(d, n)$,

- $Y = \{y_k\}_{k=1}^{|Y|} \in \Omega_{\text{parts}}(d, n)$ *be a set of paths (see Notation 66),*

- *For all $2 \leq i \leq d$, let $\mathcal{G}_1 \ldots \mathcal{G}_{i-1}$ denote the shadows of $\mathcal{F}$ (oracle corresponding to $(f_i)_i$) with respect to $\bar{S}_1 \ldots \bar{S}_{i-1}$ (constructed using Algorithm 67 with the index, Y, and $((f_i)_i, s)$ as input).*

Then, $\mathcal{G}_1 \ldots \mathcal{G}_{i-1}$ contain no information about $\bar{S}_i$.

Proof. The argument is analogous to that given for Proposition 42. $\mathcal{G}_1$ was defined using $\bar{S}_1 = (X_1 \backslash Y_1, \ldots X_d \backslash Y_d)$ and yet $\mathcal{G}_1$ contains no information $X_2 \backslash Y_2, \ldots X_d \backslash Y_d$. To see this, observe that $\mathcal{G}_1$ contains information about f_0 which in turn contains information about X_1, i.e. $f_0(X_0) = X_1$. However, the remaining sub-oracles, i.e. from 1 to d, output $\perp$ everywhere except for the paths $Y = \{\boldsymbol{y}_k\}_{k=1}^{|Y|}$. Thus, they do not reveal $X_2 \backslash Y_2, \ldots X_d \backslash Y_d$. Similarly, $\mathcal{G}_2$ is defined using $\bar{S}_2 = (\emptyset, X_2 \backslash Y_2, \ldots X_d \backslash Y_d)$ and while it contains information about X_1, X_2 and the paths Y, it contains no information about $X_3 \backslash Y_3, \ldots X_d \backslash Y_d$ (thus $\bar{S}_3$). Analogously for $\mathcal{G}_3$ and so on. $\square$

The following is a direct consequence of Lemma 64 but expressed in a form more convenient for the following discussion.

Proposition 69. *Let the premise be as in Proposition 68. Suppose the d-Shuffler is sampled from a δ non-uniform distribution, i.e. $(\Xi, s) \sim \mathbb{F}_{\text{SS}}^{\delta}$ and let $\mathcal{F}$ be the oracle associated with Ξ. Further, let $\vec{I} = (I_j)_{j=1}^d$ be the "excluded domain", i.e. $I_j \cap \text{dom}_j(\Xi) = \emptyset$, where $I_j \subseteq \{0, 1\}^{2n}$, such that $|I_j| \leq \text{poly}(n)$. Also suppose that Y satisfies $Y \subseteq \text{parts}(\Xi)$ and $|Y| \leq \text{poly}(n)$.*

Let $\boldsymbol{x} \in \Omega_{\text{paths}^*}(d, n)$ be a query (in the query domain of $\mathcal{F}$) such that the query is not in the excluded set, i.e. $\boldsymbol{x}[j] \notin I_j$ for any $j \in \{1 \ldots d\}$ and the query does not intersect with any known paths, i.e. $\boldsymbol{x} \cap Y^* = \emptyset$ (let Y^* be the set of paths in Y with the last element removed).

Then, the probability that any part of $\boldsymbol{x}$ lands on $\bar{S}_i$—given that $\bar{S}_i$ is not at locations specified by $\vec{I}$ and that the paths Y are included in the d-Shuffler—is vanishingly small, i.e. $\Pr[\boldsymbol{x} \cap \bar{S}_i \neq \emptyset | \vec{I} \wedge Y] \leq O(2^{\delta} \cdot \text{poly}(n) \cdot 2^{-n})$.

Proof sketch. Apply Lemma 64 with $\beta = (\beta_{\text{incl}}, \beta_{\text{excl}})$ where $\beta'_{\text{incl}} := (\emptyset, \emptyset, Y, \emptyset)$, β_{incl} is the proper version of β_{incl} and $\beta_{\text{excl}} := (\emptyset, \emptyset, \emptyset, \vec{I})$ (see Notation 63). $\square$

7.2 Depth lower bounds for d-Shuffled Simon's Problem (d-SS) (cont.)

7.2.1 d-Shuffled Simon's is hard for CQ_d

Theorem 70 (d-SS is hard for CQ_d). *Let $(\mathcal{F}, s) \sim \mathbb{O}_{\mathrm{SS}}(d, n)$, i.e. let $\mathcal{F}$ be an oracle for a (uniformly) random d-Shuffled Simon's problem of size n and period s. Let $C^{\mathcal{F}}$ be any CQ_d circuit (see Definition 7 and Remark 11) acting on $O(n)$ qubits, with query access to $\mathcal{F}$. Then $\Pr[s \leftarrow C^{\mathcal{F}}] \leq \mathrm{negl}(n)$, i.e. the probability that the algorithm finds the period is negligible.*

To prove the theorem, we first describe the main argument while asserting some intermediate results, which are proven separately next.

We begin with setting up the basic notation we need for the proof.

- Denote the initial state by σ_0 which is initialised to all zeros (without loss of generality).

- Recall from Notation 9 that a CQ_d circuit can be represented as $C = C_{\tilde{n}} \circ \cdots C_2 \circ C_1$ where[24] $\tilde{n} \leq \mathrm{poly}(n)$. Here, we write $C_i := \vec{U}_i \circ \mathcal{A}_{c,i}$ where $\vec{U}_i$ contains d layers of unitaries, followed by a measurement. We drop the subscript c (which stood for "classical") from $\mathcal{A}_{c,i}$ for brevity.

- We abuse the notation slightly. We drop the distinction between oracles and functions and work solely with the latter. Let $(f, s) \sim \mathbb{F}_{\mathrm{Simon}}(n)$, $((f_i)_{i=0}^d, s) \sim \mathbb{F}_{\mathrm{Shuff}}(n, d, f)$ and $\mathcal{F}$ correspond to $(f_i)_{i=0}^d$ (see Definition 23 and Definition 36, Notation 58). In the following, we use $\tilde{n}$ sets of shadow oracles each set is denoted by $\vec{\mathcal{G}}_i = (\mathcal{G}_{d,i}, \ldots \mathcal{G}_{1,i})$, one set for each C_i.

    - We write $\vec{U}_i^{\vec{\mathcal{G}}_i}$ to denote $\Pi_i \circ U_{d+1,i} \circ \mathcal{G}_{d,i} \circ U_{d,i} \circ \ldots \mathcal{G}_{1,i} \circ U_{1,i}$. We drop the "$\circ$" for brevity when no confusion arises.

    - These shadows are constructed using Algorithm 67 where the paths are specified as we outline the main argument.

- Note that, for each i, after C_i the state is classical allowing use to consider "transcripts". Parameters: Set $\delta = \Delta/\tilde{n}$, $\gamma = 2^{-m}$ where $\Delta > 0$ is some small constant and m is such that $m - \tilde{m} \geq \Omega(n)$, where $\tilde{m}$ is the length of the "advice", i.e. number of bits $\mathcal{A}_i$ sends to $\vec{U}_i$.

To simplify the notation, we assume that for each query made by a classical circuit $\mathcal{A}_{c,i}$ that results in a non-$\perp$ query, the circuit learns the entire path associated with that query. This can only improve the success probability of the CQ_d circuit C and thus it suffices to upper bound this quantity instead.

Proof. The main argument consists of two steps as usual. The first is that the output of any CQ_d circuit with access to $\mathcal{F}$ differs from that of the same circuit with access to shadow oracles $\{\vec{\mathcal{G}}_i\}_i$, with negligible probability, i.e.

$$\mathrm{B}\left[\mathcal{A}_{\tilde{n}+1}^{\mathcal{F}} \vec{U}_{\tilde{n}}^{\mathcal{F}} \mathcal{A}_{\tilde{n}}^{\mathcal{F}} \ldots \vec{U}_1^{\mathcal{F}} \mathcal{A}_1^{\mathcal{F}}(\sigma_0), \quad \mathcal{A}_{\tilde{n}+1}^{\mathcal{F}} \vec{U}_{\tilde{n}}^{\vec{\mathcal{G}}_{\tilde{n}}} \mathcal{A}_{\tilde{n}}^{\mathcal{F}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1} \mathcal{A}_1^{\mathcal{F}}(\sigma_0)\right] \leq \mathrm{poly}(n) \cdot 2^{-n} \tag{19}$$

given $d \leq \mathrm{poly}(n)$. The second is that no CQ_d circuit with access to shadow oracles can find the period with non-negligible probability, i.e.

$$\Pr[s \leftarrow \mathcal{A}_{\tilde{n}+1}^{\mathcal{F}} \vec{U}_{\tilde{n}}^{\vec{\mathcal{G}}_{\tilde{n}}} \mathcal{A}_{\tilde{n}}^{\mathcal{F}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1} \mathcal{A}_1^{\mathcal{F}}(\sigma_0)] \leq O(2^{-n}). \tag{20}$$

