QIP = PSPACE

Rahul Jain* Zhengfeng Ji† Sarvagya Upadhyay‡ John Watrous‡

*Department of Computer Science and Centre for Quantum Technologies
National University of Singapore
Republic of Singapore

†Perimeter Institute for Theoretical Physics
Waterloo, Ontario, Canada

‡Institute for Quantum Computing and School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

August 3, 2009

Abstract

We prove that the complexity class QIP, which consists of all problems having quantum interactive proof systems, is contained in PSPACE. This containment is proved by applying a parallelized form of the matrix multiplicative weights update method to a class of semidefinite programs that captures the computational power of quantum interactive proofs. As the containment of PSPACE in QIP follows immediately from the well-known equality IP = PSPACE, the equality QIP = PSPACE follows.

1 Introduction

Efficient proof verification is a fundamental notion in computational complexity theory. The most direct complexity-theoretic abstraction of efficient proof verification is represented by the complexity class NP, wherein a deterministic polynomial-time *verification procedure* decides whether a given polynomial-length *proof string* is valid for a given input. One cannot overstate the importance of this class and its presently unknown relationship to P, the class of problems solvable deterministically in polynomial time. This problem, which is known as the P versus NP problem, is one of the greatest of all unsolved problems in mathematics.

In the early to mid 1980's, Babai [Bab85] and Goldwasser, Micali, and Rackoff [GMR85] introduced a computational model that extends the notion of efficient proof verification to *interactive settings*. (Journal versions of these papers appeared later as [BM88] and [GMR89].) In this model, which is known as the *interactive proof system* model, a computationally bounded *verifier* interacts with a *prover* of unlimited computation power. The interaction comprises one or more rounds of communication between the prover and verifier, and the verifier may make use of randomly generated bits during the interaction. After the rounds of communication are finished, the verifier makes a decision to *accept* or *reject* based on the interaction.

A decision problem A is said to have an interactive proof system if there exists a verifier, always assumed to run in polynomial time, that meets two conditions: the *completeness* condition

and the *soundness* condition. The completeness condition formalizes the requirement that true statements can be proved, which in the present setting means that if an input string x is a yes-instance of A, then there exists a course of action for the prover that causes the verifier to accept with high probability. The soundness condition formalizes the requirement that false statements cannot be proved, meaning in this case that if an input string x is a no-instance of A, then the verifier will reject with high probability no matter what course of action the prover takes. One denotes by IP the collection of decision problems having interactive proof systems. (Here, and throughout the rest of the paper, we take the term *problem* to mean *promise problem*, and consider that all complexity classes to be discussed are classes of promise problems. Promise problems were defined by Even, Selman and Yacobi [ESY84], and readers unfamiliar with them are referred to the survey of Goldreich [Gol05].)

The expressive power of interactive proof systems was not initially known when they were first defined, but it was soon determined to coincide with PSPACE, the class of problems solvable deterministically in polynomial space. The containment IP $\subseteq$ PSPACE, which is generally attributed to Feldman [Fel86], is fairly straightforward—and readers not interested in proving this fact for themselves can find a proof in [HO02]. Known proofs [LFKN92, Sha92, She92] of the reverse containment PSPACE $\subseteq$ IP, on the other hand, are not straightforward, and make essential use of a technique commonly known as *arithmetization*. This technique involves the extension of Boolean formulas to multivariate polynomials over large finite fields whose 0 and 1 elements are taken to represent Boolean values. Through the use of randomness and polynomial interpolation, verifiers may be constructed for arbitrary PSPACE problems.

Many variants of interactive proof systems have been studied, including public-coin interactive proofs [Bab85, BM88, GS89], multi-prover interactive proofs [BOGKW88], zero-knowledge interactive proofs [GMR89, GMW91], and competing-prover interactive proofs [FK97]. The present paper is concerned with *quantum interactive proof systems*, which were first studied a decade after IP = PSPACE was proved [Wat99, KW00]. The fundamental notions of this model are the same as those of classical interactive proof systems, except that the prover and verifier may now process and exchange quantum information. Similar to the classical case, several variants of quantum interactive proof systems have been studied (including those considered in [HKSZ08, KKMV09, KM03, Kob08, MW05, Wat09]).

One of the most interesting aspects of quantum interactive proof systems, which distinguishes them from classical interactive proof systems (at least to the best of our current knowledge), is that they can be *parallelized* to three messages. That is, quantum interactive proof systems consisting of just three messages exchanged between the prover and verifier already have the full power of quantum interactive proofs having a polynomial number of messages [KW00]. Classical interactive proofs are not known to hold this property, and if they do the polynomial-time hierarchy collapses to the second level [BM88].

The complexity class QIP is defined as the class of decision problems having quantum interactive proof systems. QIP trivially contains IP, as the ability of a verifier to process quantum information is never a hindrance—a quantum verifier can simulate a classical verifier, and a computationally unbounded prover can never use quantum information to an advantage against a verifier behaving classically. The inclusion PSPACE $\subseteq$ QIP is therefore immediate. The best upper bound on QIP known prior to the present paper was QIP $\subseteq$ EXP, which was proved in [KW00] through the use of semidefinite programming. The optimal probability with which a given verifier can be made to accept in a quantum interactive proof system can be represented as an exponential-size semidefinite program, and known polynomial-time algorithms for semidefinite programming

provide the required tool to prove the containment. It has been an open problem for the last decade to establish more precise bounds on the class QIP.

It was recently shown in the paper [JUW09] that QIP(2), the class of problem having 2-message quantum interactive proof systems, is contained in PSPACE. That paper made use of a parallel algorithm, based on a method known as the *matrix multiplicative weights update method*, to approximate optimal solutions for a class of semidefinite programs that represent the maximum acceptance probabilities for verifiers in two-message quantum interactive proofs. In this paper we extend this result to all of QIP, establishing the relationship QIP = PSPACE. Similar to [JUW09], we use the matrix multiplicative weights update method, together with parallel methods for matrix computations.

The *multiplicative weights method* is a framework for algorithm design having its origins in various fields, including learning theory, game theory, and optimization. Its matrix variant, as discussed in the survey paper [AHK05] and the PhD thesis of Kale [Kal07], gives an iterative way to approximate the optimal value of semidefinite programs [AK07, WK06]. In addition to its application in [JUW09], it was applied to quantum complexity in [JW09] to prove the containment of the complexity class QRG(1) in PSPACE. The key strength of this method for these applications is that it can be parallelized for some special classes of semidefinite programs.

A key result that allows our technique to work for the entire class QIP is the characterization QIP = QMAM proved in [MW05]. This characterization, which is described in greater detail in the next section, concerns a restricted notion of interactive proof systems known as *Arthur–Merlin games*. An Arthur–Merlin game is an interactive proof system wherein the verifier can only send uniformly generated random bits to the prover. Following Babai [Bab85], one refers to the verifier as *Arthur* and to the prover as *Merlin* in this setting. It is also typical to refer to the individual bits of Arthur's messages as *coins*, given that they are each uniformly generated like the flip of a fair coin. The restriction that Arthur sends only uniformly generated bits to Merlin, and therefore does not have the option to base his messages on private information unknown to Merlin, would seem to limit the power of Arthur–Merlin games in comparison to ordinary interactive proof systems. But in fact this is known not to be the case, both for classical [GS89] and quantum [MW05] interactive proof systems. In the quantum setting, this characterization admits a significant simplification in the semidefinite programs that capture the complexity of the class QIP.

