Oracle Separations for Non-adaptive Collapse-free Quantum Computing*

Henrique Hepp[1], Murilo V. G. da Silva[1], and L. M. Zatesko[2]

[1] DINF, Federal University of Paraná, Brazil {hhepp,murilo}@inf.ufpr.br
[2] DAINF, Federal University of Technology — Paraná, Brazil zatesko@utfpr.edu.br

Abstract Aaronson et al. (2016) introduced the quantum complexity class naCQP (*non-adaptive Collapse-free Quantum Polynomial time*), also referred to as PDQP (*Product Dynamical Quantum Polynomial time*), aiming to define a class larger that BQP, but not large enough to include NP-complete problems. Aaronson et al. showed that $\mathsf{SZK} \subseteq \mathsf{naCQP}$ and that there is an oracle A for which $\mathsf{NP}^A \not\subseteq \mathsf{naCQP}^A$. We prove that: there is an oracle A for which $\mathsf{P}^A = \mathsf{BQP}^A = \mathsf{SZK}^A = \mathsf{naCQP}^A \neq (\mathsf{UP} \cap \mathsf{coUP})^A$; relative to an oracle A chosen uniformly at random, it holds $(\mathsf{UP} \cap \mathsf{coUP})^A \not\subseteq \mathsf{naCQP}^A$ with probability 1; there is an oracle A for which $\mathsf{P}^A = \mathsf{naCQP}^A \neq \mathsf{UP}^A = \mathsf{EXP}^A$. Our results are not only a strengthening of the result by Aaronson et al., but they also generalise, from BQP to naCQP, results by Bennett et al. (1997), Fortnow and Rogers (1999), and Tamon and Yamakami (2001).

Keywords: Computational Complexity · Oracle Separation · Unambiguous Polynomial-time · Collapse-free Quantum Computing.

1 Introduction

Aaronson et al. [4] introduced the complexity classes CQP (*Collapse-free Quantum Polynomial time*) and naCQP (*non-adaptive Collapse-free Quantum Polynomial time*). The former is the class of the problems that can be solved by a polynomial-time quantum algorithm allowing measurements not to cause the collapse of the states. The latter is the class CQP with the restriction that the quantum operations do not depend on the results of non-collapsing measurements. We refer the reader to Section 2 for technical details in the definition of naCQP. Note that the aim of defining such complexity classes in this line of research is not to propose alternative models of physically realizable computation, but to investigate classes of problems that seem to be larger that BQP, but not large enough to include NP-complete problems. The class naCQP is also called as PDQP (*Product Dynamical Quantum Polynomial time*) in the preprint version [3] and in two subsequent articles [2,10]. The class naCQP, or PDQP, is a subclass of the class DQP (*Dynamical Quantum Polynomial-Time*) introduced by Aaronson [1]. The

* Partially supported by CAPES (grant 001) and CNPq (grant 420079/2021-1).

class DQP was defined assuming a *hidden variable theory*, and that is possible to access in real time the evolution of the hidden variables. The complexity relation between DQP and CQP is unknown.

Aaronson et al. [4] showed that naCQP includes not only BQP, but also SZK, which is the class of the problems that admit zero-knowledge interactive proof systems, such as the Graph Isomorphism Problem. Assuming derandomisation hypotheses, we would have $\mathsf{P} = \mathsf{BPP}$ and $\mathsf{NP} = \mathsf{MA} = \mathsf{AM}$ [9], which would imply $\mathsf{SZK} \subseteq \mathsf{NP} \cap \mathsf{coNP}$. Moreover, Aaronson et al. showed that there is an oracle A for which $\mathsf{NP}^A \not\subseteq \mathsf{naCQP}^A$, which indicates that naCQP, although a superclass of BQP and SZK, seems not to be large enough to contain NP-complete problems. Remark that, since there is an oracle A for which $\mathsf{BQP}^A \not\subseteq \mathsf{NP}^A$ (in fact, $\mathsf{BQP}^A \not\subseteq \mathsf{PH}^A$ [12]), we also have $\mathsf{naCQP}^A \not\subseteq \mathsf{NP}^A$ for this oracle A.

The class UP (*Unambiguous Polynomial-Time*), a subclass of NP, is the class of problems Π for which there is a non-deterministic Turing machine M such that: if x is a positive instance of Π, then there is a single accepting computation path of M on x; if x is a negative instance of Π, then all computations of M on x end in rejection. Fortnow and Rogers [8] showed that $\mathsf{P}^A = \mathsf{BQP}^A \neq \mathsf{UP}^A \cap \mathsf{coUP}^A$ for the same oracle A. The relation between $\mathsf{UP} \cap \mathsf{coUP}$ and other complexity classes has also been explored. For instance, Menda and Watrous [11] showed that there is an oracle A for which $\mathsf{QSZK}^A \not\supseteq \mathsf{UP}^A \cap \mathsf{coUP}^A$, which implies $\mathsf{SZK}^A \not\supseteq \mathsf{UP}^A \cap \mathsf{coUP}^A$. We show the following.

Theorem 1. *There is an oracle A for which $\mathsf{P}^A = \mathsf{BQP}^A = \mathsf{SZK}^A = \mathsf{naCQP}^A \neq \mathsf{UP}^A \cap \mathsf{coUP}^A$.*

Our proof for Theorem 1 follows the structure of the proof by Fortnow and Rogers for BQP, but using properties for naCQP shown by Aaronson et al. combined with a technical lemma by Bennett et al. [7]. Remark that:

- our result is a strengthening of the result by Aaronson et al., concerning naCQP and NP, and also of the result by Fortnow and Rogers, concerning P, BQP, and $\mathsf{UP} \cap \mathsf{coUP}$;
- concerning SZK and $\mathsf{UP} \cap \mathsf{coUP}$, our result implies not only that there is an oracle A for which $\mathsf{SZK}^A \not\supseteq \mathsf{UP}^A \cap \mathsf{coUP}^A$, something which is already implied by Menda and Watrous, but also that $\mathsf{P}^A = \mathsf{SZK}^A \not\supseteq \mathsf{UP}^A \cap \mathsf{coUP}^A$.