Step One | Using shadow oracles causes negligible change in output
We outline the proof of step one. Using the hybrid argument, one can bound the LHS of Equation (19) as

$$\leq \sum_{i=1}^{\tilde{n}} \mathrm{B}\Big[\underbrace{\mathcal{A}_{\tilde{n}+1}^{\mathcal{F}} \vec{U}_{\tilde{m}}^{\mathcal{F}} \mathcal{A}_{\tilde{m}}^{\mathcal{F}} \ldots \vec{U}_{i+1}^{\mathcal{F}} \mathcal{A}_{i+1}^{\mathcal{F}}}_{\text{can be dropped} \because \text{ monotonicity B}} \quad \vec{U}_i^{\mathcal{F}} \mathcal{A}_i^{\mathcal{F}} \vec{U}_{i-1}^{\vec{\mathcal{G}}_i} \mathcal{A}_{i-1}^{\mathcal{F}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1} \mathcal{A}_1^{\mathcal{F}},$$

$$\overbrace{\mathcal{A}_{\tilde{n}+1}^{\mathcal{F}} \vec{U}_{\tilde{m}}^{\mathcal{F}} \mathcal{A}_{\tilde{m}}^{\mathcal{F}} \ldots \vec{U}_{i+1}^{\mathcal{F}} \mathcal{A}_{i+1}^{\mathcal{F}}} \quad \vec{U}_i^{\vec{\mathcal{G}}_i} \mathcal{A}_i^{\mathcal{F}} \vec{U}_{i-1}^{\vec{\mathcal{G}}_i} \mathcal{A}_{i-1}^{\mathcal{F}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1} \mathcal{A}_1^{\mathcal{F}}\Big]$$

$$\leq \sum_{i=1}^{\tilde{n}} \mathrm{B}\left[\vec{U}_i^{\mathcal{F}} \mathcal{A}_i^{\mathcal{F}} \vec{U}_{i-1}^{\vec{\mathcal{G}}_i} \mathcal{A}_{i-1}^{\mathcal{F}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1} \mathcal{A}_1^{\mathcal{F}}, \quad \vec{U}_i^{\vec{\mathcal{G}}_i} \mathcal{A}_i^{\mathcal{F}} \vec{U}_{i-1}^{\vec{\mathcal{G}}_i} \mathcal{A}_{i-1}^{\mathcal{F}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1} \mathcal{A}_1^{\mathcal{F}}\right] \tag{21}$$

---

[24] We dropped $\mathcal{A}_{c,\tilde{n}+1}$ without loss of generality as it can be absorbed into another $C_{\tilde{n}+1}$.

where we suppressed σ_0 for brevity.

The $i = 1$ case

Begin with $i = 1$. Let $\mathcal{A}_1^{\mathcal{F}}(\sigma_0) =: \sigma_1$. Then, one has

$$\mathrm{B}(\vec{U}_1^{\mathcal{F}}(\sigma_1), \vec{U}_1^{\vec{\mathcal{G}}_1}(\sigma_1)) \le \mathrm{B}(\mathcal{F}U_{d,1} \ldots \mathcal{F}U_{1,1}(\sigma_1), \quad \mathcal{G}_{d,1}U_{d,1} \ldots \mathcal{G}_{1,1}U_{1,1}(\sigma_1))$$

$$\le \sum_{j=1}^{d} \mathrm{B}(\mathcal{F}U_{j,1} \underbrace{\mathcal{G}_{j-1,1}U_{j-1,1} \ldots \mathcal{G}_{1,1}U_{1,1}(\sigma_1)}_{:= \rho_{j-1,1}}, \qquad \text{hybrid argument}$$

$$\mathcal{G}_{j,1}U_{j,1} \overbrace{\mathcal{G}_{j-1,1}U_{j-1,1} \ldots \mathcal{G}_{1,1}U_{1,1}(\sigma_1)}).$$

Using the O2H lemma (see Lemma 19), one can bound each term in the sum as

$$\mathrm{B}(\mathcal{F}U_{j,1}\rho_{j-1,1}, \mathcal{G}_{j,1}U_{j,1}\rho_{j-1,1}) \le \sqrt{\Pr[\text{find} : U_{j,1}^{\mathcal{F}\backslash \bar{S}_{j,1}}, \rho_{j-1,1}]} \tag{22}$$

where $\bar{S}_{j,1}$ is used to define the shadow oracle $\mathcal{G}_{j,1}$ and $\bar{S}_{j,1}$ is defined momentarily.

Let $Y_1 = \{\boldsymbol{y}_{k,1}\}_{k=1}^{|Y_1|}$ denote the paths uncovered by $\mathcal{A}_1^{\mathcal{F}}$ and $\vec{I}_1 = (I_{l,1})_{l=1}^{d}$ denote the queries which yielded $\perp$, given the paths uncovered were Y_1. Note that these are random variables (even if we take $\mathcal{A}_1$ to be deterministic) correlated with $\mathcal{F}$. Define the transcript $T(\sigma_1) := (\vec{I}_1, Y_1)$. We generalise these as we proceed.

With this in place, define for each $j \in \{1, \ldots d\}$, $\bar{S}_{j,1}$ to be the output of Algorithm 67 with j, Y_1 and $((f_i)_{i=0}^d, s)$ as inputs. To bound Equation (22), condition its RHS squared, as

$$\le \sum_{\vec{I}_1, Y_1} \Pr[\vec{I}_1 \wedge Y_1] \Pr[\text{find} : U_{j,1}^{\mathcal{F}\backslash \bar{S}_{j,1}}, \rho_{j-1,1} | \vec{I}_1 \wedge Y_1]$$

$$\le \mathrm{poly}(n) \cdot 2^{-n} \cdot \sum_{\vec{I}_1, Y_1} \Pr[\vec{I}_1 \wedge Y_1] = \mathrm{poly}(n) \cdot 2^{-n}$$

where the second inequality follows from Lemma 20 which bounds the probability of finding in terms of landing queries on $\bar{S}_{j,1}$ which in turn is bounded by Proposition 69 with parameters[25] $\delta \leftarrow 0$, $\vec{I} \leftarrow \vec{I}_1$ and $Y \leftarrow Y_1$. Note that the conditions to apply Lemma 20 (via Corollary 21) are satisfied once we condition all the variables (as in Corollary 21) to $\vec{I}_1 \wedge Y_1$ and use Proposition 68 to show they are uncorrelated.[26]

The $i = 2$ case

For $i = 2$ may be proved by following the $i = 1$ case, with one key difference—the use of δ non-uniform distributions over the d-Shuffler. Let $\sigma_2 := \mathcal{A}_2^{\mathcal{F}}\vec{U}_1^{\vec{\mathcal{G}}_1}\mathcal{A}_1^{\mathcal{F}}(\sigma_0)$ and $\rho_{j-1,2} := \mathcal{G}_{j-1,2}U_{j,2} \ldots \mathcal{G}_{1,2}U_{1,2}(\sigma_2)$. Proceeding as before, one can write the $i = 2$ term of Equation (21) as

$$\mathrm{B}(\vec{U}_2^{\mathcal{F}}(\sigma_2), \vec{U}_2^{\vec{\mathcal{G}}_2}(\sigma_2)) \le \sum_{j=1}^{d} \sqrt{\Pr[\text{find} : U_{j,2}^{\mathcal{F}\backslash \bar{S}_{j,2}}, \rho_{j-1,2}]} \tag{23}$$

where, again, $\bar{S}_{j,2}$ is used to define the shadow oracle $\mathcal{G}_{j,2}$ and $\bar{S}_{j,2}$ is defined after the transcript for σ_2, $T(\sigma_2)$, is defined.

Let $T(\sigma_2) := (\vec{I}_2, Y_2, S_1, s_1, \vec{I}_1, Y_1)$ where Y_1 and $\vec{I}_1$ were defined for the $i = 1$ case. Here s_1 is the output produced by the first QNC_d circuit, conditioned on $\vec{I}_1 \wedge Y_1$ (i.e. the information learnt by the classical algorithm $\mathcal{A}_1$ being $\vec{I}_1$ and Y_1). This next step is the key difference between the two cases and to this end, it may be helpful to recall that the main random variable being conditioned is the d-Shuffler, $\mathcal{F}$ (which is abstractly referred to as Ξ). It may also be helpful to look at Example 7.2.1 before proceeding. We use Proposition 62 with the parameters $\gamma = 2^{-m}$, $\delta = \Delta/\tilde{n}$ (recall), $\delta' = 0$ and β encoding $\vec{I}_1$ and Y_1. To prove our result, one may treat the parts fixed by this conditioning process (i.e. the parts fixed in each term of Equation (18)) as a random variable which the circuit can access (as

---

[25] The δ here refers to that in the proposition and not the one we defined; excuse the minor the notational conflict.

[26] Once Y_1 are fixed, this may be viewed as a special case of the proof of Theorem 34 where the $\perp$s are known right at the start.

illustrated in Example 7.2.1). Let S_1^* be[27] the aforementioned random variable (given $s_1, \vec{I}_1, Y_1$) and note that its size $|S_1^*|$ is at most $2m/\delta \leq \text{poly}(n)$. Now $\mathcal{A}_2^{\mathcal{F}}$ takes as input $(S_1^*, s_1, \vec{I}_1, Y_1)$ so it can learn the last coordinate of the paths in S_1^*; let S_1 denote these complete paths corresponding to S_1^*. Suppose it further learns paths Y_2 (given the transcript so far) and $\vec{I}_2$ (given the transcript and Y_2). These completely specify the transcript $T(\sigma_2)$.