The remainder of this paper has the following organization. Section 2 includes background information, notation, and other preliminary discussions that are relevant to the remainder of the paper. Section 3 describes a semidefinite programming problem that captures the complexity of the class QIP based on quantum Arthur–Merlin games, and Section 4 presents the main algorithm that solves this problem. Finally, Section 5 discusses a parallel approximation to the algorithm from Section 4 and explains how its properties lead to the containment QIP $\subseteq$ PSPACE.

2 Preliminaries

This section contains a summary of the notation and terminology on linear algebra, quantum information, semidefinite programming, quantum Arthur–Merlin games, and bounded-depth circuits that is used later in the paper. For the most part, these discussions are intended only to make clear the notation and terminology that we use, and not to provide introductions to these topics. We assume that the reader already has familiarity with complexity theory and quantum computing, and refer readers who are not to [AB09] and [NC00].

2.1 Linear algebra and quantum information

A *quantum register* refers to a collection of qubits, or more generally a finite-size component in a quantum computer. Every quantum register V has associated with it a finite, non-empty set Σ of classical states and a complex vector space of the form $\mathcal{V} = \mathbb{C}^{\Sigma}$. We use the Dirac notation $\{|a\rangle : a \in \Sigma\}$ to refer to the *standard basis* (or elementary unit vectors) in $\mathcal{V}$, and define the inner product and Euclidean norm on $\mathcal{V}$ in the standard way. The set $\{\langle a| : a \in \Sigma\}$ consists of the elements in the dual space of $\mathcal{V}$ that are in correspondence with the standard basis vectors.

For such a space $\mathcal{V}$, we write $\mathrm{L}(\mathcal{V})$ to denote the space of linear mappings, or *operators*, from $\mathcal{V}$ to itself, which is identified with the set of square complex matrices indexed by Σ in usual way. An inner product on $\mathrm{L}(\mathcal{V})$ is defined as

$$\langle A, B \rangle = \mathrm{Tr}(A^* B),$$

where A^* denotes the adjoint (or conjugate transpose) of A. The identity operator on $\mathcal{V}$ is denoted $\mathbb{1}_{\mathcal{V}}$ (or just $\mathbb{1}$ when $\mathcal{V}$ is understood).

The following special types of operators are relevant to the paper:

1. An operator $A \in \mathrm{L}(\mathcal{V})$ is *Hermitian* if $A = A^*$. The eigenvalues of a Hermitian operator are always real, and for $m = \dim(\mathcal{V})$ we write

   $$\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_m(A)$$

   to denote the eigenvalues of A sorted from largest to smallest.

2. An operator $P \in \mathrm{L}(\mathcal{V})$ is *positive semidefinite* if it is Hermitian and all of its eigenvalues are nonnegative. The set of such operators is denoted $\mathrm{Pos}(\mathcal{V})$. The notation $P \geq 0$ also indicates that P is positive semidefinite, and more generally the notations $A \leq B$ and $B \geq A$ indicate that $B - A \geq 0$ for Hermitian operators A and B.

   Every Hermitian operator A can be expressed uniquely as $A = P - Q$ for positive semidefinite operators P and Q satisfying $\langle P, Q \rangle = 0$. The operator P is said to be the *positive part* of A, while Q is the *negative part*.

3. A positive semidefinite operator $P \in \mathrm{Pos}(\mathcal{V})$ is also said to be *positive definite* if all of its eigenvalues are positive (which implies that P must be invertible). The notation $P > 0$ also indicates that P is positive definite, and the notations $A < B$ and $B > A$ indicate that $B - A > 0$ for Hermitian operators A and B.

4. An operator $\rho \in \mathrm{Pos}(\mathcal{V})$ is a *density operator* if it is both positive semidefinite and has trace equal to 1. The set of such operators is denoted $\mathrm{D}(\mathcal{V})$.

5. An operator $\Pi \in \mathrm{Pos}(\mathcal{V})$ is a *projection* if all of its eigenvalues are either 0 or 1.

A *quantum state* of a register V is a density operator $\rho \in \mathrm{D}(\mathcal{V})$, and a *measurement* on V is a collection $\{P_b : b \in \Gamma\} \subseteq \mathrm{Pos}(\mathcal{V})$ satisfying

$$\sum_{b \in \Gamma} P_b = \mathbb{1}_{\mathcal{V}}.$$

The set Γ is the set of *measurement outcomes*, and when such a measurement is performed on V while it is in the state ρ, each outcome $b \in \Gamma$ occurs with probability $\langle P_b, \rho \rangle$.

The *spectral norm* of an operator $A \in \mathrm{L}(\mathcal{V})$ is defined as

$$\|A\| = \max\{\|Av\| : v \in \mathcal{V}, \|v\| = 1\}.$$

The spectral norm is sub-multiplicative, meaning that $\|AB\| \leq \|A\| \|B\|$ for all operators $A, B \in \mathrm{L}(\mathcal{V})$, and it holds that $\|P\| = \lambda_1(P)$ for every positive semidefinite operator P. For any operator $A \in \mathrm{L}(\mathcal{V})$, the exponential of A is defined as

$$\exp(A) = \mathbb{1} + A + A^2/2 + A^3/6 + \cdots$$

The *Golden-Thompson Inequality* (see Section IX.3 of [Bha97]) states that, for any two Hermitian operators A and B on $\mathcal{V}$, we have

$$\mathrm{Tr}\left[\exp(A+B)\right] \leq \mathrm{Tr}\left[\exp(A)\exp(B)\right].$$

The tensor product $\mathcal{V} \otimes \mathcal{W}$ of vector spaces $\mathcal{V} = \mathbb{C}^{\Sigma}$ and $\mathcal{W} = \mathbb{C}^{\Gamma}$ may be associated with the space $\mathbb{C}^{\Sigma \times \Gamma}$, and the tensor product of operators $A \in \mathrm{L}(\mathcal{V})$ and $B \in \mathrm{L}(\mathcal{W})$ is then taken to be the unique operator $A \otimes B \in \mathrm{L}(\mathcal{V} \otimes \mathcal{W})$ satisfying $(A \otimes B)(v \otimes w) = (Av) \otimes (Bw)$ for all $v \in \mathcal{V}$ and $w \in \mathcal{W}$. These notions may be associated with the usual Kronecker product of vectors and matrices. For quantum registers V and W, the space $\mathcal{V} \otimes \mathcal{W}$ is associated with the pair (V, W), viewed as a single register. Tensor products involving three or more spaces are handled similarly.