Figure 1 illustrates the known relationship between the main complexity classes relevant to this paper, highlighting the result stated in Theorem 1.

A *random oracle* is an oracle chosen uniformly at random. Bennett et al. [7] showed that, relative to a random oracle A, we have $\mathsf{NP}^A \not\subseteq \mathsf{BQP}^A$ with probability 1. In fact, the authors further show that $(\mathsf{NP} \cap \mathsf{coNP})^A \not\subseteq \mathsf{BQP}^A$ with probability 1. We also show the following.

Theorem 2. *Relative to a random oracle A, we have $(\mathsf{UP} \cap \mathsf{coUP})^A \not\subseteq \mathsf{naCQP}^A$ with probability* 1.

The proof for Theorem 2 follows the structure of the proof by Bennett et al. for BQP, but using the same properties for naCQP that we have used in the proof of Theorem 1. Remark that:

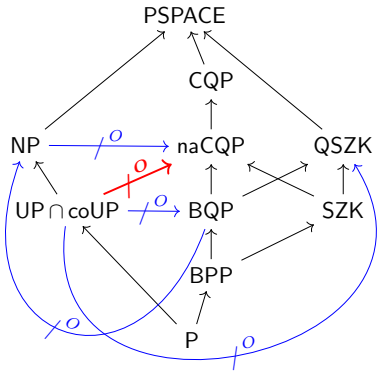

Figure 1. Relationship between complexity classes in this paper. The black arrows indicate inclusion between the classes. The striked-out arrows indicate that there is an oracle for which the class is not included in the other, being the blue ones results from the literature, and the red thicker one our new result.

- our result is a strengthening of the result by Bennett et al. concerning $\mathsf{UP} \cap \mathsf{coUP}$ and BQP;
- our result is also a strengthening of the result by Aaronson et al. that there is an oracle A for which $\mathsf{NP}^A \not\subseteq \mathsf{naCQP}^A$.

Beigel et al. [6] also showed that there is an oracle A for which $\mathsf{P}^A = \oplus\mathsf{P}^A$ and $\mathsf{RP}^A = \mathsf{EXP}^A$ (recall that $\mathsf{P}^A \neq \mathsf{EXP}^A$ for all oracle A). As noted by Tamon and Yamakami [13], since $\mathsf{UP}^A \subseteq \oplus\mathsf{P}^A$ and $\mathsf{RP}^A \subseteq \mathsf{BQP}^A$, we have $\mathsf{P}^A = \mathsf{UP}^A$ and $\mathsf{BQP}^A = \mathsf{EXP}^A$. Curiously, Tamon and Yamakami [13] also claimed that there is an oracle A for which $\mathsf{P}^A = \mathsf{BQP}^A$ and $\mathsf{UP}^A = \mathsf{EXP}^A$, presenting a proof sketch for this. We prove the following strengthening of their claim.

Theorem 3. *There is an oracle A for which $\mathsf{P}^A = \mathsf{naCQP}^A$ and $\mathsf{UP}^A = \mathsf{EXP}^A$.*

We follow the idea of Tamon and Yamakami's sketch of proof. Since their sketch is not easy to follow and contains some technical details that we are not sure that are correct, we present the full proof of our more general theorem.

Some of the properties for naCQP with which we adapt the proofs for BQP by Fortnow and Rogers, by Bennett et al., and by Tamon and Yamakami in order to prove our results, are stated in Theorem 4 below, which is a version of the well-known *BBBV Theorem* for BQP by Bennett et al. [7].

Theorem 4. *Let M^A be an naCQP algorithm with an oracle A and let $p(n)$ be the running time of M^A. Let x be an n-bit string given as input to M. For every $\varepsilon > 0$ there is a set of strings S with $|S| \leq 200(p(n))^4/\varepsilon^2$ such that, for any oracle A', if A' differs from A only on a single string y and $y \notin S$, then*

$$\left|\Pr\left[M^{A'} \text{ accepts } x\right] - \Pr\left[M^{A} \text{ accepts } x\right]\right| \leq \varepsilon\,.$$

This paper is organized as follows. In Section 2, we present the technical definition of naCQP. In Section 3, we present the proof of Theorem 4. In Section 4, we present the proofs of Theorems 1, 2, and 3.

2 Definition of naCQP and technical assumptions

Consider a quantum circuit C with n qubits defined by the following sequence of operators $C = (U_1, \mathsf{M}_1, U_2, \mathsf{M}_2, \ldots, U_R, \mathsf{M}_R)$, where each U_i is a unitary operator of n qubits, and each M_i is a quantum measurement, in the computational basis, of m_i qubits, being $0 \leq m_i \leq n$. The initial state of the circuit is $|\psi_0\rangle = |0\rangle^{\otimes n}$, and after the t-th measurement the state is $|\psi_t\rangle = \mathsf{M}_t\, U_t\, |\psi_{t-1}\rangle$, so that the states at the different stages of the circuit are given by the sequence of random variables $\{|\psi_t\rangle\}_{t=0}^{R}$, subject to a probability distribution that depends on the circuit C. Consider a procedure that takes as input the circuit C and samples the sequence $\{|\psi_t\rangle\}_{t=1}^{R}$ from this probability distribution. Then, the procedure measures, on the computational basis, the states $|\psi_t\rangle$ for each t independently. The R results of these measurements are returned, named $v_1, \ldots, v_R$. Assuming that this procedure is done in only one step, we call this procedure the *oracle* Q. Note that if non-collapsing measurements were allowed, the result of Q would be equivalent to the case wherein the states $\{|\psi_t\rangle\}$ are measured in the computational basis without collapsing, but still being $\mathsf{M}_1, \mathsf{M}_2, \ldots, \mathsf{M}_R$ collapsing measurements. Recall that it is possible that the measurements $\mathsf{M}_1, \mathsf{M}_2, \ldots, \mathsf{M}_R$ are of less than n (even possibly zero) qubits. Observe that Q samples all states $\{|\psi_t\rangle\}$ at once, yielding a *non-adaptive* model of non-collapsing measurements.