We may now define, for each $j \in \{1, \ldots d\}$, $\bar{S}_{j,2}$ to be the output of Algorithm 67 with j, $Y_1 \cup S_1 \cup Y_2$ and $((f_i)_i, s)$ as inputs. To bound Equation (23), one may condition each the square of each term of the RHS, $\Pr[\text{find} : U_{j,2}^{\mathcal{F}\backslash \bar{S}_{j,2}}, \rho_{j-1,2}]$, as

$$= \sum_{s_1, \vec{I}_1, Y_1} \Pr[s_1] \Pr[\vec{I}_1] \Pr[Y_1] \Pr[\text{find} : U_{j,2}^{\mathcal{F}\backslash \bar{S}_{j,2}}, \rho_{j-1,2} | Y_1 \wedge \vec{I}_1 \wedge s_1]$$

$$\leq \sum_{\substack{\vec{I}_2, Y_2, S_1, \vec{I}_1 Y_1 \\ s_1 : \Pr[s_1] \geq 2^{-m}}} \Pr_\delta[\vec{I}_2] \Pr_\delta[Y_2] \Pr[S_1] \Pr[s_1] \Pr[\vec{I}_1] \Pr[Y_1] \Pr_\delta[\text{find} : U_{j,2}^{\mathcal{F}\backslash \bar{S}_{j,2}}, \rho_{j-1,2} | T(\sigma_2)] + 2^{-(m-\tilde{m})}$$

$$\leq 2^\delta \cdot \text{poly}(n) \cdot 2^{-n} \cdot \sum_{\substack{\vec{I}_2, Y_2, S_1, \vec{I}_1 Y_1 \\ s_1 : \Pr[s_1] \geq 2^{-m}}} \Pr_\delta[\vec{I}_2] \Pr_\delta[Y_2] \Pr[S_1] \Pr[s_1] \Pr[\vec{I}_1] \Pr[Y_1] \quad + \quad 2^{-(m-\tilde{m})} \leq \text{negl}(n)$$

where the first step follows from the rules of conditional probabilities. The second step follows from an application of Proposition 62 where by $\Pr_\delta$ we mean that the random variable $\mathcal{F}$ is, instead of being sampled from $\mathbb{F}_{\text{Shuff}}$, is sampled from $\mathbb{F}_{\text{Shuff}}^\delta$; we needn't use β here as the conditioning is explicitly stated. The third step, as in the $i = 1$ case, follows from the application of Lemma 20 (via Corollary 21) by observing that after conditioning, the variables are uncorrelated (using Proposition 68) and by using the bound asserted by Proposition 69 with parameters $\delta \leftarrow \delta$, $\vec{I} \leftarrow \vec{I}_1 \cup \vec{I}_2$ and $Y \leftarrow Y_1 \cup Y_2 \cup S_1$.

Let C be a CQ_d circuit (for concreteness) with query access to a d-Shuffler $\mathcal{F}$, hiding the period of a random Simon's function. Let the period by s (a random variable). Then,

$$\max_C \Pr[s \leftarrow C^{\mathcal{F}}] = \max_C \sum_S \Pr[S] \cdot \Pr[s \leftarrow C^{\mathcal{F}|S}]$$

$$\leq \max_C \sum_S \Pr[S] \cdot \Pr[a \leftarrow C^{\mathcal{F}|S}(S)].$$

The general $i \in \{1, \ldots \tilde{n}\}$ case
This is a straightforward generalisation of the $i = 2$ case and hence only key steps are outlined. Let $\sigma_i :=$ $\mathcal{A}_i^{\mathcal{F}} \vec{U}_{i-1}^{\vec{\mathcal{G}}_{i-1}} \ldots \mathcal{A}_2^{\mathcal{F}} \vec{U}_1^{\vec{\mathcal{G}}_1} \mathcal{A}_1^{\mathcal{F}}(\sigma_0)$ and the transcript $T(\sigma_i) =: (\vec{I}_i, Y_i, S_{i-1}, s_{i-1}, \vec{I}_{i-1}, Y_{i-1}, \ldots S_1, s_1, \vec{I}_1, Y_1)$ where $\vec{I}_j$ encodes the locations which yielded $\perp$ when queried by $\mathcal{A}_j$ (given the transcript before that), Y_j encodes the paths uncovered by $\mathcal{A}_j$ (given the transcript before), s_j denotes the output of the jth quantum circuit (given the transcript before that) and S_j denotes the paths uncovered using Proposition 62 and allowing the circuit access to it (as in Example 7.2.1). Let $\rho_{j-1,i} := \mathcal{G}_{j-1,i} U_{j-1,i} \ldots \mathcal{G}_{1,i} U_{1,i}(\sigma_i)$. The ith term in Equation (21) can be expressed as

$$B(\vec{U}_i^{\mathcal{F}}(\sigma_i), \vec{U}_i^{\vec{\mathcal{G}}_i}(\sigma_i)) \leq \sum_{j=1}^d \sqrt{\Pr[\text{find} : U_{j,i}^{\mathcal{F}\backslash \bar{S}_{j,i}}, \rho_{j-1,i}]} \tag{24}$$

using Lemma 19, where $\vec{\mathcal{G}}_{j,i}$ is the shadow oracle defined using $\bar{S}_{j,i}$ which in turn is defined as the output of Algorithm 67 with j, $Y_1 \cup S_1 \cup \ldots Y_{i-1} \cup S_{i-1} \cup Y_i$ and $((f_i)_i, s)$ as inputs. The square of the jth term of the RHS of Equation (24), $\Pr[\text{find} : U_{j,i}^{\mathcal{F}\backslash \bar{S}_{j,i}}, \rho_{j-1,i}]$, can be bounded as

$$\leq \sum_{\substack{\vec{I}_i, Y_i, S_{i-1}, \ldots S_1, \vec{I}_1, Y_1 \\ s_{i-1} : \Pr[s_{i-1}] \geq 2^{-m}], \ldots s_1 : \Pr[s_1] \geq 2^{-m}}} \Pr_{i \cdot \delta}[\vec{I}_i] \Pr_{i \cdot \delta}[Y_i] \Pr_{(i-1) \cdot \delta}[S_{i-1}] \ldots \Pr[\vec{I}_1] \Pr[Y_1] \Pr_{i \cdot \delta}[\text{find} : U_{j,i}^{\mathcal{F}\backslash \bar{S}_{j,i}}, \rho_{j-1,i} | T(\sigma_i)] + i \cdot 2^{-(m-\tilde{m})}$$

$$\tag{25}$$

$$\leq 2^\Delta \cdot \text{poly}(n) \cdot 2^{-n} + i \cdot 2^{-(m-\tilde{m})} \leq \text{negl}(n) \tag{26}$$

---

[27] The star, as usual, indicates that the last coordinate is not specified.

where $\Pr_{i\cdot\delta}$ is evaluated with $\mathcal{F}$ sampled from $\mathbb{F}_{\text{Shuff}}^{i\cdot\delta}$ instead of $\mathbb{F}_{\text{Shuff}}$. This follows from repeated applications of Proposition 62 (with, for the kth application, $\gamma = 2^{-m}$, $\delta = \Delta/\tilde{n}$, $\delta' = (k-1)\cdot\delta$ and β encoding $\vec{I} := \vec{I}_1 \cup \vec{I}_2 \cdots \cup \vec{I}_k$ and $Y := Y_1 \cup Y_2 \cup \ldots Y_k \cup S_1 \cup S_2 \ldots S_{k-1}$). After conditioning on the transcript, the variables can be shown to be uncorrelated using Proposition 68 allowing the application of Lemma 20 (via Corollary 21). The "p" in Lemma 20 can be bounded using Proposition 69 with parameters $\delta \leftarrow i\cdot\delta$, $\vec{I} \leftarrow \vec{I}_1 \cup \ldots \vec{I}_i$ and $Y \leftarrow Y_1 \cup \ldots Y_i \cup S_1 \cup \ldots S_{i-1}$.

Since each term in Equation (21) is bounded by $\text{poly}(n)\cdot 2^{-n}$, and there are at most polynomially many terms, one obtains the bound asserted by Equation (19).

Step Two | The problem is hard using only shadow oracles

It remains to show that Equation (20) holds. This is easily seen by observing that for the poly time classical algorithm, $\mathcal{A}_j^{\mathcal{F}}$, having access to the Simon's function directly or having access via the d-Shuffler, is equivalent.

Further, the d-depth quantum circuits $\vec{U}_j^{\vec{\mathcal{G}}_j}$ cannot reveal any more information about the Simon's function than what the classical algorithm already possessed. This is because the shadows do not contain any more information than that already possessed by the classical algorithm. Thus, the success probability of the classical algorithm is bounded by the probability of a BPP machine solving the Simon's problem. $\qquad\square$

8 d-Shuffled Collisions-to-Simon's (d-SCS) Problem « Main Result 2

We describe a problem, the d-Shuffled Collisions-to-Simon's problem, which a QC_4 circuit can solve but no $\text{CQ}_{d'}$ circuit for $d' \leq d$. However, we show this using a non-standard oracle model. We first describe this model.

8.1 Intrinsically Stochastic Oracle

We define an *Intrinsically Stochastic Oracle* to be a standard oracle except that it is allowed to sample (and use) new instances of a random variable, each time it is queried. We begin with an example. Suppose O takes no input and produces a uniformly random bit b. Two identical copies O_1 and O_2 of O, when queried, have a probability $1/2$ of agreeing. To contrast with the standard oracle model, let c be a random bit sampled uniformly at random. Let Q be a (standard) oracle which outputs c. Multiple copies of Q would output the same c.

Classically, the latter (i.e. the standard oracle) makes sense, because one can pull out the randomness from all reasonable models of computations into a random string specified at the beginning. Thus one can make any implementation of an oracle repeatable, making the aforesaid notion of intrinsically stochastic oracles questionable at best. Quantumly, however, it is conceivable that a process produces superpositions and a part of this is concealed from the circuit making the query; for instance, perhaps it is somehow scrambled so that it is effectively unavailable to the circuit. In particular, this can be used to simulate the measurement of a quantum superposition which results in a random outcome, each time the measurement is performed; even if the process is repeated identically.

We should note that non-standard oracles have been used before to prove separations between complexity classes. An example is the so-called quantum oracle used to prove a separation between the classes QMA and QCMA [AK07].

Definition 71 (Intrinsically Stochastic Oracle). Let X, Y be finite sets and let $\mathbb{F}_Y$ be some distribution over Y. Let $g(x,y) : X \times Y \to Z$ be a function. An *intrinsically stochastic oracle (ISO) O with respect to $\mathbb{F}_Y$*, corresponding to g is defined by its action at each query: O samples $y \sim \mathbb{F}_Y$ and on input $|x\rangle |z\rangle$, produces $|x\rangle |z \oplus g(x,y)\rangle$. Its action on a superposition query is defined by linearity as in the standard oracle model.

Remark 72. Note that for superposition queries, the same y is used for each part constituting the superposition. A possible quantum realisation of O could be the following: $O |x\rangle_Q |z\rangle_R |0\rangle_{R'}$ produces $\sum_y \frac{1}{\sqrt{\Pr[y]}} |x\rangle_Q |z \oplus g(x,y)\rangle_R |y\rangle_{R'}$ and scrambling the last register, R', effectively tracing it out and producing a mixed state. Or, one could imagine that $O |x\rangle_Q |z\rangle_R$ produces the same state except that the oracle holds R'. This can arise naturally in an interactive setting.