For a given linear mapping of the form $\Phi : \mathrm{L}(\mathcal{V}) \to \mathrm{L}(\mathcal{W})$, one defines the adjoint mapping $\Phi^* : \mathrm{L}(\mathcal{W}) \to \mathrm{L}(\mathcal{V})$ to be the unique linear mapping that satisfies

$$\langle B, \Phi(A) \rangle = \langle \Phi^*(B), A \rangle$$

for all operators $A \in \mathrm{L}(V)$ and $B \in \mathrm{L}(\mathcal{W})$.

Finally, for spaces $\mathcal{V}$ and $\mathcal{W}$, one defines the *partial trace* $\mathrm{Tr}_{\mathcal{V}} : \mathrm{L}(\mathcal{V} \otimes \mathcal{W}) \to \mathrm{L}(\mathcal{W})$ to be the unique linear mapping that satisfies $\mathrm{Tr}_{\mathcal{V}}(A \otimes B) = (\mathrm{Tr}\, A)B$ for all $A \in \mathrm{L}(\mathcal{V})$ and $B \in \mathrm{L}(\mathcal{W})$. A similar notation is used for the partial trace $\mathrm{Tr}_{\mathcal{W}}$, or partial traces defined on three or more tensor factors. When this notation is used, the spaces on which the trace is not taken are determined by context. When a pair of registers (V, W) is viewed as a single register and has the quantum state $\rho \in \mathrm{D}(\mathcal{V} \otimes \mathcal{W})$, one defines the state of W to be $\mathrm{Tr}_{\mathcal{V}}(\rho)$. In other words, the partial trace describes the action of destroying, or simply ignoring, a given quantum register.

2.2 Semidefinite programming

A *semidefinite program* over complex vector spaces $\mathcal{V}$ and $\mathcal{W}$ is a pair of optimization problems as follows.

Primal problem		Dual problem	
maximize:	$\langle C, X \rangle$	minimize:	$\langle D, Y \rangle$
subject to:	$\Psi(X) \leq D$,	subject to:	$\Psi^*(Y) \geq C$,
	$X \in \mathrm{Pos}(\mathcal{V})$.		$Y \in \mathrm{Pos}(\mathcal{W})$.

Here, the operators $C \in \mathrm{L}(\mathcal{V})$ and $D \in \mathrm{L}(\mathcal{W})$ are Hermitian and $\Psi : \mathrm{L}(\mathcal{V}) \to \mathrm{L}(\mathcal{W})$ must be a linear mapping that maps Hermitian operators to Hermitian operators. Readers familiar with semidefinite programming will note that the above form of a semidefinite program is different

from the well-known *standard form*, but it is equivalent and better suited for this paper's needs. The form given above is, in essence, the one that is typically followed for general conic programming [BV04].

It is typical that semidefinite programs are stated in forms that do not explicitly describe Ψ, C and D, and the same is true for the semidefinite programs we will consider. It is, however, routine to put them into the above form.

With the above optimization problems in mind, one defines the *primal feasible* set $\mathcal{P}$ and the *dual feasible* set $\mathcal{D}$ as

$$\mathcal{P} = \{X \in \mathrm{Pos}(\mathcal{V}) \,:\, \Psi(X) \leq D\},$$
$$\mathcal{D} = \{Y \in \mathrm{Pos}(\mathcal{W}) \,:\, \Psi^*(Y) \geq C\}.$$

Operators $X \in \mathcal{P}$ and $Y \in \mathcal{D}$ are also said to be *primal feasible* and *dual feasible*, respectively. The functions $X \mapsto \langle C, X\rangle$ and $Y \mapsto \langle D, Y\rangle$ are called the primal and dual *objective functions*, and the *optimal values* associated with the primal and dual problems are defined as

$$\alpha = \sup_{X \in \mathcal{P}} \langle C, X\rangle \qquad \text{and} \qquad \beta = \inf_{Y \in \mathcal{D}} \langle D, Y\rangle.$$

Semidefinite programs have associated with them a powerful theory of *duality*, which refers to the special relationship between the primal and dual problems. The property of *weak duality*, which holds for all semidefinite programs, states that $\alpha \leq \beta$. This property implies that every dual feasible operator $Y \in \mathcal{D}$ provides an upper bound of $\langle D, Y\rangle$ on the value $\langle C, X\rangle$ that is achievable over all choices of a primal feasible $X \in \mathcal{P}$, and likewise every primal feasible operator $X \in \mathcal{P}$ provides a lower bound of $\langle C, X\rangle$ on the value $\langle D, Y\rangle$ that is achievable over all choices of a dual feasible $Y \in \mathcal{D}$.

It is not always the case that $\alpha = \beta$ for a given semidefinite program, but in most natural cases it does hold. The situation in which $\alpha = \beta$ is known as *strong duality*, and several conditions have been identified that imply strong duality. One such condition is *strict dual feasibility*: if α is finite and there exists an operator $Y > 0$ such that $\Psi^*(Y) > C$, then $\alpha = \beta$. The symmetric condition of *strict primal feasibility* also implies strong duality.

2.3 Single-coin quantum Arthur–Merlin games

Quantum Arthur–Merlin games were proposed in [MW05] as a natural quantum variant of classical Arthur–Merlin games. Here, one simply mimics the classical definition in requiring that Arthur's messages to Merlin consist of uniformly generated random bits. Merlin's messages to Arthur, however, may be quantum; and after all of the messages have been exchanged Arthur is free to perform a quantum computation when deciding to accept or reject.

Of particular interest to us are quantum Arthur–Merlin games in which three messages are exchanged, and where Arthur's only message consists of a single bit. In more precise terms, such an interaction takes the following form:

1. Merlin sends a quantum register W to Arthur. Merlin is free to initialize this register to any quantum state of his choice, and may entangle it with a register of his own if he chooses.
2. After receiving W from Merlin, Arthur chooses a bit $a \in \{0, 1\}$ uniformly at random. Merlin learns the value of a.

3. Merlin sends Arthur a second quantum register Y. He does this after step 2, so he has the option to condition the state of Y upon the value of a. The register Y could, of course, be entangled with W in any way that quantum information theory permits.
4. After receiving Y, Arthur performs one of two binary-valued measurements, determined by the value of the random bit a, on the pair (W, Y). The measurement outcome 1 is interpreted as *acceptance*, while 0 is interpreted as *rejection*.

Arthur's measurements must of course be efficiently implementable. This notion is formalized by requiring that the measurements are implementable by polynomial-time generated families of quantum circuits, which naturally requires the registers W and Y to consist of a number of qubits that is polynomial in the length of the input. Further details may be found in [MW05].

The result of [MW05] that we make use of is that every problem $A \in \mathrm{QIP}$ has a single-coin Arthur–Merlin game as just described. The game is such that if x is a yes-instance of the problem A, then Arthur accepts with probability 1, whereas if the input x is a no-instance of the problem then Arthur accepts with probability at most $1/2 + \varepsilon$, for any desired constant $\varepsilon > 0$. (In the construction given in [MW05], Arthur's measurements are always nontrivial projective measurements. This implies that even for no-instance inputs, Merlin can cause Arthur to accept with probability at least $1/2$ by simply guessing in advance Arthur's random bit.)