Now suppose that the oracle Q does not sample the states $\{|\psi_t\rangle\}$ at once, but, instead, the following procedure is performed. The oracle Q samples each $|\psi_t\rangle$ one at a time, but not allowing the result of one sampling interfere in the other. Since Q samples each $|\psi_t\rangle$ independently and now one at a time, we always assume that all measurements $\mathsf{M}_1, \ldots, \mathsf{M}_t$ are delayed, by the well-known Principle of Deferred Measurement (PDM), and that a single equivalent measurement occurs immediately before the sampling of $|\psi_t\rangle$. From the non-adaptiveness of the model, it is clear that the results $v_1, \ldots, v_R$ of the measurements of the states sampled by this procedure follow the same probability distribution than the results when Q samples all the states at once. Remark that this does not hold in the adaptive model, wherein the variables $\{|\psi_t\rangle\}$ are not independent. Throughout this text, therefore, we assume without loss of generality that the states are sampled independently and one at a time, with all the collapsing measurements delayed. Also, for a fixed t, we use $|\phi_t\rangle$ to denote the state immediately before the single measurement performed by Q to obtain $|\psi_t\rangle$, and $|\phi_0\rangle, \ldots, |\phi_{t-1}\rangle$ are used to denote the previous states, with no measurements performed. Moreover, although applying PDM modifies gates $U_1, \ldots, U_t$, by abuse we maintain the names $U_1, \ldots, U_t$, assuming that the gates have been modified so that the measurements are deferred.

We define naCQP (*non-adaptive Collapse-free Quantum Polynomial*) as the class of promise problems that can be solved by an naCQP *algorithm*, i.e. a

polynomial-time deterministic Turing machine with error probability less than or equal to 1/3 that can make a single query to an oracle Q as defined above. Clearly, $\mathsf{BQP} \subseteq \mathsf{naCQP}$, since we can provide the oracle with a polynomial quantum circuit and use only the measurement result at the end of the circuit.

By definition, a BQP and a naCQP algorithm are of polynomial time. However, when it is clear in the context, we abuse the definition by using the terms *$O(T(n))$-time* BQP *algorithm* and *$O(T(n))$-time* naCQP *algorithm*, even when $T(n)$ is not a polynomially bounded function. This way, we can use expressions like "an $\Omega(2^{n/2})$ lower bound for a non-structured search with a BQP algorithm [7]" without saying nonsense.

An naCQP *algorithm with an oracle query* is an naCQP algorithm whose unitary operators $U_1, \cdots, U_R$ can query a function $f : \{0,1\}^n \to \{0,1\}$, that is, for the construction of $U_1, \cdots, U_R$, we are given access to the n-qubit unitary transformation defined as $U_f : |x, b\rangle \mapsto |x, b \oplus f(x)\rangle$, for $x \in \{0,1\}^n$ and $b \in \{0,1\}$, where the operation $\oplus$ indicates addition modulo 2, or, equivalently,

$$U_f |x\rangle = (-1)^{f(x)} |x\rangle \ , \quad \text{for } x \in \{0,1\}^n.$$

Throughout this text, we assume without loss of generality that in an naCQP algorithm with an oracle query to f, each of the unitary transformations $U_1, \ldots, U_R$ is either a copy of U_f, or an n-qubit circuit constructed using only gates from a fixed finite universal gate set. Furthermore, being A a language, an naCQP *algorithm with oracle* A is an naCQP algorithm with an oracle query to the function f_A defined by $f_A(x) = 1$ if and only if $x \in A$.

Let M^A be an naCQP algorithm with oracle A, obeying our assumption on the transformations $U_1, \ldots, U_R$ and U_f as above. Being y a fixed binary string, let $q_y(|\phi_t\rangle)$ be the modulus squared of the amplitude of $|y\rangle$ in $|\phi_t\rangle$ if U_t is a copy of U_f, and 0 otherwise. Note that, since $|\phi_t\rangle$ can be viewed as a superposition of all the configurations of the algorithm at time t, we have that $q_y(|\phi_t\rangle)$ is the sum of modulus squared of amplitudes in $|\phi_t\rangle$ corresponding only to the configurations at time t which are querying the oracle A on string y.

3 Proof of Theorem 4

Before we present the proof of Theorem 4, we first need a few technical lemmas.

Lemma 1 (adapted from Bennett et al., 1997 [7]). *Let M^A be an* naCQP *algorithm with an oracle A as in Section 2. For some fixed t, let T be the number of copies of U_f amongst gates $U_1, \ldots, U_t$, and, for $\varepsilon > 0$, let $F \subseteq [1,t] \times \Sigma^*$*

1. *for each $(i,y) \in F$, U_i is a copy of U_f;*
2. *$\sum_{(i,y)\in F} q_y(|\phi_i\rangle) \leq \varepsilon^2/2T$.*

Now suppose that the answer to each query $(i,y) \in F$ is modified to some arbitrary fixed bit $a_{i,y}$, being these answers not necessarily consistent with an oracle. Let $|\phi'_i\rangle$ be defined as $|\phi_i\rangle$, but with respect to oracle A modified as above. Then, $\||\phi_t\rangle - |\phi'_t\rangle\| \leq \varepsilon$. □

The proof of Lemma 1 follows by inspection on the same proof showed by Bennett et al. for BQP, but adapting the arguments for naCQP. Recall that we have assumed that no collapsing measurements have occurred before time t, which means that, until that point, our naCQP algorithm is behaving like a BQP algorithm. Furthermore, the reader may notice that the authors originally stated $\sum_{(i,y)\in F} q_y(|\phi_i\rangle) \leq \varepsilon^2/T$, but we remark that this is a typo and it should be $\sum_{(i,y)\in F} q_y(|\phi_i\rangle) \leq \varepsilon^2/2T$.