8.2 Oracles and Distributions | The d-SCS Problem

When access to the Simon's function is restricted via a d-Shuffler, we saw that the problem is hard for both CQ_d and QC_d. Here, we consider a variant of this restriction. The basic idea is that one is given access to a 2-to-1 function, f (not necessarily a Simon's function). For each colliding pair, one can associate exactly one colliding pair of some Simon's function g. The objective is to find the period of g but access to g is not provided directly. Instead, one

is given restricted access to a mapping which takes colliding pairs of f to colliding pairs of g. The details of this restriction, which use the d-Shuffler, determine the hardness of the problem. In one case, we essentially reduce to d-SS while in another, we obtain d-Shuffled Collisions-to-Simon's (d-SCS).

We proceed more precisely now, beginning with a few definitions.

Definition 73 (Distribution for $2 \rightarrow 1$ functions)**.** Let F be a set of all functions $f : \{0,1\}^n \rightarrow \{0,1\}^n$ such that for each $x_0 \in \{0,1\}^n$, there is exactly one $x_1 \in \{0,1\}^n$ (distinct from x_0) such that $f(x_0) = f(x_1)$. Define the *uniform distribution over $2 \rightarrow 1$ functions*, $\mathbb{F}_{2\rightarrow1}(n)$, to be the uniformly random distribution over F and denote the corresponding oracle distribution by $\mathbb{O}_{2\rightarrow1}(n)$.

Definition 74 (Collisions-to-Simons (CS) map)**.** Consider any 2-to-1 function $f : \{0,1\}^n \rightarrow \{0,1\}^n$, and any Simon's function $g : \{0,1\}^n \rightarrow \{0,1\}^n$. Suppose by $f^{-1}(y)$ we denote pre-images under f listed in the ascending order (and similarly for $g^{-1}(z)$). We call a bijective map $p : \{0,1\}^n \rightarrow \{0,1\}^n$ a *Collisions to Simons map* from f to g, if for all $k \in \{1, \ldots, 2^n/2\}$, it maps the pre-images $f^{-1}(y)$ to the pre-images $g^{-1}(z)$ where y is the kth largest value of f and z the kth largest value of g. Denote the inverse map by p_{inv}.

Remark 75. Note that $f(x_0)$ may not equal $g(p(x_0))$.

If one were to give access to p and p^{-1} only via a d-Shuffler, this problem would essentially reduce to d-SS because without access to p, the Simon's function g is effectively hidden and finding its period with non-vanishing probability is impossible using both QC_d and CQ_d circuits.

We therefore consider the following restriction. One is given access to a 2-to-1 function $f : \{0,1\}^n \rightarrow \{0,1\}^n$ via a stochastic oracle which allows a quantum circuit to hold two colliding pre-images in superposition. Further, instead of the CS map p (and p_{inv}), one is given access to an "encrypted" CS map p' (and p'_{inv}). Further, one can access a random permutation $h : \{0,1\}^n \rightarrow \{0,1\}^n$ via a d-Shuffler. The role of p' is to allow evaluation of $p(x)$ only when $h(f(x))$ is also accompanied with the input. The idea is that while holding a superposition of images, one needs a $(d+1)$-depth computation to evaluate h which a QC_d circuit can but a CQ_d circuit cannot. We now describe the problem formally.

Definition 76 (d-SCS distribution)**.** Define the *d-Shuffled Collisions-to-Simons Function distribution* $\mathbb{F}_{\text{SCS}}(n)$ by its sampling procedure.

- *The two-to-one function, Simons function and the CS map:* Sample a random $2 \rightarrow 1$ function, $f \sim \mathbb{F}_{2\rightarrow1}(n)$ (see Definition 73) and a random Simon's function, $(g, s) \sim \mathbb{F}_{\text{Simons}}(n)$ (see Definition 23). Let p be the Collisions-to-Simons map for f and g.

- *Stochastic Oracle for f:* Let $Y = \{y : f(x) = y, x \in \{0,1\}^n\}$ and denote by $\mathbb{F}_Y$ the uniform distribution over Y. Let $b \in \{0,1\}$. Define $\mathcal{S}$ to be a stochastic oracle wrt $\mathbb{F}_Y$ corresponding to the function $g(b) = (f^{-1}(y)[b], y)$, i.e. $\mathcal{S}\left(|0\rangle_Q + |1\rangle_Q\right)|0\rangle_{RR'} = \left(|0\rangle_Q |x_0\rangle_R + |1\rangle_Q |x_1\rangle_R\right)|y\rangle_{R'}$ where $f^{-1}(y) = (x_0, x_1)$ and y is stochastic.

- *Random permutation hidden in a d-Shuffler:* Sample $h \sim \mathbb{F}_R(n)$ from the uniformly random distribution over one-to-one functions (see Definition 22) and let $\Xi \sim \mathbb{F}_{\text{Shuff}}(d, n, h)$ be a d-Shuffler (see Definition 36) encoding h.

- *Encrypted CS map:* Define $p' : \{0,1\}^{2n} \rightarrow \{0,1\}^n \cup \{\bot\}$ to be $p'(h(f(x)), x) = p(x)$ when the input is of the form $(h(f(x)), x)$ and $\bot$ otherwise. Similarly, define $p'_{\text{inv}} : \{0,1\}^{2n} \rightarrow \{0,1\}^n \cup \{\bot\}$ to be $p'_{\text{inv}}(h(f(x)), p(x)) = p_{\text{inv}}(p(x)) = x$ when the input is of the form $(h(f(x)), p(x))$ and $\bot$ otherwise.

Return $(\mathcal{S}, \Xi, p', p'_{\text{inv}}, s)$ when $\mathbb{F}_{\text{SCS}}(n)$ is sampled.

Definition 77 (The d-SCS problem)**.** The d-Shuffled Collisions to Simons problem is defined as follows. Let $(\mathcal{S}, \Xi, p', p'_{\text{inv}}, s) \sim \mathbb{F}_{\text{SCS}}(n)$. Given oracle access to $\mathcal{S}, \Xi, p'$ and p'_{inv}, find s.

8.3 Depth Upper Bounds for d-SCS

8.3.1 QC_4 can solve the d-SCS problem

As we asserted, a QC_d circuit can solve d-SCS. We prove a stronger statement below.

Proposition 78. *The d-Shuffled Collisions-to-Simon's problem can be solved using* QC_4.

Proof. Run polynomially many copies of the following parallelly.

1. Quantumly, query $\mathcal{S}$ on a superposition $(|0\rangle + |1\rangle)_Q$ (neglecting normalisation) to obtain $(|0\rangle_Q |x_0\rangle_R + |1\rangle_Q |x_1\rangle_R) |y = f(x_0) = f(x_1)\rangle_{R'}$ for some randomly chosen y in the range of f. Measure R' to learn y.

2. Classically, compute, using the d-Shuffler Ξ, the function $h(y)$.

3. Quantumly, use the encrypted CS map p' with $h(y)$ to get

$$|0\rangle |x_0\rangle |p(x_0)\rangle + |1\rangle |x_1\rangle |p(x_1)\rangle$$

   where note that $\{p(x_0), p(x_1)\} = g^{-1}(y')$ for some y' (or equivalently, it holds that $g(p(x_0)) = g(p(x_1))$).

4. Use p'_{inv} with $h(y)$ to erase x_0 and x_1. We are left with $|0\rangle |p(x_0)\rangle + |1\rangle |p(x_1)\rangle$.

Proceed as in Simon's algorithm (Hadamard and measure; solve the equations to obtain $p(x_0) \oplus p(x_1) = s$ (since these are preimages of a collision in g which is a Simon's function)). □

8.3.2 CQ_{d+6} can also solve the d-SCS problem

It is easy to see that a CQ_{d+6} circuit can also solve d-SCS by running the d-Shuffler in the algorithm above using $d+1$ quantum depth. Note that the upper bound for d-SCS, i.e. $d+6$, is tighter than that for d-SS, i.e. $2d+1$. It remains to establish a lower bound, i.e. no CQ_d circuit can solve d-SCS with non-vanishing probability.

8.4 Depth Lower Bounds for d-SCS

8.4.1 d-SCS is hard for QNC_d

Theorem 79. *Let $(\mathcal{S}, \Xi, p', p'_{\mathrm{inv}}, s) \sim \mathbb{F}_{\mathrm{SCS}}(n)$ and let O denote the oracles $\mathcal{S}, \Xi, p', p'_{\mathrm{inv}}$ collectively. Denote an arbitrary QNC_d circuit with oracle access to O, (followed by classical post-processing) by $\mathcal{A}^O$. Then, $\Pr[s \leftarrow \mathcal{A}^O(\rho_0)] \leq \mathrm{negl}(n)$ where ρ_0 is some fixed (wrt O) initial state.*