2.4 Bounded-depth circuit complexity

In the last section of the paper, we will require the definitions of two complexity classes based on bounded-depth circuit families: NC and $\mathrm{NC}(\mathit{poly})$. It is convenient for us to define these as classes of *functions* rather than decision problems, and when we wish to view them as classes of decision problems we simply restrict our attention to binary-valued functions. The class NC contains all functions computable by logarithmic-space uniform Boolean circuits of polylogarthmic depth, and $\mathrm{NC}(\mathit{poly})$ contains all functions that can be computed by polynomial-space uniform families of Boolean circuits having polynomial-depth. For decision problems it is known [Bor77] that $\mathrm{NC}(\mathit{poly}) = \mathrm{PSPACE}$, and the proof of our main result will make use of this fact.

There are two fundamental properties of $\mathrm{NC}(\mathit{poly})$ that we will take advantage of. The first is that functions in NC and $\mathrm{NC}(\mathit{poly})$ compose well, and the second is that many computational problems involving matrices are in NC. In more precise terms, the first property is as follows. If $F : \{0,1\}^* \to \{0,1\}^*$ is a function in $\mathrm{NC}(\mathit{poly})$ and $G : \{0,1\}^* \to \{0,1\}^*$ is a function in NC, then the composition $G \circ F$ is also in $\mathrm{NC}(\mathit{poly})$. This follows from the most straightforward way of composing the families of circuits that compute F and G.

To discuss the second property, it will be helpful to make clear our assumptions concerning matrix computations. We will always assume that the matrices on which computations are performed have entries with rational real and imaginary parts, and that the rational numbers are represented as pairs of integers in binary notation. Unless it is explicitly noted otherwise, any other rational numbers involved in our computations will be represented in a similar way.

With these assumptions in place, we first note that elementary matrix operations, including inverses and iterated sums and products of matrices, are known to be in NC. There is an extensive literature on this topic, and we refer the reader to von zur Gathen's survey [Gat93] for more details. We also note that matrix exponentials and spectral decompositions can be *approximated* to high accuracy in NC. In more precise terms, the following two problems are in NC.

Matrix exponentials

Input: An $n \times n$ matrix M, a positive rational number η, and an integer k expressed in unary notation (i.e., 1^k).

Promise: $\|M\| \leq k$.

Output: An $n \times n$ matrix X such that $\|\exp(M) - X\| < \eta$.

Spectral decompositions

Input: An $n \times n$ Hermitian matrix H and a positive rational number η.

Output: An $n \times n$ unitary matrix U and an $n \times n$ real diagonal matrix Λ such that

$$\|M - U\Lambda U^*\| < \eta.$$

The reader will note that in these problems, the description of the error parameter η could require as few as $O(\log(1/\eta))$ bits. This implies that highly accurate approximations, for instance where $\eta = 2^{-n}$, are possible in NC. The fact that matrix exponentials can be approximated in NC follows by truncating the series

$$\exp(M) = \mathbb{1} + M + M^2/2 + M^3/6 + \cdots$$

to a number of terms linear in $k + \log(1/\eta)$. (From a numerical point of view this is not a very good way to compute matrix exponentials [ML03], but it is arguably the simplest way to prove that the stated problem is in NC.) The fact that spectral decompositions can be approximated in NC follows from a composition of known facts: in NC one can compute characteristic polynomials and null spaces of matrices, perform orthogonalizations of vectors, and approximate roots of integer polynomials to high precision [Csa76, BGH82, BCP83, BOFKT86, Gat93, Nef94].

3 A semidefinite programming formulation of the problem

Consider Arthur's verification procedure for a given single-coin QMAM protocol on a fixed input string x. Arthur first receives a register W, then generates a random bit $a \in \{0,1\}$, and then receives a second register Y. He then measures (W,Y) with respect to a binary-valued measurement

$$\{P_a, \mathbb{1} - P_a\} \subset \mathrm{Pos}\left(\mathcal{W} \otimes \mathcal{Y}\right),$$

where we take each of the operators P_0 and P_1 to represent acceptance and $\mathbb{1} - P_0$ and $\mathbb{1} - P_1$ to represent rejection. If the quantum state of (W,Y) is given by a density operator $\rho \in \mathrm{D}\left(\mathcal{W} \otimes \mathcal{Y}\right)$ when Arthur measures, he will therefore accept with probability $\langle P_a, \rho \rangle$.

Now define

$$Q = \frac{1}{2} |0\rangle\langle 0| \otimes P_0 + \frac{1}{2} |1\rangle\langle 1| \otimes P_1 \in \mathrm{Pos}\left(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y}\right),$$

where we take $\mathcal{X} = \mathbb{C}^{\{0,1\}}$ to be the vector space corresponding to Arthur's random choice of $a \in \{0,1\}$, and consider the optimal probability that Merlin can cause Arthur to accept. If, for each of the values $a \in \{0,1\}$, Merlin is able to leave the state ρ_a in the registers (W,Y) right before Arthur measures, he will convince Arthur to accept with probability

$$\frac{1}{2} \langle P_0, \rho_0 \rangle + \frac{1}{2} \langle P_1, \rho_1 \rangle = \langle Q, X \rangle \tag{1}$$

for

$$X = |0\rangle\langle 0| \otimes \rho_0 + |1\rangle\langle 1| \otimes \rho_1.$$

There is, of course, a constraint on Merlin's choice of ρ_0 and ρ_1, which is that they must agree on W, as Merlin cannot touch the register W at any point after Arthur chooses the random bit a. In more precise terms, it must hold that

$$\mathrm{Tr}_{\mathcal{Y}}(\rho_0) = \sigma = \mathrm{Tr}_{\mathcal{Y}}(\rho_1) \tag{2}$$

for some density operator $\sigma \in \mathrm{D}(\mathcal{W})$. This, in fact, is Merlin's only constraint—for if he holds a purification of the state σ, he is free to set the state of (W, Y) to any choice of ρ_0 and ρ_1 satisfying (2) without needing access to W.

Now, we note that the condition (2) implies that

$$\mathrm{Tr}_{\mathcal{Y}}(X) = \mathbb{1}_{\mathcal{X}} \otimes \sigma. \tag{3}$$

Moreover, for an arbitrary operator $X \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y})$ satisfying the constraint (3), one has that the operators ρ_0 and ρ_1 defined as

$$\rho_a = (\langle a| \otimes \mathbb{1}_{\mathcal{W}\otimes\mathcal{Y}})\, X\, (|a\rangle \otimes \mathbb{1}_{\mathcal{W}\otimes\mathcal{Y}})$$

for $a \in \{0, 1\}$ satisfy the conditions (1) and (2). It follows that the following semidefinite program represents the optimal probability with which Merlin can convince Arthur to accept.