Lemma 2 (Aaronson et al., 2016 [4]). *Let $R \geq 1$ and let $v = (v_0, \cdots, v_R)$ be a random variable governed by a Markov distribution. That is, for all $1 \leq i \leq R$, we have that v_i is independent of $v_0, \cdots, v_{i-2}$ conditioned on a particular value of v_{i-1}. Let $w = (w_0, \cdots, w_R)$ be another random variable governed by a Markov distribution. Then, the total variation distance between these random variables is $d_{TV}(v, w) \leq 2\sum_{i=1}^{R} d_{TV}\left((v_{i-1}, v_i), (w_{i-1}, w_i)\right)$.* □

Aaronson et al. [4] showed, in Theorem 6.1 in their paper, an $\Omega(2^{n/4})$ lower bound for a non-structured search with a naCQP algorithm, analogous to the well-known $\Omega(2^{n/2})$ lower bound for a non-structured search with a BQP algorithm [7]. The main argument in their proof can be summarised in what is stated below in Lemma 3.

Lemma 3. *Let M^A be an naCQP algorithm with an oracle A as in Section 2. Being A' an oracle which answers arbitrarily to any query, let v and w be the results of the non-collapsing measurements for M^A and $M^{A'}$ respectively. Let $d_i = d_{TV}((v_{i-1}, v_i), (w_{i-1}, w_i))$, then $d_i \leq 5\||\phi_i\rangle - |\phi_i'\rangle\|$.* □

Now we present Theorem 5, from which follows Corollary 1 and Theorem 4.

Theorem 5. *Let M^A be an naCQP algorithm with an oracle A as in Section 2. Let T be the number of copies of U_f amongst gates $U_1, \ldots, U_R$, and, for $\varepsilon > 0$, let $F \subseteq [1, R] \times \Sigma^*$ be a set of time-string pairs such that:*

1. *for each each $(i, y) \in F$, U_i is a copy of U_f;*
2. *$\sum_{(i,y)\in F} q_y(|\phi_i\rangle) \leq \varepsilon^2/2T$.*

Now suppose that the answer to each query $(i, y) \in F$ is modified to some arbitrary fixed $a_{i,y}$, being these answers not necessarily consistent with an oracle. Let $|\phi_i'\rangle$ be defined as $|\phi_i\rangle$, but with respect to oracle A modified as above. Then, being $v = (v_1, \ldots, v_R)$ and $w = (w_1, \ldots, w_R)$ the random variables returned by the sampling of $\{|\psi_t\rangle\}$ and $\{|\psi_t'\rangle\}$, respectively, the total variation distance between v and w is $d_{TV}(v, w) \leq 2\sum_{i=1}^{R} 5\varepsilon \leq 10R\varepsilon$.

Proof. First, let us fix some i. Since we can delay all collapsing measurements to occur immediately before the sampling of v_i and w_i, we have, by Lemma 3, $d_i = d_{TV}((v_{i-1}, v_i), (w_{i-1}, w_i)) \leq 5\||\phi_i\rangle - |\phi_i'\rangle\|$. Also, by Lemma 1, since $\sum_{(i,y)\in F} q_y(|\phi_i\rangle) \leq \varepsilon^2/2T$, we have $\| |\phi_i\rangle - |\phi_i'\rangle \| \leq \varepsilon$. Therefore, by Lemma 2,

$$d_{TV}(v, w) \leq 2\sum_{i=1}^{R} d_{TV}\left((v_{i-1}, v_i), (w_{i-1}, w_i)\right) \leq 2\sum_{i=1}^{R} 5\varepsilon = 10R\varepsilon\,. \qquad \square$$

Corollary 1. *Let M^A be an* naCQP *algorithm with an oracle A as in Section 2. Let T be the number of copies of U_f amongst gates $U_1, \ldots, U_R$. For every $\varepsilon > 0$, there is a set of strings S with $|S| \leq 200T^2R^2/\varepsilon^2$ such that, for any oracle A', if A' differs from A only on a single string y and $y \notin S$, then, being $v = (v_1, \ldots, v_R)$ and $w = (w_1, \ldots, w_R)$ the random variables returned by M^A and $M^{A'}$, we have $d_{TV}(v, w) \leq \varepsilon$.*

Proof. First, we follow the proof by Bennett et al. [7] for the analogous result concerning BQP. Since each $|\phi_i\rangle$ has unit length, $\sum_{i=1}^{R} \sum_y q_y(|\phi_i\rangle) \leq T$. Let S be the set of strings y such that $\sum_{i=1}^{R} q_y(|\phi_i\rangle) \geq \frac{\varepsilon^2}{2T}$. Since for any y in S we have $|S| \sum_{i=1}^{R} q_y(|\phi_i\rangle) \leq T$, we can conclude that $|S| \leq \frac{2T^2}{\varepsilon^2}$. If $y \notin S$ then

$$\sum_{i=1}^{R} q_y(|\phi_i\rangle) < \frac{\varepsilon^2}{2T}\,. \tag{1}$$

Now, from (1), we apply Theorem 5, obtaining that for all $y \notin S$, $d_{TV}(v, w) \leq 10R\varepsilon$. Considering $\varepsilon = \frac{\varepsilon'}{10R}$,we have

$$|S| \leq \frac{2T^2}{(\varepsilon'/10R)^2} = \frac{200T^2R^2}{\varepsilon'^2}\,.$$