Proof. As we saw, for instance in the proof of Theorem 44, that the function h hidden by a d-Shuffler is essentially inaccessible to QNC_d circuits (i.e. in any QNC_d circuit, one can replace the oracle with a shadow oracle (which contains no information about h) and the output distributions change at most negligibly). We assume this part of the analysis has already been performed. More precisely, we only show[28]

$$\left| \Pr[s \leftarrow \mathcal{A}^\mathcal{M}(\rho_0)] \right| \leq \mathrm{negl}(n)$$

where $\mathcal{M}$ denotes the oracles $\mathcal{S}, p', p'_{\mathrm{inv}}$. We write the circuits more explicitly as $\mathcal{A}^\mathcal{M} := \Pi \circ U_{d+1} \circ \mathcal{M} \circ U_d \dots \mathcal{M} \circ U_1$ and $\mathcal{A}^\mathcal{N} := \Pi \circ U_{d+1} \circ \mathcal{N} \circ U_d \dots \mathcal{N} \circ U_1$ where $\mathcal{N}$ denotes the oracles $\mathcal{S}, p'', p''_{\mathrm{inv}}$ where p'', p''_{inv} are defined as p', p'_{inv} except that they always output $\bot$. One can apply the hybrid argument as before to obtain

$$\left| \Pr[q \leftarrow \mathcal{A}^\mathcal{M}(\rho_0)] - \Pr[q \leftarrow \mathcal{A}^\mathcal{N}(\rho_0)] \right| \leq B(\mathcal{M}U_d \dots \mathcal{M}U_1(\rho_0), \mathcal{N}U_d \dots \mathcal{N}U_1(\rho_0))$$

$$\leq \sum_{i=1}^{d} B(\mathcal{M}U_i(\rho_{i-1}), \mathcal{N}U_i(\rho_{i-1}))$$

$$\leq \sum_{i=1}^{d} \sqrt{2 \Pr[\mathrm{find} : U_i^{\mathcal{M}\backslash \bar{X}}, \rho_{i-1}]}$$

where $\rho_{i-1} = \mathcal{N}U_{i-1} \dots \mathcal{N}U_1(\rho_0)$ and $\bar{X} = (\emptyset, X, X_{\mathrm{inv}})$ where $X, X_{\mathrm{inv}} \subseteq \{0,1\}^{2n}$ is the non-trivial domain of p', p'_{inv} respectively, i.e. for all $x \in X$, $p'(x) \neq \bot$ and similarly for all $x \in X_{\mathrm{inv}}$, $p'_{\mathrm{inv}}(x) \neq \bot$. Since ρ_{i-1} contains no information about h, one can bound $\Pr[\mathrm{find} : U_i^{\mathcal{M}\backslash \bar{X}}, \rho_{i-1}]$ using Lemma 20 by $\mathrm{poly}(n) \cdot 2^{-n}$. Further, $\Pr[s \leftarrow \mathcal{A}^\mathcal{N}(\rho_0)] \leq \mathrm{negl}(n)$ since $\mathcal{N}$ contains no information about the period s, so one cannot do better than guessing. Together, these yield the asserted result. □

---

[28]To be exact, we should use shadows for Ξ, which means the circuit may learn something about the "paths" inside the d-Shuffler (but not the value of h). As will be evident, these do not influence the analysis for QNC_d. For QC_d, we take this into account.

8.4.2 d-SCS is hard for CQ_d

We now show that CQ_d circuits solve d-SCS with at most negligible probability.

Theorem 80. *Let $(\mathcal{S}, \Xi, p', p'_{\mathrm{inv}}, s) \sim \mathbb{F}_{\mathrm{SCS}}(n)$ and let O denote the oracles $\mathcal{S}, \Xi, p', p'_{\mathrm{inv}}$. Denote an arbitrary CQ_d circuit with oracle access to O by C^O. Then, $\Pr[s \leftarrow C^O(\rho_0)] \leq \mathrm{negl}(n)$ where ρ_0 is some fixed (wrt O) initial state.*

We use the proof of Theorem 70 as a template.

- We again write the CQ_d circuit as $C = C_{\tilde{n}} \circ \cdots \circ C_1$ where $\tilde{n} \leq \mathrm{poly}(n)$ and $C_i := \vec{U}_i \circ \mathcal{A}_i$ where $\vec{U}_i$ contains d layers of unitaries, followed by a measurement and $\mathcal{A}_i$ denotes a poly time classical computation.

- Denote[29] by $\mathcal{F}$ the d-Shuffler Ξ. Denote by $\mathcal{M}$ the oracles corresponding to p', p'_{inv}. We write $C^{\mathcal{F},\mathcal{M}}$ to denote a CQ_d circuit with oracle access to both and to $\mathcal{S}$ which we do not write explicitly in this section.

- For each C_i (which together constitute C), the shadow oracles are defined differently.

  - Denote by $\mathcal{N}_i$ the oracle corresponding to the shadows of p' and p'_{inv}, with respect to $\bar{X}_i$ which is defined later.

  - Denote by $\vec{\mathcal{G}}_i = (\mathcal{G}_{d,i}, \ldots \mathcal{G}_{1,i})$ a tuple of shadow oracles of $\mathcal{F}$ with respect to $\bar{S}_i$ which is also defined later.

  - Write $\vec{U}_i^{\vec{\mathcal{G}}_i, \mathcal{N}_i}$ to denote $\Pi_i \circ U_{d+1,i} \circ (\mathcal{G}_{d,i}, \mathcal{N}_i) \circ U_{d,i} \circ \ldots (\mathcal{G}_{1,i}, \mathcal{N}_i) \circ U_{1,i}$ where $(\mathcal{G}_{k,i}, \mathcal{N}_i)$ may be viewed as a single oracle constituted by $\mathcal{G}_{k,i}$ and $\mathcal{N}_i$ (much like the sub-oracles in the d-Shuffler or d-Serial Simons[30]).

  - The shadows $\vec{\mathcal{G}}_i$ are constructed, as before, using Algorithm 67, but the way the paths are specified is slightly different. The details appear in the proof.

- The parameters, $\delta, \Delta, m, \tilde{m}$ are the same as before.

- When we write $\Pr[\mathrm{find} : U^{\mathcal{F};\mathcal{M}\backslash X}, \rho]$ or $\Pr[\mathrm{find} : U^{\mathcal{M};\mathcal{F}\backslash\bar{S}}, \rho]$, we allow U unrestricted access to the oracles before the semi-colon ";" and understand the rest in the usual $\Pr[\mathrm{find} : \ldots]$ notation (see Definition 17).

Proof. We first show that replacing $\mathcal{F}$ and $\mathcal{M}$ with their shadows makes almost no difference, i.e.

$$\mathrm{B}\left[\mathcal{A}_{\tilde{n}+1}^{\mathcal{F},\mathcal{M}} \vec{U}_{\tilde{n}}^{\mathcal{F},\mathcal{M}} \ldots \vec{U}_1^{\mathcal{F},\mathcal{M}} \mathcal{A}_1^{\mathcal{F},\mathcal{M}}(\rho_0), \quad \mathcal{A}_{\tilde{n}+1}^{\mathcal{F},\mathcal{M}} \vec{U}_{\tilde{n}}^{\vec{\mathcal{G}}_{\tilde{n}}, \mathcal{N}_{\tilde{n}}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1, \mathcal{N}_1} \mathcal{A}_1^{\mathcal{F},\mathcal{M}}(\rho_0)\right] \leq \mathrm{negl}(n) \tag{27}$$

given $d \leq \mathrm{poly}(n)$ and then we show that with the shadows, no CQ_d circuit can solve the problem with non-negligible probability, i.e.

$$\Pr\left[s \leftarrow \mathcal{A}_{\tilde{n}+1}^{\mathcal{F},\mathcal{M}} \vec{U}_{\tilde{n}}^{\vec{\mathcal{G}}_{\tilde{n}}, \mathcal{N}_{\tilde{n}}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1, \mathcal{N}_1} \mathcal{A}_1^{\mathcal{F},\mathcal{M}}(\rho_0)\right] \leq \mathrm{negl}(n). \tag{28}$$

Step one

Let $\sigma_i := \mathcal{A}_i^{\mathcal{F},\mathcal{M}} \vec{U}_{i-1}^{\vec{\mathcal{G}}_{i-1}, \mathcal{N}_{i-1}} \ldots \mathcal{A}_2^{\mathcal{F},\mathcal{M}} \vec{U}_1^{\vec{\mathcal{G}}_1, \mathcal{N}_1} \mathcal{A}_1^{\mathcal{F},\mathcal{M}}(\sigma_0)$ and the transcript

$$T(\sigma_i) := (\vec{I}_i, H_i, I_i, R_i, \quad S_{i-1}, s_{i-1}, \vec{I}_{i-1}, H_{i-1}, I_{i-1}, R_{i-1} \quad \ldots \quad S_1, s_1, \vec{I}_1, H_1, I_1, R_1)$$

where R_j encodes the tuple $(x, p(x), p_{\mathrm{inv}}(x))$ obtained by $\mathcal{A}_j$, I_j encodes the locations of p and p_{inv} which output $\perp$ when queried by $\mathcal{A}_j$ (given the transcript before that), H_j encodes the paths of $\mathcal{F}$ uncovered by $\mathcal{A}_j$ (given the transcript before), $\vec{I}_j$ encodes the locations in sub-oracles of $\mathcal{F}$ which yielded $\perp$ when queried by $\mathcal{A}_j$ (given the transcript before), s_j denotes the output of the jth quantum circuit (given the transcript before) and S_j denotes the paths uncovered by the "sampling argument" and allowing the circuit to access it (see Proposition 62 and Example 7.2.1).[31]

We make the following assumptions about the transcript, without loss of generality because these only make it easier for the circuit to solve the problem. We assume s_j contains all the stochastic values y obtained by the jth quantum part of the circuit[32] by querying $\mathcal{S}$. We assume that $\mathcal{A}_j$ learns the paths, H_j, in $\mathcal{F}$ corresponding to all

---

[29] Just notation; to preserve similarity with Theorem 70.

[30] More explicitly, we mean that one may parallelly apply polynomially many copies of $\mathcal{G}_{k,i}$ and parallelly, polynomially many copies of $\mathcal{N}_i$.

[31] We take conditionals only to make the later argument easier; it doesn't reflect or require that certain variables are known before the other.