Primal problem		Dual problem	
maximize:	$\langle Q, X\rangle$	minimize:	$\lVert \mathrm{Tr}_{\mathcal{X}}(Y) \rVert$
subject to:	$\mathrm{Tr}_{\mathcal{Y}}(X) \leq \mathbb{1}_{\mathcal{X}} \otimes \sigma$,	subject to:	$Y \otimes \mathbb{1}_{\mathcal{Y}} \geq Q$,
	$X \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y})$,		$Y \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{W})$.
	$\sigma \in \mathrm{D}(\mathcal{W})$.		

Note that the inequality in the primal problem can be exchanged for an equality without changing the optimal value. This is because any primal feasible X can be inflated to achieve the equality $\mathrm{Tr}_{\mathcal{Y}}(X) = \mathbb{1}_{\mathcal{X}} \otimes \sigma$ for some choice of σ, and this can only increase the value of the objective function by virtue of the fact that Q is positive semidefinite. It is immediate that the optimal solution to the primal problem is bounded and the dual problem is strictly feasible, from which strong duality follows; the primal and dual problems have the same optimal values.

Now, under the assumption that Q is invertible, one may perform a change of variables to put the above semidefinite program into a form that more closely resembles the one in [JUW09]. To do this we define a linear mapping $\Phi : \mathrm{L}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y}) \to \mathrm{L}(\mathcal{X} \otimes \mathcal{W})$ as

$$\Phi(X) = \mathrm{Tr}_{\mathcal{Y}}\left(Q^{-1/2} X Q^{-1/2}\right), \tag{4}$$

whose adjoint mapping $\Phi^* : \mathrm{L}(\mathcal{X} \otimes \mathcal{W}) \to \mathrm{L}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y})$ is given by

$$\Phi^*(Y) = Q^{-1/2}(Y \otimes \mathbb{1}_{\mathcal{Y}})Q^{-1/2},$$

and consider the following semidefinite program.

Primal problem		Dual problem	
maximize:	$\mathrm{Tr}(X)$	minimize:	$\|\mathrm{Tr}_{\mathcal{X}}(Y)\|$
subject to:	$\Phi(X) \leq \mathbb{1}_{\mathcal{X}} \otimes \sigma$,	subject to:	$\Phi^*(Y) \geq \mathbb{1}_{\mathcal{X}\otimes\mathcal{W}\otimes\mathcal{Y}}$,
	$X \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y})$,		$Y \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{W})$.
	$\sigma \in \mathrm{D}(\mathcal{W})$.		

It is clear that this semidefinite program has the same optimal value as the previous one.

We will be interested in the optimal value of this semidefinite program in the case that $\|Q^{-1}\|$ is upper-bounded by a fixed constant and where there is a promise on the optimal value. The promise, which will come from the properties of the quantum Arthur–Merlin games under consideration, is that the optimal value does not lie in the interval $(5/8, 7/8)$, and the goal is to determine whether the optimal value is larger than $7/8$ or smaller than $5/8$.

For readers familiar with the semidefinite program for QIP(2) presented in [JUW09], we note that there are two essential differences between it and the one above. The first difference is that the semidefinite program in [JUW09] effectively replaces the density operator σ with the scalar value 1, which would seem to suggest added difficulty for the case at hand. The second difference is that $\mathcal{X}$ is two-dimensional for the semidefinite program above, whereas it has arbitrary size in [JUW09]. This second difference more than compensates for the difficulty induced by the first, and we find that the above semidefinite program is actually much easier to solve than the one for QIP(2).

4 The main algorithm and its analysis

We now present the main algorithm for the semidefinite programming problem from the previous section. The algorithm, which is described in Figure 1, takes as input an operator

$$Q \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y}).$$

It is assumed that Q is invertible and satisfies $\|Q^{-1}\| \leq 64$. (The algorithm could easily be adapted to handle any other fixed constant in place of 64, but this choice is sufficient for our needs.) Moreover, it is assumed that the optimal value of the semidefinite program in Section 3 that is defined by Q does not lie in the interval $(5/8, 7/8)$. Our goal is to prove that the algorithm accepts when the optimal value is at least $7/8$ and rejects when the optimal value is at most $5/8$.

Here we present the correctness of the algorithm under the assumption that all computations are performed exactly. Issues that arise due to inaccuracies in the computation are discussed in the next section.

Assume first that the algorithm accepts, and write

$$\rho = \rho_t, \quad \Pi = \Pi_t, \quad \xi = \xi_t \quad \text{and} \quad \beta = \beta_t$$

for $t \in \{0, \ldots, T-1\}$ corresponding to the iteration in which acceptance occurs. For the sake of clarity, let us note explicitly that

$$\rho \in \mathrm{D}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y}), \qquad \Pi \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{W}) \qquad \text{and} \qquad \xi \in \mathrm{D}(\mathcal{W}).$$

We wish to prove that the optimal value of our semidefinite program is at least $7/8$, and we will do this by constructing a primal feasible solution that achieves an objective value strictly larger than $5/8$.

1. Let $N = \dim(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y})$ and $M = \dim(\mathcal{W})$, and define

$$W_0 = \mathbb{1}_{\mathcal{X}\otimes\mathcal{W}\otimes\mathcal{Y}}, \qquad \rho_0 = W_0/N, \qquad Z_0 = \mathbb{1}_{\mathcal{W}} \qquad \text{and} \qquad \xi_0 = Z_0/M.$$

   Also let

$$\gamma = \frac{4}{3}, \qquad \varepsilon = \frac{1}{64}, \qquad \delta = \frac{\varepsilon}{2\,\|Q^{-1}\|} \qquad \text{and} \qquad T = \left\lceil \frac{4\log(N)}{\varepsilon^3\delta} \right\rceil.$$

2. Repeat for each $t = 0, \ldots, T-1$:

   (a) Let Π_t be the projection onto the positive eigenspaces of the operator

$$\Phi(\rho_t) - \gamma\,\mathbb{1}_{\mathcal{X}} \otimes \xi_t,$$

   where Φ is defined from Q as in (4), and set $\beta_t = \langle \Pi_t, \Phi(\rho_t)\rangle$.

   (b) If $\beta_t \leq \varepsilon$ then *accept*, else let

$$W_{t+1} = \exp\left(-\epsilon\delta \sum_{j=0}^{t} \Phi^*(\Pi_j/\beta_j)\right), \qquad \rho_{t+1} = W_{t+1}/\operatorname{Tr}(W_{t+1}),$$

   and

$$Z_{t+1} = \exp\left(\varepsilon\delta \sum_{j=0}^{t} \operatorname{Tr}_{\mathcal{X}}(\Pi_j/\beta_j)\right), \qquad \xi_{t+1} = Z_{t+1}/\operatorname{Tr}(Z_{t+1}).$$

3. If acceptance did not occur in step 2, then *reject*.

Figure 1: An algorithm that *accepts* if the optimal value of the semidefinite program in Section 3 is larger than 7/8, and *rejects* if the optimal value is smaller than 5/8.