Hence, for all $y \notin S$, we have $d_{TV}(v, w) \leq \varepsilon'$ as desired. □

Proof of Theorem 4. Follows directly from Corollary 1. □

4 Oracle separations for naCQP

4.1 Proof of Theorem 1

We first discuss how to use Theorem 4 to prove Theorem 1 in the same manner that Fortnow and Rogers [8] used an analogous result by Bennett et al. [7], referred to as *BBBV Theorem* in Fortnow and Rogers' paper, to prove that there is an oracle A relative to which $\mathsf{P}^A = \mathsf{BQP}^A \neq (\mathsf{UP} \cap \mathsf{coUP})^A$.

BBBV Theorem. *Let M^A be a* BQP *algorithm with an oracle A and let $p(n)$ be the running time of M^A. Let x be n-bit string given as input to M. For every $\varepsilon > 0$ there is a set of strings S with $|S| \leq 4(p(n))^2/\varepsilon^2$ such that, for any oracle A', if A' differs from A only on a single string y and $y \notin S$, then*

$$\Big|\Pr\big[M^{A'} \text{ accepts } x\big] - \Pr\big[M^{A} \text{ accepts } x\big]\Big| \leq \varepsilon\,.$$

Fortnow and Rogers first argue, by the use of the diagonalization argument of Baker–Gill–Solovay Theorem [5], that, relative to the oracle A constructed as follows, $\mathsf{BQP}^A \not\supseteq (\mathsf{UP} \cap \mathsf{coUP})^A$. Being B any PSPACE-complete language, and being C a $\mathsf{UP} \cap \mathsf{coUP}$ oracle satisfying a set of conditions defined in their paper, Fortnow and Rogers define A as $A = B \oplus C = \{0x : x \in B\} \cup \{1y : y \in C\}$.

We remark that relative to this oracle A, it holds that $\mathsf{naCQP}^A \not\supseteq (\mathsf{UP} \cap \mathsf{coUP})^A$ also by Baker–Gill–Solovay Theorem [5].

Fortnow and Rogers use BBBV Theorem to prove that $\mathsf{P}^A = \mathsf{BQP}^A$ relative to the oracle A as constructed. We point out the details that should be adapted so that the same holds for $\mathsf{P}^A = \mathsf{naCQP}^A$, thus concluding the proof of Theorem 1. Let M be a BQP^A (in our case, an naCQP^A) algorithm that runs in $O(p(n))$ time, and let W be a P^A algorithm. The authors prove that, given M, it is possible to construct W so that, with high probability, M^A accepts some input x if and only if W^A also accepts x. Since B is PSPACE-complete, it is clearly easy to have $M^B(x) = W^B(x)$. Now, concerning oracle C, recall that M can query C only for a polynomially limited amount of string lengths. Also, by the conditions imposed on the definition of the oracle, C contains only a polynomial amount of strings that can alter the computation of M, with respect to the case wherein M queries only B. Since one of the conditions imposed on C is that its string lengths are exponentially far apart, the authors argue that there is at most one string y, of a polynomially limited size ℓ, such that W can find in polynomial time all strings in C shorter than y only by querying C. If $y \notin C$, then W querying B would be able to find out if M^A accepts or rejects x. So, the only possibility for the M^A output for x to diverge from the W^A output is if $y \in C$. However, by BBBV Theorem (in our case, Theorem 4), we know that the set S of such strings y that could alter with probability greater than ε the output of M from accepting to rejecting (or vice-versa) is polynomially small: $4(p(n))^2/\varepsilon$ in BBBV Theorem for BQP; $200(p(n))^4/\varepsilon$ in our Theorem 4 for naCQP. This way, as the authors show, the problem of finding all strings in S of length ℓ is in PSPACE, so W can find these strings only by querying B and, by querying C on each one of them, find out, with high probability, whether the output of M is altered or not.

4.2 Proof of Theorem 2

The original version of Lemma 1 for BQP by Bennett et al. [7] coincides with the one that we have presented for naCQP, since in our statement we are considering a time t before which no collapsing measurements have occurred, meaning that, up to that point, our naCQP algorithm is behaving like a BQP algorithm. In this section, we discuss how Bennett et al. [7] used the original version of Lemma 1 for BQP to prove that, relative to a random oracle A, we have $(\mathsf{UP} \cap \mathsf{coUP})^A \not\subseteq \mathsf{BQP}^A$ with probability 1 (actually they prove this for $\mathsf{NP} \cap \mathsf{coNP}$, but the same proof works for $\mathsf{UP} \cap \mathsf{coUP}$). Throughout the discussion, we point the details that should be adapted so that, by the use of Theorem 5 instead of Lemma 1 for BQP, we have also $(\mathsf{UP} \cap \mathsf{coUP})^A \not\subseteq \mathsf{naCQP}^A$ with probability 1, thus concluding the proof of Theorem 2.

First, remark that an oracle A can be conveniently thought of as a length-preserving function f on Σ^*, which can be accomplished by interpreting the oracle answer on the pair (x, i) as i-th bit of $f(x)$. We can also define a permutation oracle as an length-preserving function that for each $n \geq 0$ gives a permutation

on Σ^n. Thus, for any such oracle A, let $L_A = \{y\colon$ first bit of $A^{-1}(y)$ is $1\}$, which is clearly in $(\mathsf{UP} \cap \mathsf{coUP})^A$.