[32] We also assume that the quantum circuit learns all tuples $(x, f(x) = y)$ learnt by the classical circuit preceding it.

the stochastic values y returned to it by $\mathcal{S}$ and those contained in s_{j-1} (it only possibly makes $\mathcal{A}$ better at solving the problem, so this is without loss of generality). Finally, we assume that if for some j, R_j contains x_0, x_1 which are collisions in f (i.e. $y = f(x_0) = f(x_1)$) and the path corresponding to y is in H_j, then for all $j' \geq j$, $R_{j'}$ is "collision-complete"—i.e. for each x in $R_{j'}$ such that the path corresponding to $y = f(x)$ is H_j, it also contains x' such that $f(x') = y$. Again, this only makes it easier for the circuit to solve the problem.[33]

Let $\rho'_{j-1,i} := (\mathcal{G}_{j-1}, \mathcal{M}) \circ U_{j-1,i} \ldots (\mathcal{G}_{1,i}, \mathcal{M}) \circ U_{1,i}(\sigma_i)$ and $\rho_{j-1,i} := (\mathcal{G}_{j-1,i}, \mathcal{N}_i) \circ U_{j-1,i} \ldots (\mathcal{G}_{1,i}, \mathcal{N}_i) \circ U_{1,i}(\sigma_i)$. Following the proof of Theorem 70, we note that to bound the LHS of Equation (27), it suffices to bound the analogue of Equation (24), i.e.

$$\mathrm{B}(\vec{U}_i^{\mathcal{F},\mathcal{M}}(\sigma_i), \vec{U}_i^{\vec{\mathcal{G}}_i,\mathcal{N}_i}(\sigma_i)) \leq \mathrm{B}(\vec{U}_i^{\mathcal{F},\mathcal{M}}(\sigma_i), \vec{U}_i^{\vec{\mathcal{G}}_i,\mathcal{M}}(\sigma_i)) + \mathrm{B}(\vec{U}_i^{\vec{\mathcal{G}}_i,\mathcal{M}}(\sigma_i), \vec{U}_i^{\vec{\mathcal{G}}_i,\mathcal{N}_i}(\sigma_i)) \tag{29}$$

$$\leq \sum_{j=1}^{d} \Pr\left[\mathrm{find} : U^{\mathcal{M};\mathcal{F}\backslash \bar{S}_{j,i}}, \rho'_{j-1,i}\right] + \sum_{j=1}^{d} \Pr\left[\mathrm{find} : U^{\vec{\mathcal{G}}_i;\mathcal{M}\backslash \bar{X}_i}, \rho_{j-1,i}\right] \tag{30}$$

where $\mathcal{G}_{j,i}$ is the shadow oracle with respect to $\bar{S}_{j,i}$ which in turn is defined the output of Algorithm 67 with j, $H_1 \cup S_1 \cup \ldots H_{i-1} \cup S_{i-1} \cup H_i$ and $((f_i)_i, s)$ as inputs (recall that Ξ was just an abstract symbol for a d-Shuffler which can be used to specify $(f_i)_i$ explicitly). We define $\bar{X}_i$ so that the shadows of $\mathcal{M}$ output $\bot$ everywhere except for the locations in H, where they behave as $\mathcal{M}$. Let $X = \{(x, p(x), h(f(x)) : x \in \{0,1\}^n\}$ and $X_{\mathrm{inv}} = \{(p(x), x, h(f(x))) : x \in \{0,1\}^n\}$. Define $\bar{X}_i = (X\backslash\{(x, p(x), h(f(x)) : (x, p(x), p_{\mathrm{inv}}(x)) \in R_j\}, X_{\mathrm{inv}}\backslash\{(p(x), x, h(f(x)) : (x, p(x), p_{\mathrm{inv}}(x)) \in R_j\})$. Recall $\mathcal{N}$ is the shadow of $\mathcal{M}$ with respect to $\bar{X}_i$. One can now proceed as we did in the proof of Theorem 70 and use Equation (26) to conclude that the first term in Equation (30) is negligible. The second term is also negligible using Equation (26) and noting that one needs $h(f(x))$ in the third argument of p' and p'_{inv} to obtain a non-$\bot$ response. This, for any y which is not already known, can be at best guessed with negligible probability (for ys which are already known, the value is already exposed in the shadow). That is because neither $\vec{\mathcal{G}}_i$ nor $\rho_{j-1,i}$ contain any information about h at the new y values (other than the set of locations where h outputs $\bot$s which rule out polynomially many locations; one can proceed as in the proof Theorem 34). Therefore the probability of "find" is negligible.

Step Two

To see that Equation (28) holds, we first define the event y-distinct as follows:

- all ys returned by $\mathcal{S}$ are distinct and

- if $\mathcal{S}$ is queried after h has been evaluated[34] for values in Y, then the y returned by $\mathcal{S}$ is not in Y.

This event happens with overwhelming (i.e. $1 - \mathrm{negl}(n)$) probability because y is stochastically chosen (so doesn't depend on any other random variable) and all excluded ys constitute a set of size at most polynomial. Conditioned on this event, we make the following observations about the classical and quantum parts of an execution of the CQ_d circuit $\mathcal{A}_{\bar{n}+1}^{\mathcal{F},\mathcal{M}} \vec{U}_{\bar{n}}^{\vec{\mathcal{G}}_{\bar{n}},\mathcal{N}_{\bar{n}}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1,\mathcal{N}_1} \mathcal{A}_1^{\mathcal{F},\mathcal{M}}(\rho_0)$.

- Classical part.

  - Observe that classical queries to $\mathcal{S}$, yield non-colliding xs. Assume that the quantum advice also yields non-colliding xs and $y = f(x)$ it learnt from $\mathcal{S}$. Only for these xs can the classical algorithm learn $p(x)$ with non-negligible probability. Since these are non-colliding, $p(x)$ contains no information about the period s. Thus, finding collisions are necessary for finding s with non-negligible probability.

  - Note also that the output, c, of the classical part cannot contain any collisions (assuming the quantum output contained no collisions) with non-negligible probability. This is because the quantum advice contains no information about f at points other than those xs (because the quantum algorithm only learnt $f(x)$ at stochastically chosen locations).

- Quantum part.

---

[33]We do this so that if the classical algorithm reveals information about the period s, then for values of p' it can access (because it learns $h(y)$), images of both x_0 and x_1 are exposed in the shadows of p' where x_0 and x_1 are such that $f(x_0) = f(x_1) = y$). Otherwise the subsequent quantum part would be able to distinguish between a shadow of p' and p' by trying $x_0 \oplus s$ for instance with the same y. Note that the classical algorithm can also not evaluate p (and similarly p_{inv}) at arbitrary xs directly because it needs $h(f(x))$ and for this it needs $f(x)$ which can only be accessed via the stochastic oracle $\mathcal{S}$ but that in turn does not take x as an input; $\mathcal{S}$ simply outputs a stochastic y and $x \in f^{-1}(y)$. Thus the probability that a classical algorithm learns a colliding pair x_0, x_1 is anyway negligible. We account for this possibility here, regardless.

[34]This will include, and therefore account for, the values in S_i (as defined in the transcript) exposed by the sampling argument.

– Suppose the classical input revealed no collision. The only way to learn a collision is by querying $\mathcal{S}$. Since we conditioned on the y-distinct event, the ys that the quantum algorithm learns, at those values $\mathcal{N}$ contains $\perp$s. Thus it could not have learnt s from these ys. It can give some output q to the next classical circuit but since finding collisions for a random $2 \to 1$ function is hard for an efficient quantum circuit (with direct oracle access [Aar01; Shi01]; here the access is even more restricted), the output, q, of the quantum part cannot contain any collisions (with non-negligible probability).

One can apply the two steps above repeatedly, starting with $\mathcal{A}_1^{\mathcal{F},\mathcal{M}}$ which receives no quantum input, to conclude that s can be learnt by $\mathcal{A}_{\tilde{n}+1}^{\mathcal{F},\mathcal{M}} \vec{U}_{\tilde{n}}^{\vec{\mathcal{G}}_{\tilde{n}},\mathcal{N}_{\tilde{n}}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1,\mathcal{N}_1} \mathcal{A}_1^{\mathcal{F},\mathcal{M}}(\rho_0)$ with at most negligible probability. More formally, denote the output of $\mathcal{A}_j^{\mathcal{F},\mathcal{M}}$ by a random variable c_j and that of $\vec{U}_j^{\vec{\mathcal{G}}_j,\mathcal{N}_j}$ by q_j. Assume (it only potentially makes the algorithms better) that q_j contains c_{j-1} and c_j contains q_{j-1} (i.e. all information is propagated to the very last algorithm). Let $\mathcal{E}$ denote any arbitrary algorithm which has no access to any oracles. Let $\text{coll}_f := \{(x_0, x_1) : f(x_0) = f(x_1) \wedge x_0 \neq x_1\}$ be the set of all colliding pre-images of f. Denote the y-distinct event by E and recall that $\Pr[\neg E] = \text{negl}(n)$. The two observations may be stated as the following.

- If $\max_{\mathcal{E}} \Pr[\mathcal{E}(q_{j-1}) \in \text{coll}_f | E] \leq \text{negl}(n)$ then, $\max_{\mathcal{E}} \Pr[\mathcal{E}(c_j) \in \text{coll}_f | E] \leq \text{negl}(n)$ and $\max_{\mathcal{E}} \Pr[s \leftarrow \mathcal{E}(c_j)|E] \leq \text{negl}(n)$.

- If $\max_{\mathcal{E}} \Pr[\mathcal{E}(c_{j-1}) \in \text{coll}_f | E] \leq \text{negl}(n)$ then, $\max_{\mathcal{E}} \Pr[\mathcal{E}(q_j) \in \text{coll}_f | E] \leq \text{negl}(n)$ and $\max_{\mathcal{E}} \Pr[s \leftarrow \mathcal{E}(q_j)|E] \leq \text{negl}(n)$.

Since $\mathcal{A}_1^{\mathcal{F},\mathcal{M}}$ receives no inputs, one can apply these repeatedly to conclude

$$\Pr[s \leftarrow \mathcal{A}_{\tilde{n}+1}^{\mathcal{F},\mathcal{M}} \vec{U}_{\tilde{n}}^{\vec{\mathcal{G}}_{\tilde{n}},\mathcal{N}_{\tilde{n}}} \ldots \vec{U}_1^{\vec{\mathcal{G}}_1,\mathcal{N}_1} \mathcal{A}_1^{\mathcal{F},\mathcal{M}}(\rho_0)] \leq \Pr(E)\Pr[s \leftarrow \mathcal{A}_{\tilde{n}+1}^{\mathcal{F},\mathcal{M}}(q_{\tilde{n}})|E] + \Pr[\neg E] \leq (1-\text{negl}(n))\cdot\text{negl}(n)+\text{negl}(n)$$

yielding Equation (28).