By the definition of Π, it holds that

$$\Pi\Phi(\rho)\Pi \geq \Pi(\Phi(\rho) - \gamma\,\mathbb{1}_{\mathcal{X}} \otimes \xi)\Pi \geq \Phi(\rho) - \gamma\,\mathbb{1}_{\mathcal{X}} \otimes \xi, \tag{5}$$

and by Lemma 1 (which is stated and proved below) it holds that

$$2\mathbb{1}_{\mathcal{X}} \otimes \operatorname{Tr}_{\mathcal{X}}(\Pi\Phi(\rho)\Pi) \geq \Pi\Phi(\rho)\Pi. \tag{6}$$

Combining the equations (5) and (6) one has

$$\Phi(\rho) \leq \mathbb{1}_{\mathcal{X}} \otimes (\gamma\,\xi + 2\operatorname{Tr}_{\mathcal{X}}(\Pi\Phi(\rho)\Pi)). \tag{7}$$

It therefore holds that

$$X = \frac{\rho}{\gamma + 2\langle \Pi, \Phi(\rho)\rangle} \qquad \text{and} \qquad \sigma = \frac{\gamma\xi + 2\operatorname{Tr}_{\mathcal{X}}(\Pi\Phi(\rho)\Pi)}{\gamma + 2\langle \Pi, \Phi(\rho)\rangle}$$

represent a feasible solution to the primal problem under consideration, achieving the objective value

$$\frac{1}{\gamma + 2\langle \Pi, \Phi(\rho)\rangle} = \frac{1}{\gamma + 2\beta} \geq \frac{1}{\gamma + 2\varepsilon} > \frac{5}{8}$$

as required.

Now assume that the algorithm rejects, and consider the operator

$$Y = \frac{(1+2\varepsilon)}{T} \sum_{t=0}^{T-1} \Pi_t / \beta_t.$$

We claim that Y is dual feasible and achieves an objective value that is strictly smaller than $7/8$. This will imply that the optimal value of the semidefinite program is at most $5/8$.

Let us first prove that Y is dual feasible. It is clear that Y is positive semidefinite, so it suffices to prove that $\Phi^*(Y) \geq \mathbb{1}_{\mathcal{X}\otimes\mathcal{W}\otimes\mathcal{Y}}$, or equivalently that $\lambda_N(\Phi^*(Y)) \geq 1$. Observe, for each $t = 0, \ldots, T-1$, that

$$\begin{aligned}
\operatorname{Tr}(W_{t+1}) &= \operatorname{Tr}\left[\exp\left(-\varepsilon\delta\Phi^*(\Pi_0/\beta_0 + \cdots + \Pi_t/\beta_t)\right)\right] \\
&\leq \operatorname{Tr}\left[\exp\left(-\varepsilon\delta\Phi^*(\Pi_0/\beta_0 + \cdots + \Pi_{t-1}/\beta_{t-1})\right)\exp\left(-\varepsilon\delta\Phi^*(\Pi_t/\beta_t)\right)\right] \\
&= \operatorname{Tr}\left[W_t \exp\left(-\varepsilon\delta\Phi^*(\Pi_t/\beta_t)\right)\right]
\end{aligned}$$

by the Golden–Thompson inequality. As each Π_t is a projection operator, we have

$$\|\Phi^*(\Pi_t)\| = \left\|Q^{-1/2}(\Pi_t \otimes \mathbb{1}_{\mathcal{Y}})Q^{-1/2}\right\| \leq \left\|Q^{-1/2}\right\|^2 = \left\|Q^{-1}\right\|,$$

where we have used the sub-multiplicativity of the spectral norm to obtain the inequality. Given that $\beta_t > \varepsilon$ in the case at hand, it follows that $\|\delta\Phi^*(\Pi_t/\beta_t)\| < 1$. By Lemma 2 (also presented below) it therefore follows that

$$\exp\left(-\varepsilon\delta\Phi^*(\Pi_t/\beta_t)\right) \leq \mathbb{1} - \varepsilon\delta\exp(-\varepsilon)\Phi^*(\Pi_t/\beta_t).$$

As each W_t is positive semidefinite, we obtain

$$\operatorname{Tr}(W_{t+1}) \leq \operatorname{Tr}(W_t)\left(1 - \varepsilon\delta\exp(-\varepsilon)\left\langle \frac{W_t}{\operatorname{Tr}(W_t)}, \Phi^*(\Pi_t/\beta_t)\right\rangle\right). \tag{8}$$

Substituting $\rho_t = W_t/\operatorname{Tr}(W_t)$ yields

$$\begin{aligned}
\operatorname{Tr}(W_{t+1}) &\leq \operatorname{Tr}(W_t)\left(1 - \varepsilon\delta\exp(-\varepsilon)\left\langle\rho_t, \Phi^*(\Pi_t/\beta_t)\right\rangle\right) \\
&= \operatorname{Tr}(W_t)\left(1 - \varepsilon\delta\exp(-\varepsilon)\right) \\
&\leq \operatorname{Tr}(W_t)\exp(-\varepsilon\delta\exp(-\varepsilon)),
\end{aligned}$$

where the equality follows from $\langle\rho_t, \Phi^*(\Pi_t)\rangle = \langle\Phi(\rho_t), \Pi_t\rangle = \beta_t$ and the last inequality follows from the fact that $1 + z \leq \exp(z)$ for all real numbers z. As $\operatorname{Tr}(W_0) = N$, it follows that

$$\operatorname{Tr}(W_T) \leq \operatorname{Tr}(W_0)\exp(-T\varepsilon\delta\exp(-\varepsilon)) = \exp(-T\varepsilon\delta\exp(-\varepsilon) + \log(N)). \tag{9}$$

On the other hand, we have

$$\operatorname{Tr}(W_T) = \operatorname{Tr}\left[\exp\left(-\varepsilon\delta\sum_{t=0}^{T-1}\Phi^*\left(\Pi_t/\beta_t\right)\right)\right] \geq \exp\left(-\varepsilon\delta\lambda_N\left(\Phi^*\left(\sum_{t=0}^{T-1}\Pi_t/\beta_t\right)\right)\right). \tag{10}$$

Combining (9) and (10), we have

$$\lambda_N\left(\Phi^*\left(\sum_{t=0}^{T-1}\Pi_t/\beta_t\right)\right) \geq T\exp(-\varepsilon) - \frac{\log(N)}{\varepsilon\delta}.$$

Using the inequality $\exp(-\varepsilon) - \varepsilon^2/4 > 1 - \varepsilon$, and substituting the value of T specified by the algorithm, we have

$$\lambda_N(\Phi^*(Y)) \geq (1+2\varepsilon)\left(\exp(-\varepsilon) - \frac{\log(N)}{T\varepsilon\delta}\right) > (1+2\varepsilon)(1-\varepsilon) > 1$$

as required.