Let $M^?$ be a BQP algorithm (in our case, an naCQP algorithm) which can query an oracle and runs in time at most $T(n)$. Let $T(n) = o(2^{n/3})$ (resp. $T(n) = o(2^{n/5})$). Bennett et al. show that, by sampling uniformly at random a permutation oracle A, we have with probability 1 that M^A gives the wrong answer on input 1^n, for some large enough picked n, implying that M^A does not accept L_A, which suffices to show that $(\mathsf{UP} \cap \mathsf{coUP})^A \not\subseteq \mathsf{BQP}^A$ (resp. $(\mathsf{UP} \cap \mathsf{coUP})^A \not\subseteq \mathsf{naCQP}^A$). They also argue that this probability 1, taken over the choice of a random *permutation* oracle (i.e. thought as a length-preserving function, the oracle acts as a permutation on Σ^*), remains the same when taken over the choice of a random oracle. In what follows, we briefly describe the idea of their proof that M^A gives the wrong answer on input 1^n.

Bennett et al. sample random permutations on $\{0,1\}^n$ as follows. Let x_0, x_1, ..., $x_{T(n)+1}$ be strings chosen uniformly at random in $\{0,1\}^n$. Let π_0 be a permutation uniformly sampled amongst permutations π satisfying $\pi(x_0) = 1^n$. Now, being τ the transposition (x_{i-1}, x_i), define $\pi_i = \pi_{i-1} \cdot \tau$, i.e. $\pi_i(x_i) = \pi_{i-1}(x_{i-1})$ and $\pi_i(x_{i-1}) = \pi_{i-1}(x_i)$. Clearly each π_i is a random permutation on $\{0,1\}^n$. Let $\{A_i\}$ be a sequence of permutation oracles such that $A_i(y) = A_j(y)$ if $y \notin \{0,1\}^n$, and $A_i(y) = \pi_i(y)$ if $y \in \{0,1\}^n$.

For BQP (Bennett et al.'s proof), let $|\phi_i\rangle$ and $|\phi'_i\rangle$ be the time-i superposition of $M^{A_{T(n)}}$ and $M^{A_{T(n)-1}}$, respectively, on input 1^n. For naCQP (our proof), let $\{|\psi_t\rangle\}$ and $\{|\psi'_t\rangle\}$ be the states sampled by $M^{A_{T(n)}}$ and $M^{A_{T(n)-1}}$, respectively, on input 1^n, being $v = (v_1, \ldots, v_R)$ and $w = (w_1, \ldots, w_R)$ the corresponding random variables, as in Theorem 5. Bennett et al. showed that $E\big[\,\big||\phi_{T(n)}\rangle - |\phi'_{T(n)}\rangle\big|\,\big] \leq 1/50$. We show that $d_{VT}(v,w) \leq 1/50$. From now on, we focus only on our proof, following Bennet et al.'s construction for BQP, but adapting the choice of constants in the inequalities so that our Theorem 5 can be used.

By construction, the probabilities that $1^n \in L_{A_{T(n)}}$ and $1^n \in L_{A_{T(n)-1}}$ are both exactly $1/2$. By Markov's bound, if $d_{VT}(v,w) \leq 1/50$, as we show in the sequel, then $\Pr[d_{VT}(v,w) \leq 2/25] \geq 3/4$, i.e. with probability at least $3/4$ we have $d_{VT}(v,w) \leq 2/25 < 1/3$, meaning that both $M^{A_{T(n)}}$ and $M^{A_{T(n)-1}}$ either accept or reject 1^n. Hence, with probability at least $1/4$, either $M^{A_{T(n)}}$ or $M^{A_{T(n)-1}}$ gives the wrong answer on input 1^n, implying that with probability at least $1/8$, $M^{A_{T(n)}}$ is the one to give the wrong answer (recall that $A_{T(n)}$ and $A_{T(n)-1}$ are chosen from the same distribution). This leads to the conclusion that M decides L_A with probability 0 for a uniformly random permutation oracle A.

Now we prove that $d_{VT}(v,w) \leq 1/50$. Consider that the naCQP algorithm $M^?$ queries a different oracle A_t for each $t = 1, \ldots, R$, being $\{|\overline{\psi}_t\rangle\}$ the corresponding states. Consider the set of time-string pairs $S = \{(i, x_j) : j \geq i, 0 \leq i \leq T(n)\}$. By construction, the oracle queries described above and those of $M^A_{T(n)}$ and $M^A_{T(n)+1}$ differ only on the set S. Since each x_j, for $j \geq i$, may be thought of having been randomly chosen during step j, after the superposition of oracle queries to be performed has already been written on the oracle tape, the expected query magnitude of any pair in S is at most $1/2^n$. Hence, being α the sum of

the query magnitudes after step $T(n) \geq 4$ for the elements of S,

$$E[\alpha] \leq \frac{|S|}{2^n} = \frac{\binom{T(n)+1}{2}}{2^n} \leq \frac{(T(n))^2}{2^n}\,.$$

Let ε be a random variable such that $\alpha = \frac{\varepsilon^2}{200R^2T(n)}$. Then by Theorem 5, and making a variable substitution of $\varepsilon = \frac{\varepsilon'}{10R}$, we have $d_{VT}(q,v) \leq \varepsilon$ and $d_{VT}(q,w) \leq \varepsilon$. We showed above that $E[\varepsilon^2/200R^2T(n)] = E[\alpha] \leq (T(n))^2/2^n$. But $E\left[\varepsilon/R\sqrt{200T(n)}\right]^2 \leq E\left[\varepsilon^2/200R^2T(n)\right]$. Therefore,

$$\begin{aligned} E[\varepsilon] &= \sqrt{200R^2T(n)}\,E\left[\frac{\varepsilon}{\sqrt{200R^2T(n)}}\right] \\ &\leq \sqrt{200R^2T(n)\,E\left[\frac{\varepsilon^2}{200R^2T(n)}\right]} \leq \sqrt{200R^2T(n)\frac{(T(n))^2}{2^n}}\,. \end{aligned}$$

As $T(n) \leq \frac{2^{n/5}}{200}$ and $R \leq \frac{2^{n/5}}{200}$, we have

$$E[\varepsilon] \leq \sqrt{200R^2T(n)\frac{(T(n))^2}{2^n}} \leq \sqrt{\frac{200}{200^5}} < \frac{1}{100}\,.$$

Therefore $d_{VT}(q,v) \leq E[\varepsilon] < 1/100$ and $d_{VT}(q,w) \leq E[\varepsilon] < 1/100$. It follows that $d_{VT}(v,w) < 1/50$, thus concluding the proof.