$\square$

References

[Aar01]   Scott Aaronson. "Quantum Lower Bound for the Collision Problem". In: (Nov. 2001). arXiv: quant-ph/0111102 [quant-ph].

[Aar05]   Scott Aaronson. *Ten semi-grand challenges for quantum computing theory*. 2005.

[AHU18]   Andris Ambainis, Mike Hamburg and Dominique Unruh. *Quantum security proofs using semi-classical oracles*. Tech. rep. 904. 2018. URL: http://eprint.iacr.org/2018/904.

[AJL06]   Dorit Aharonov, Vaughan Jones and Zeph Landau. "A Polynomial Quantum Algorithm for Approximating the Jones Polynomial". In: *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '06. 2006, pp. 427–436. ISBN: 1595931341. DOI: 10.1145/1132516.1132579. URL: https://doi.org/10.1145/1132516.1132579.

[AK07]   Scott Aaronson and Greg Kuperberg. "Quantum versus classical proofs and advice". In: *Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07)*. IEEE. 2007, pp. 115–128.

[Aru+19]   Frank Arute et al. "Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (2019), pp. 505–510. DOI: 10.1038/s41586-019-1666-5. URL: https://doi.org/10.1038/s41586-019-1666-5.

[Bra+18]   Z. Brakerski et al. "A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device". In: *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. Oct. 2018, pp. 320–331. DOI: 10.1109/FOCS.2018.00038.

[Bra+20]   Zvika Brakerski et al. "Simpler Proofs of Quantumness". In: *arXiv:2005.04826* (2020).

[BV97]   Ethan Bernstein and Umesh Vazirani. "Quantum complexity theory". In: *SIAM Journal on computing* 26.5 (1997), pp. 1411–1473.

[CCL20]   Nai-Hui Chia, Kai-Min Chung and Ching-Yi Lai. "On the Need for Large Quantum Depth". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2020. 2020, pp. 902–915. ISBN: 9781450369794. DOI: 10.1145/3357713.3384291. URL: https://doi.org/10.1145/3357713.3384291.

[CM20]   Matthew Coudron and Sanketh Menda. "Computations with Greater Quantum Depth Are Strictly More Powerful (Relative to an Oracle)". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2020. 2020, pp. 889–901. ISBN: 9781450369794. DOI: 10.1145/3357713.3384269. URL: https://doi.org/10.1145/3357713.3384269.

[Cor+17]   Sandro Coretti et al. *Random Oracles and Non-Uniformity*. Tech. rep. 937. 2017. URL: http://eprint.iacr.org/2017/937 (visited on 29/06/2021).

[CW00]   R. Cleve and J. Watrous. "Fast parallel circuits for the quantum Fourier transform". In: *Proceedings 41st Annual Symposium on Foundations of Computer Science*. Vol. 1. 2000, p. 526. DOI: 10.1109/SFCS.2000.892140. URL: https://doi.ieeecomputersociety.org/10.1109/SFCS.2000.892140.

[Göö+15]   Mika Göös et al. "Rectangles Are Nonnegative Juntas". en. In: *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing*. Portland Oregon USA: ACM, June 2015, pp. 257–266. ISBN: 978-1-4503-3536-2. DOI: 10.1145/2746539.2746596. URL: https://dl.acm.org/doi/10.1145/2746539.2746596 (visited on 23/10/2021).

[Hal02]   Sean Hallgren. "Polynomial-Time Quantum Algorithms for Pell's Equation and the Principal Ideal Problem". In: *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*. STOC '02. 2002, pp. 653–658. ISBN: 1581134959. DOI: 10.1145/509907.510001. URL: https://doi.org/10.1145/509907.510001.

[Jor]   Stephen Jordan. *The quantum algorithm zoo*. URL: https://quantumalgorithmzoo.org/.

[Joz06]   Richard Jozsa. "An introduction to measurement based quantum computation". In: *Quantum Information Processing-From Theory to Ex- periment*. Vol. 199. NATO Science Series, III: Computer and Systems Sciences. arXiv:quant-ph/05081246. 2006, pp. 137–158.

[KMR17]   Pravesh K. Kothari, Raghu Meka and Prasad Raghavendra. "Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of CSPs". en. In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. Montreal Canada: ACM, June 2017, pp. 590–603. ISBN: 978-1-4503-4528-6. DOI: 10.1145/3055399.3055438. URL: https://dl.acm.org/doi/10.1145/3055399.3055438 (visited on 23/10/2021).

[Shi01]   Yaoyun Shi. "Quantum lower bounds for the collision and the element distinctness problems". In: *Proceedings of The 43rd Annual IEEE Symposium on Foundations of Computer Science, 513 - 519, 2002* (Dec. 2001). DOI: 10.1109/SFCS.2002.1181975. arXiv: quant-ph/0112086 [quant-ph].

[Sho97]   Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM Journal on Computing* 26.5 (1997), pp. 1484–1509. DOI: 10.1137/S0097539795293172. eprint: https://doi.org/10.1137/S0097539795293172. URL: https://doi.org/10.1137/S0097539795293172.

[Sim97]   Daniel R. Simon. "On the Power of Quantum Computation". In: *SIAM Journal on Computing* 26.5 (1997), pp. 1474–1483. DOI: 10.1137/S0097539796298637. eprint: https://doi.org/10.1137/S0097539796298637. URL: https://doi.org/10.1137/S0097539796298637.

[Unr07]   Dominique Unruh. "Random Oracles and Auxiliary Input". en. In: *Advances in Cryptology - CRYPTO 2007*. Ed. by Alfred Menezes. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, pp. 205–223. ISBN: 978-3-540-74143-5. DOI: 10.1007/978-3-540-74143-5_12.

A Permutations and Combinations

Fact 81. *One has*

$$\frac{^aP_b}{^{a+1}P_{b+1}} = \frac{1}{a+1} \quad and \quad \frac{^aC_b}{^{a+1}C_{b+1}} = \frac{b}{a+1}.$$

Remark 82. Let $M \geq N$ be an integer and fix some element $x \in \{1, 2 \dots M\}$. Suppose t is a *tuple* of size N, sampled uniformly from the collection of all size N *tuples* containing distinct elements from $\{1, 2 \dots M\}$. Then

$$\Pr(x \in t) = \frac{^{M-1}P_{N-1} \cdot N}{^MP_N} = \frac{N}{M}.$$

Similarly, suppose X is a *set* of size N, sampled uniformly from the collection of all size N *subsets* of $\{1 \dots M\}$. Then, again,

$$\Pr(x \in X) = \frac{^{M-1}C_{N-1}}{^MC_N} = \frac{N}{M}.$$

B Deferred Proofs

B.1 Hardness of d-Shuffled Simon's Problem for QNC_d

Proof of Theorem 44. Suppose $((f_i)_{i=0}^d, s) \sim \mathbb{F}_{SS}(d, n)$ (see Definition 38) and let $\mathcal{F}$ be the oracle associated with $(f_i)_{i=0}^d$. Define $f : \{0, 1\}^n \to \{0, 1\}^n$ to be the Simon's function encoded in the d-Shuffler, i.e. $f(x) := f_d \circ \dots \circ f_0(x)$. Denote an arbitrary $\mathrm{QNC}_d^{\mathcal{F}}$ circuit, $\mathcal{A}^{\mathcal{F}}$, by

$$\mathcal{A}^{\mathcal{F}} := \Pi \circ U_{d+1} \circ \mathcal{F} \circ U_d \dots \mathcal{F} \circ U_2 \circ \mathcal{F} \circ U_1$$

and suppose Π corresponds to the algorithm outputting the string s. For each $i \in \{1, \dots d\}$, construct the tuples $\bar{S}_i$ using Algorithm 41. Let $\mathcal{G}_i$ be the shadow of $\mathcal{F}$ with respect to $\bar{S}_i$ (see Definition 40 and Figure 5). Define

$$\mathcal{A}^{\mathcal{G}} := \Pi \circ U_{d+1} \circ \mathcal{G}_d \circ U_d \dots \mathcal{G}_2 \circ U_2 \circ \mathcal{G}_1 \circ U_1.$$

Note that $\Pr[s \leftarrow \mathcal{A}^{\mathcal{G}}] \leq \frac{1}{2^n}$ because no $\mathcal{G}_i$ contains any information about f as f_d is completely blocked (see Figure 5). Thus, no algorithm can do better than making a random guess. We now show that the output distributions of $\mathcal{A}^{\mathcal{F}}$ and $\mathcal{A}^{\mathcal{G}}$ cannot be noticeably different using the O2H lemma (see Lemma 19).