Now it remains to establish an upper bound on the dual objective value achieved by Y. A similar method to the one used to prove the feasibility of Y above will provide a suitable bound. We begin by observing, for each $t = 0, \ldots, T-1$, that

$$\begin{aligned}
\operatorname{Tr}(Z_{t+1}) &= \operatorname{Tr}\left[\exp\left(\varepsilon\delta \operatorname{Tr}_{\mathcal{X}}\left(\Pi_0/\beta_0 + \cdots + \Pi_t/\beta_t\right)\right)\right] \\
&\leq \operatorname{Tr}\left[\exp\left(\varepsilon\delta \operatorname{Tr}_{\mathcal{X}}\left(\Pi_0/\beta_0 + \cdots + \Pi_{t-1}/\beta_{t-1}\right)\right)\exp\left(\varepsilon\delta \operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t)\right)\right] \\
&= \operatorname{Tr}\left[Z_t \exp\left(\varepsilon\delta \operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t)\right)\right].
\end{aligned}$$

Given that

$$\left\|\operatorname{Tr}_{\mathcal{X}}(\Pi_t)\right\| \leq \left\|\left(\langle 0| \otimes \mathbb{1}_{\mathcal{W}}\right)\Pi_t\left(|0\rangle \otimes \mathbb{1}_{\mathcal{W}}\right)\right\| + \left\|\left(\langle 1| \otimes \mathbb{1}_{\mathcal{W}}\right)\Pi_t\left(|1\rangle \otimes \mathbb{1}_{\mathcal{W}}\right)\right\| \leq 2,$$

and using the fact that $\beta_t > \epsilon$ in the case at hand, it follows that $\left\|\delta \operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t)\right\| < 1$. We now apply Lemma 2 to obtain

$$\exp\left(\varepsilon\delta \operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t)\right) \leq \mathbb{1} + \varepsilon\delta\exp(\varepsilon)\operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t).$$

As each Z_t is positive semidefinite it follows that

$$\operatorname{Tr}(Z_{t+1}) \leq \operatorname{Tr}(Z_t)\left(1 + \varepsilon\delta\exp(\varepsilon)\left\langle \frac{Z_t}{\operatorname{Tr}(Z_t)}, \operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t)\right\rangle\right). \tag{11}$$

Substituting $\xi_t = Z_t/\operatorname{Tr}(Z_t)$ gives

$$\operatorname{Tr}(Z_{t+1}) \leq \operatorname{Tr}(Z_t)\left(1 + \varepsilon\delta\exp(\varepsilon)\left\langle \xi_t, \operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t)\right\rangle\right) = \operatorname{Tr}(Z_t)\left(1 + \varepsilon\delta\exp(\varepsilon)\left\langle \mathbb{1}_{\mathcal{X}} \otimes \xi_t, \Pi_t/\beta_t\right\rangle\right).$$

Now, as $\left\langle \Phi(\rho_t) - \gamma\mathbb{1}_{\mathcal{X}} \otimes \xi_t, \Pi_t\right\rangle \geq 0$, we may again use the fact that $1 + z \leq \exp(z)$ for all real numbers z to obtain

$$\operatorname{Tr}(Z_{t+1}) \leq \operatorname{Tr}(Z_t)\left(1 + \frac{\varepsilon\delta\exp(\varepsilon)}{\gamma}\left\langle \Phi(\rho_t), \Pi_t/\beta_t\right\rangle\right) \leq \operatorname{Tr}(Z_t)\exp\left(\frac{\varepsilon\delta\exp(\varepsilon)}{\gamma}\right). \tag{12}$$

Consequently

$$\operatorname{Tr}(Z_T) \leq \operatorname{Tr}(Z_0)\exp\left(\frac{T\varepsilon\delta\exp(\varepsilon)}{\gamma}\right) = \exp\left(\frac{T\varepsilon\delta\exp(\varepsilon)}{\gamma} + \log(M)\right).$$

On the other hand we have

$$\operatorname{Tr}(Z_T) = \operatorname{Tr}\left[\exp\left(\varepsilon\delta\sum_{t=0}^{T-1}\operatorname{Tr}_{\mathcal{X}}(\Pi_t/\beta_t)\right)\right] \geq \exp\left(\varepsilon\delta\lambda_1\left(\operatorname{Tr}_{\mathcal{X}}\left(\sum_{t=0}^{T-1}\Pi_t/\beta_t\right)\right)\right),$$

and therefore

$$\lambda_1\left(\operatorname{Tr}_{\mathcal{X}}\left(\sum_{t=0}^{T-1}\Pi_t/\beta_t\right)\right) \leq \frac{T\exp(\varepsilon)}{\gamma} + \frac{\log(M)}{\varepsilon\delta}.$$

Given that $M < N$ it follows that

$$\|\mathrm{Tr}_{\mathcal{X}}(Y)\| = \lambda_1(\mathrm{Tr}_{\mathcal{X}}(Y)) \leq (1+2\varepsilon)\left(\frac{\exp(\varepsilon)}{\gamma} + \frac{\log(M)}{T\varepsilon\delta}\right) < \frac{7}{8}.$$

Thus, Y is a dual feasible solution whose objective value is smaller than $7/8$, and we conclude that the optimal value of our semidefinite program is at most $5/8$ as required.

It remains to state and prove the two lemmas that were required in the analysis above. They are as follows.

Lemma 1. *Let $P \in \mathrm{Pos}(\mathcal{X} \otimes \mathcal{Z})$ be any positive semidefinite operator, and assume that $\dim(\mathcal{X}) = 2$. Then $P \leq 2\mathbb{1}_{\mathcal{X}} \otimes \mathrm{Tr}_{\mathcal{X}}(P)$.*

Proof. Let σ_x, σ_y and σ_z denote the Pauli operators on $\mathcal{X}$. In matrix form they are

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad \text{and} \qquad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

As each of these operators is Hermitian, we have that $(\sigma_x \otimes \mathbb{1}_{\mathcal{Z}})P(\sigma_x \otimes \mathbb{1}_{\mathcal{Z}})$, $(\sigma_y \otimes \mathbb{1}_{\mathcal{Z}})P(\sigma_y \otimes \mathbb{1}_{\mathcal{Z}})$ and $(\sigma_z \otimes \mathbb{1}_{\mathcal{Z}})P(\sigma_z \otimes \mathbb{1}_{\mathcal{Z}})$ are positive semidefinite. It therefore holds that

$$2\mathbb{1}_{\mathcal{X}} \otimes \mathrm{Tr}_{\mathcal{X}}(P) = P + (\sigma_x \otimes \mathbb{1}_{\mathcal{Z}})P(\sigma_x \otimes \mathbb{1}_{\mathcal{Z}}) + (\sigma_y \otimes \mathbb{1}_{\mathcal{Z}})P(\sigma_y \otimes \mathbb{1}_{\mathcal{Z}}) + (\sigma_z \otimes \mathbb{1}_{\mathcal{Z}})P(\sigma_z \otimes \mathbb{1}_{\mathcal{Z}}) \geq P$$

as required. □

Lemma 2. *Let P be an operator satisfying $0 \leq P \leq \mathbb{1}$. Then for every real number $\eta > 0$, the following two inequalities hold:*

$$\exp(\eta P) \leq \mathbb{1} + \eta \exp(\eta) P,$$
$$\exp(-\eta P) \leq \mathbb{1} - \eta \exp(-\eta) P.$$

Proof. It is sufficient to prove the inequalities for P replaced by a scalar $\lambda \in [0,1]$, for then the operator inequalities follow by considering a spectral decomposition of P. If $\lambda = 0$ both inequalities are immediate, so let us assume $\lambda > 0$. By the Mean Value Theorem there exists a value $\lambda_0 \in (0, \lambda)$ such that

$$\frac{\exp(\eta\lambda) - 1}{\lambda} = \eta \exp(\eta\lambda_0) \leq \eta \exp(\eta),$$

from which the first inequality follows. Similarly, there exists a value $\lambda_0 \in (0, \lambda)$ such that

$$\frac{\exp(-\eta\lambda) - 1}{\lambda} = -\eta \exp(-\eta\lambda_0) \leq -\eta \exp(-\eta),$$

which yields the second inequality. □

5 Proof that QIP is contained in PSPACE

With the algorithm from the previous section in hand, the proof that QIP $\subseteq$ PSPACE follows the same approach used in [JUW09] to prove QIP(2) $\subseteq$ PSPACE. The proof is described in the two subsections that follow.