4.3 Proof of Theorem 3

We use 4-tuples $\langle\cdot,\cdot,\cdot,\cdot\rangle$ such that $|\langle a,b,c,d\rangle| > \max\{|a|,|b|,|c|,|d|\}$ for all a, b, c, d. Let $M^?$ be a naCQP algorithm which can query an oracle such that given the input $(0^j,x)$, $j \leq |x|$: runs in linear time on $j+|x|$ for any oracle; makes a query of size smaller than $|x|$; has error probability smaller than $2^{-j-|x|}$. Let $\{M_k^?\}_{k\in\mathbb{N}}$ be an enumeration of such algorithms.

We assume w.l.o.g. that every algorithm $M_k^?$ runs in linear time using the following padding argument. Let $W^?$ be an naCQP algorithm that queries an oracle and runs in time at most $|x|^c$ for some $c \in \mathbb{N}$. Then, $M_k^?$ is the naCQP algorithm that queries an oracle and that, on input $(0^{|x|}, x0^{|x|^c-|x|})$, emulates $W^?(x)$. Assuming that $M_k^?$ rejects if the input does not fit in this format, we have clearly that $M_k^?$, on input $(0^j,y)$, runs in linear time on the length of its input (i.e. $j+|y|$) and makes oracle queries only of length smaller than $|y|$. Also, we assume w.l.o.g. that if $x \in L$, then $\Pr[M_k(0^{|x|}, x0^{|x|^c-|x|}) = 1] \geq 1 - 2^{-|x|-|x|^c}$, and if $x \notin L$, then $\Pr[M_k(0^{|x|}, x0^{|x|^c-|x}) = 1] \leq 2^{-|x|-|x|^c}$. Therefore, we have that, for any oracle A, naCQP^A can be given as the set of the languages *decided*, in the sense above, by the algorithms $\{M_k^A\}_{k\in\mathbb{N}}$.

For the oracle A we want to construct, let L^A be a EXP^A-complete language and let E^A be a deterministic Turing machine which queries A that decides L^A in exponential time. We can also assume w.l.o.g that E^A runs in time $2^{|x|}$ given

input $(0^{j\leq|x|}, x)$. Given $k > 0$ and $s \in \{0,1\}^*$, we denote s_k^n as the $(k+1)$-th string in $\{0,1\}^n$. If $k = 0$ we have $s_0^n = 0^n$. We construct A such that for any sufficiently large n and all strings x of size n and all k with $0 \leq k < n$,

(a) $\Pr[M_k^A(0^{j\leq n}, x) = 1] \geq 1 - 2^{-2n} \Rightarrow \langle 0, s_k^{\lceil \log(n) \rceil}, x, 1^{n^2} \rangle \in A$;
(b) $\Pr[M_k^A(0^{j\leq n}, x) = 0] \geq 1 - 2^{-2n} \Rightarrow \langle 0, s_k^{\lceil \log(n) \rceil}, x, 1^{n^2} \rangle \notin A$;
(c) $x \in L^A \Rightarrow |\{y \in \Sigma^{n^2} : \langle 1, 1^{\lceil \log(n) \rceil}, x, y \rangle \in A\}| = 1$;
(d) $x \notin L^A \Rightarrow |\{y \in \Sigma^{n^2} : \langle 1, 1^{\lceil \log(n) \rceil}, x, y \rangle \in A\}| = 0$.

Note that by ensuring (a) and (b), we have $\mathsf{P}^A = \mathsf{naCQP}^A$, since a deterministic polynomial-time Turing machine algorithm can decide the same language as M_k^A by simply querying A about $\langle 0, s_k^{\lceil \log(n) \rceil}, x, 1^{n^2} \rangle$. On the other hand, by ensuring (c) and (d), we have $\mathsf{UP}^A = \mathsf{EXP}^A$, since a non-deterministic polynomial-time Turing machine can decide L^A by guessing a unambiguous $y \in \Sigma^{n^2}$ and querying the oracle about $\langle 1, 1^{\lceil \log(n) \rceil}, x, y \rangle$.

We say that w is a 0-string if w has the form $\langle 0, s_k^{\lceil \log(n) \rceil}, x, 1^{n^2} \rangle$ with $0 \leq k < n$; and w is a 1-string if w has the form $\langle 1, 1^{\lceil \log(n) \rceil}, x, y \rangle$ with $|y| = n$.

We construct A in stages, for each input x following the lexicographic order. At each stage, the oracle is described as a partial function σ_A. The initial stage is defined as $\sigma_A(\lambda) = 0$ for the empty string λ. For each stage x, we consider all possible strings w from $w = \langle 0, 0, x, 0 \rangle$ to $w = \langle 1, 1^{\lceil \log(n) \rceil}, x, 1^{n^2} \rangle$. Let $p(w)$ be a qubit representing the oracle's answer, as recursively defined below.