To apply the lemma, one can use the hybrid method as before to obtain (we drop the $\circ$ symbol for brevity):

$$\left| \Pr[s \leftarrow \mathcal{A}^{\mathcal{F}}] - \Pr[s \leftarrow \mathcal{A}^{\mathcal{G}}] \right|$$
$$= |\mathrm{tr}[\Pi U_{d+1}\mathcal{F}U_d \dots \mathcal{F}U_2\mathcal{F}U_1\rho_0 - \Pi U_{d+1}\mathcal{G}_dU_d \dots \mathcal{G}_2U_2\mathcal{G}_1U_1\rho_0]|$$
$$\leq \sum_{i=1}^d \sqrt{2\Pr[\mathrm{find} : U_i^{\mathcal{F}\backslash\bar{S}_i}, \rho_{i-1}]}$$

where $\rho_0 = |0 \dots 0\rangle \langle 0 \dots 0|$ and $\rho_i = \mathcal{G}_i \circ U_i \circ \dots \mathcal{G}_1 \circ U_1(\rho_0)$ for $i > 0$. To bound the aforesaid, we apply Lemma 20. To this end, we must ensure the following. (1) The subset of queries at which $\mathcal{F}$ and $\mathcal{G}_i$ differ, i.e. $\bar{S}_i = (\emptyset, \dots \emptyset, X_i, X_{i+1} \dots X_d)$ (where $X_i = f_i \circ \dots \circ f_0(\{0, 1\}^n)$; see Algorithm 41), is uncorrelated to U_i and ρ_{i-1}. (2) The probability that a fixed query lands in $\bar{S}_i$ is at most $O(2^{-n})$. Granted these hold, since U acts on $\mathrm{poly}(n)$ many qubits, q in the lemma can be set to $\mathrm{poly}(n)$. Thus, one can bound the last inequality by $d \cdot \mathrm{poly}(n)/2^n$. Using the triangle inequality, one gets

$$\Pr[s \leftarrow \mathcal{A}^{\mathcal{F}}] \leq \frac{\mathrm{poly}(n)}{2^n}.$$

The following complete the proof.

(1) This readily follows from Proposition 42 and the observation that ρ_{i-1} has access to only $\mathcal{G}_1, \dots \mathcal{G}_{i-1}$. As for U_i, that is uncorrelated to all X_is by construction.

(2) Follows directly from Proposition 43. □

B.2 Technical results for δ non-uniform distributions

Proof of Claim 54. To see this for S_1, we proceed as before and recall the lower bound $\Pr[S_1 \subseteq \text{parts}(t_1')] > 2^{\delta|S_1|}\Pr[S_1 \subseteq \text{parts}(u)]$. The upper bound may be evaluated as

$$
\begin{aligned}
\Pr[S_1 \subseteq \text{parts}(t_1')] &= \Pr[S_1 \subseteq \text{parts}(t)|S \not\subseteq \text{parts}(t)] \\
&= \frac{\Pr[S_1 \subseteq \text{parts}(t) \wedge S \not\subseteq \text{parts}(t)]}{\Pr[S \not\subseteq \text{parts}(t)]} \\
&= \frac{\Pr[S_1 \subseteq \text{parts}(u) \wedge S \not\subseteq \text{parts}(u) \wedge g(u) = r']}{\Pr[S \not\subseteq \text{parts}(t)]\Pr[g(u) = r']} \\
&\leq \Pr[S_1 \subseteq \text{parts}(u)] \cdot \gamma^{-2}
\end{aligned}
$$

where we used $\alpha_1' = 1 - \Pr[S \subseteq \text{parts}(t)] = \Pr[S \not\subseteq \text{parts}(t)] \geq \gamma$, and $\Pr[g(u) = r'] \geq \gamma$. In the general case, suppose t_i's, t_is and S_is are as described in the proof of Proposition 51. Then, one would have

$$
\begin{aligned}
\Pr[S_i \subseteq \text{parts}(t_i')] &= \frac{\Pr[S_i \subseteq \text{parts}(u) \wedge S_{i-1} \not\subseteq \text{parts}(u) \wedge \ldots S \not\subseteq \text{parts}(u) \wedge g(u) = r']}{\Pr[S_{i-1} \not\subseteq \text{parts}(t) \wedge \ldots S \not\subseteq \text{parts}(t)]\Pr[g(u) = r']} \\
&\leq \Pr[S_i \subseteq \text{parts}(u)] \cdot \gamma^{-2}
\end{aligned} \tag{31}
$$

where $\alpha_i' = \Pr[S_{i-1} \not\subseteq \text{parts}(t) \wedge \ldots S \not\subseteq \text{parts}(t)] > \gamma$ is assumed (else there is nothing to prove). $\qquad\square$

Proposition (Proposition 57 restated with slightly different parameters). *Let $t \sim \mathbb{F}^{\delta'|\beta}(N)$ be sampled from a δ' non-β-uniform distribution with $N = 2^n$. Fix any $\delta > \delta'$ and let $\gamma = 2^{-m}$ be some function of n. Let $s = t|(h(t) = r')$ and suppose $\Pr[h(t) = r'] \geq \gamma$ where h is an arbitrary function and r' some string in its range. Then s is "γ-close" to a convex combination of finitely many (p, δ) non-β-uniform distributions, i.e.*

$$
s \equiv \sum_i \alpha_i s_i + \gamma' s'
$$

where $s_i \sim \mathbb{F}_i^{p,\delta|\beta}$ with $p = 2m/(\delta - \delta')$. The permutation s' may have an arbitrary distribution (over $\Omega(2^n)$) but $\gamma' \leq \gamma$.

Proof. While redundant, we follow the proof of Proposition 51 adapting it to this general setting and omitting full details this time.

(For comparison: We replace t with s and u with b)

Step A: Lower bound on $\Pr[S \subseteq \text{parts}(s)]$.

Let $b \sim \mathbb{F}^{|\beta}(N)$. Suppose s is not δ non-β-uniform. Then consider the largest $S \in \Omega_{\text{parts}}(N)$ such that

$$
\Pr[S \subseteq \text{parts}(s)] > 2^{\delta \cdot |S|} \cdot \Pr[S \subseteq \text{parts}(b)]. \tag{32}
$$

Claim 83. Let S and s be as described. The random variable s conditioned on being consistent with the paths in $S \in \Omega_{\text{parts}}(N)$, i.e. $s_S := s|(S \subseteq \text{parts}(s))$, is δ non-β-uniformly distributed.

We give a proof by contradiction. Suppose s_S is "more than" δ non-β-uniform. Then there exist some $S' \in \Omega_{\text{parts}}(N, S)$ such that

$$
\Pr[S' \subseteq \text{parts}(s)|S \subseteq \text{parts}(s)] > 2^{\delta \cdot |S'|}\Pr[S' \subseteq \text{parts}(b)|S \subseteq \text{parts}(b)].
$$

Then

$$
\begin{aligned}
\Pr[S \cup S' \subseteq \text{parts}(s)] &= \Pr[S \subseteq \text{parts}(s)]\Pr[S' \subseteq \text{parts}(s)|S \subseteq \text{parts}(s)] \\
&> 2^{\delta \cdot |S \cup S'|} \cdot \Pr[S \cup S' \subseteq \text{parts}(b)]
\end{aligned}
$$

using Equation (32) and Equation (13). That's a contradiction to S being maximal.

Step B: Upper bound on $\Pr[S \subseteq \text{parts}(s)]$.

Claim 84. One has $|S| < m/(\delta - \delta')$.

To see this, observe that

$$\Pr[S \subseteq \text{parts}(s)] = \Pr[S \subseteq \text{parts}(t) \wedge h(t) = r'] \cdot \Pr[h(t) = r']$$
$$\leq \Pr[S \subseteq \text{parts}(t)] \cdot \gamma^{-1}$$
$$\leq 2^{\delta'|S|} \Pr[S \subseteq \text{parts}(b)] \cdot \gamma^{-1}$$

and comparing this with the lower bound, one obtains $|S| < m/(\delta - \delta')$.

The remaining proof Proposition 51 similarly generalises by proceeding in the same vein. More concretely, suppose S_i, s_i, s_i' are defined analogously. Then the lower bound goes through almost unchanged while for the upper bound, the analogue of Equation (31) becomes

$$\Pr[S_i \subseteq \text{parts}(s_i')] = \frac{\Pr[S_i \subseteq \text{parts}(s) \wedge S_{i-1} \nsubseteq \text{parts}(s) \wedge \ldots S \nsubseteq \text{parts}(s)]}{\Pr[S_{i-1} \nsubseteq \text{parts}(s) \wedge \ldots S \nsubseteq \text{parts}(s)]}$$
$$\leq \Pr[S_i \subseteq \text{parts}(t) \wedge S_{i-1} \nsubseteq \text{parts}(t) \wedge \ldots S \nsubseteq \text{parts}(t) | h(t) = r'] \cdot \gamma^{-1}$$
$$\leq \frac{\Pr[S_i \subseteq \text{parts}(t)]}{\Pr[h(t) = r']} \cdot \gamma^{-1} \leq 2^{\delta'|S_i|} \Pr[S_i \subseteq \text{parts}(b)] \cdot \gamma^{-2}.$$

$\square$

C Discussions

C.1 Why δ non-β-uniform doesn't work for Simon's functions

We briefly discuss why the concept of δ non-β-uniform distribution does not prove useful when applied to the distribution over Simon's functions. Suppose an algorithm takes an oracle for a function f (sampled from an arbitrary β-uniform distribution) and an advice as an input. We want to argue that the algorithm would behave essentially the same if it were not given the advice. That is clearly not true in this case where f is a Simon's function. The proposition still holds, i.e. the distribution conditioned on the advice is still (p, δ) non β uniform but this conditioned distribution reveals too much information already.

To be more concrete, suppose the algorithm is supposed to verify if the period is s and output $r = 1$ if it succeeds at verifying. We can write $\Pr[\mathcal{A}(s, u|s) = r] = \sum_{t_i} \Pr(t_i) \Pr[\mathcal{A}(s, t_i) = r]$ using the main proposition where we know t_i are δ far from $\mathbb{F}_{\text{Simon}}$. Naively, one might see a contradiction. Suppose the algorithm simply checks if $u(s) = u(0)$ and outputs $r = 1$ if it is. For the LHS, the probability of outputting $r = 1$ is 1. For the RHS, it appears that because t_i are δ far from $\mathbb{F}_{\text{Simon}}^p$, $t_i(s) = u(0)$ will be much smaller than 1 because $\mathbb{F}_{\text{Simon}}$ is uniformly distributed over all Simon's functions and so a distribution δ far from $\mathbb{F}_{\text{Simon}}^p$ would also behave similarly because p fixes at most polynomially many paths. However, this reasoning is flawed because once even a single colliding path is specified, $\mathbb{F}_{\text{Simon}}^p$ can only contain functions with period s. Thus, each term in the RHS also outputs 1 with certainty.