- For each string w that is neither 0-string nor 1-string: $p(w) = |0\rangle$.
- If $w \in \mathrm{dom}(\sigma_A)$, then $p(w) = |\sigma_A(w)\rangle$.
- We assign for every 1-string w a variable v_w that indicates whether w belongs to the oracle A, and do $p(w) = |v_w\rangle$. The variable v_w will be valued later.
- If w is a 0-string, we define $p(w)$ as the quantum state corresponding to the output, just before the final measurement, of $M_k^A(0^{j\leq n}, x)$. Note that, since the size of each query is smaller than $|x|$, and therefore smaller than $\sqrt{|w|}$, the answers to these queries are then already defined earlier.

To complete the construction of oracle A at stage x, we now need to value v_w when w is a 1-string. We view every 2^{n^2}-length string $\mathbf{z}$ as an indexing of all strings $y \in \Sigma^{n^2}$, that is, each bit of $\mathbf{z}$ corresponds to a 1-string of the form form $\langle 1, 1^{\lceil \log(n) \rceil}, x, y \rangle$. We define $C_x(\mathbf{z})$ to be the output of the emulation of $E^A(0^{j\leq n}, x)$ with the condition that whenever E makes a query w, we compute $p(w)$ and use its majority output. We will show that for each x there is some $\mathbf{z}$ that can be in one of two cases:

1. $\mathbf{z} = \mathbf{d}_i = 0^i 10^{2^{n^2}-i-1}$ $(0 \leq i < 2^{n^2})$ and $C_x(\mathbf{d}_i) = 1$, defining $v_w = |1\rangle$ for the string w indexed by the $(i+1)$-th bit, and $v_w = |0\rangle$ for all w indexed by the remaining bits;
2. $\mathbf{z} = \mathbf{0} = 0^{2^{n^2}}$ and $C_x(\mathbf{0}) = 0$, defining $v_w = |0\rangle$ for all indexed w.

Due to the recursion used to get $p(w)$ for the 0-strings, the number of levels needed to compute $C_x(\mathbf{z})$ is at most $\lfloor \log n \rfloor - 1$. Therefore, the number of

queries that $C_x(\mathbf{z})$ makes is at most $\prod_{i=0}^{\lfloor \log n \rfloor}(2^n)^{1/2^i}$, which is less than 2^{2n-1}. Therefore, the probability that $C_x(\mathbf{z})$ obtains the correct answer is greater than $(1-2^{-2n})^{2^{2n-1}} > \sqrt{1/e} > 2/3$.

We need, finally, to consider that a subcomputation $p(w)$ may return a wrong result with probability greater than 2^{-2n}. For such cases we can consider that the subcomputation $p(w)$ returns $|1\rangle$ or $|0\rangle$ in an arbitrary way. According to Theorem 4 we have a polynomially small set S of oracle answers that can change with probability greater than ε the output of a quantum algorithm naCQP. We conclude, then, that there is always a way to set the values of v_w for these "bad" w so that neither $C_x(\mathbf{d}_i) = 0$, nor $C_x(\mathbf{0}) = 1$ occur.

References

1. Aaronson, S.: Quantum computing and hidden variables. Physical Review A **71**(3), 032325 (2005). https://doi.org/10.1103/PhysRevA.71.032325
2. Aaronson, S.: $\mathsf{PDQP/qpoly} = \mathsf{ALL}$. Quantum Information & Computation **18**(11-12), 901–909 (2018), https://doi.org/10.5555/3370220.3370221
3. Aaronson, S., Bouland, A., Fitzsimons, J., Lee, M.: The space "just above" BQP. Preprint (2014). https://doi.org/10.48550/arXiv.1412.6507
4. Aaronson, S., Bouland, A., Fitzsimons, J., Lee, M.: The space "just above" BQP. In: Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science. pp. 271–280 (2016). https://doi.org/10.1145/2840728.2840739
5. Baker, T., Gill, J., Solovay, R.: Relativizations of the $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ question. SIAM Journal on Computing **4**(4), 431–442 (1975). https://doi.org/10.1137/0204037
6. Beigel, R., Buhrman, H., Fortnow, L.: NP might not be as easy as detecting unique solutions. In: Proceedings of the thirtieth annual ACM symposium on Theory of computing. pp. 203–208 (1998). https://doi.org/10.1145/276698.276737
7. Bennett, C.H., Bernstein, E., Brassard, G., Vazirani, U.: Strengths and weaknesses of quantum computing. SIAM Journal on Computing **26**(5), 1510–1523 (1997). https://doi.org/10.1137/S0097539796300933
8. Fortnow, L., Rogers, J.: Complexity limitations on quantum computation. J. Comput. Syst. Sci. **59**(2), 240–252 (1999). https://doi.org/10.1006/jcss.1999.1651
9. Gutfreund, D., Shaltiel, R., Ta-Shma, A.: Uniform hardness versus randomness tradeoffs for Arthur-Merlin games. Computational Complexity **12**(3), 85–130 (2003). https://doi.org/10.1007/s00037-003-0178-7
10. Hiromasa, R., Mizutani, A., Takeuchi, Y., Tani, S.: Rewindable quantum computation and its equivalence to cloning and adaptive postselection. Preprint (2022). https://doi.org/10.48550/arXiv.2206.05434
11. Menda, S., Watrous, J.: Oracle separations for quantum statistical zero-knowledge. Preprint (2018). https://doi.org/10.48550/arXiv.1801.08967
12. Raz, R., Tal, A.: Oracle separation of BQP and PH. In: Proceedings of the 51st annual ACM SIGACT Symposium on Theory of Computing. pp. 13–23 (2019). https://doi.org/10.1145/3313276.3316315
13. Tamon, C., Yamakami, T.: Quantum computation relative to oracles. In: Unconventional Models of Computation, UMC'2K, pp. 273–288. Springer (2001). https://doi.org/10.1007/978-1-4471-0313-4_20