5.1 Simulation by bounded-depth Boolean circuits

Let $A = (A_{\text{yes}}, A_{\text{no}})$ be a promise problem in QIP. Our goal is to prove that $A \in \text{PSPACE}$. Given that $\text{PSPACE} = \text{NC}(\mathit{poly})$, as was mentioned in Section 2.4, it suffices to prove $A \in \text{NC}(\mathit{poly})$.

Using Theorem 5.4 of [MW05] we have that there exists a single-coin QMAM-protocol for A with perfect completeness and soundness probability $1/2 + \varepsilon$, for $\varepsilon = 1/64$. (Of course any other sufficiently small positive constant would do, and in fact one can replace ε with an exponentially small value—but this choice is sufficient for our needs.) We will make a small modification in Arthur's specification so that he always accepts outright with probability 4ε, and otherwise measures the registers sent by Merlin according to his original specification. With this modification in place, we have that if $x \in A_{\text{yes}}$, then Arthur can be made to accept with certainty, while if $x \in A_{\text{no}}$ then the maximum probability with which Arthur can be made to accept is smaller than $1/2 + 3\varepsilon$. It also holds that every strategy of Merlin causes Arthur to accept with probability at least 4ε.

Now, for any fixed choice of an input string $x \in A_{\text{yes}} \cup A_{\text{no}}$, let Q be the operator defined from this modified specification of Arthur on the input x as was described in Section 3. Give that Arthur always accepts with probability at least 4ε, it follows that the smallest eigenvalue of Q is at least 2ϵ. Therefore, Q is invertible and satisfies $\|Q^{-1}\| \leq 1/(2\varepsilon)$. Moreover, the semidefinite program defined by Q, as described in Section 3, has an optimal value that is equal to 1 when $x \in A_{\text{yes}}$ and smaller than $1/2 + 3\varepsilon$ when $x \in A_{\text{no}}$.

Next, consider a two-step computation as follows:

1. Compute from a given input string x an explicit description of the operator Q specified above.
2. Run an NC implementation of the algorithm from Section 4 on Q.

The first step of this computation can be performed in $\text{NC}(\mathit{poly})$ using an exact computation. This follows from the fact that in $\text{NC}(\mathit{poly})$ one can first compute explicit matrix representations of all of the gates in the quantum circuit specifying Arthur's measurements, and then process these matrices using elementary matrix operations to obtain Q. Note that, without loss of generality, the description of Q has length polynomial in N, which (as defined in the algorithm) is the dimension of the space on which it acts.

The second step of the computation, which is an NC implementation of the algorithm from Section 4, is not quite as straightforward as the first step. In fact, it is only possible for us to *approximate* this algorithm in NC, as we only know how to approximate the operator $Q^{-1/2}$, the matrix exponentials, and the spectral decompositions needed to obtain the projection operators $\Pi_0, \ldots, \Pi_{T-1}$. Nevertheless, we claim that such an approximation is possible in NC, with sufficient accuracy to distinguish the two cases $x \in A_{\text{yes}}$ and $x \in A_{\text{no}}$. This fact is argued in the subsection following this one.

Under the assumption that the second step is performed in NC, we have that the composition of the two steps is an $\text{NC}(\mathit{poly})$ computation. We therefore obtain that $A \in \text{NC}(\mathit{poly})$ as required.

5.2 A high precision NC implementation of the algorithm

It remains to argue that the algorithm from Section 4 can be approximated by an NC computation with sufficient accuracy to distinguish the cases $x \in A_{\text{yes}}$ and $x \in A_{\text{no}}$ as described above. It will be evident from the discussion that follows that obtaining sufficient accuracy in NC is not a significant challenge; and one could, in fact, demand much greater accuracy (by an order of magnitude) and still be able to perform the computation in NC.

The first step in the implementation of the algorithm is to approximate $Q^{-1/2}$. In more precise terms, we first compute an operator R such that R^2 is a close approximation to Q, and then compute R^{-1} in NC using an exact computation. To compute R, we may compute a spectral decomposition of Q, and then take R to be the operator that results by replacing each eigenvalue in this decomposition with its square root. It is straightforward to perform high-precision approximations of these computations in NC with sufficient accuracy so that $\left\|Q - R^2\right\| \leq \varepsilon$ and $\left\|R^{-1}\right\| \leq 1/\varepsilon$. Now, if we compare two semidefinite programs, one defined by Q as specified in Section 3 and the other defined similarly with Q replaced by R^2, we find that the optimal values are close. More specifically, given that $\left\|Q - R^2\right\| \leq \varepsilon$, the optimal values of the two semidefinite programs can differ by at most 2ε. Thus, the optimal value of the semidefinite program for R^2 is at least $1 - 2\varepsilon > 7/8$ in case $x \in A_{\text{yes}}$ and at most $1/2 + 5\varepsilon < 5/8$ in case $x \in A_{\text{no}}$.

In the interest of clarity, to avoid introducing a new variable R into the analysis that follows, let us simply redefine Q at this point to be R^2. Thus, $Q^{-1/2} = R^{-1}$ is known exactly by our implementation of the algorithm and all of the requirements on Q are in place—which are that $\left\|Q^{-1/2}\right\| \leq 1/\varepsilon = 64$ and the optimal value of the semidefinite program in Section 3 defined by Q is at least $7/8$ if $x \in A_{\text{yes}}$ and at most $5/8$ if $x \in A_{\text{no}}$.

Next, let us focus on the projection operators

$$\Pi_0, \ldots, \Pi_{T-1} \in \operatorname{Pos}(\mathcal{X} \otimes \mathcal{W}) \tag{13}$$

and the density operators

$$\rho_0, \ldots, \rho_T \in \operatorname{D}(\mathcal{X} \otimes \mathcal{W} \otimes \mathcal{Y}) \qquad \text{and} \qquad \xi_0, \ldots, \xi_T \in \operatorname{D}(\mathcal{W}) \tag{14}$$

that are to be computed in the course of the algorithm. We will choose an integer K that we take to represent the number of bits of accuracy with which these operators are stored. In more precise terms, the algorithm will store the real and imaginary parts of each of the entries of the above operators (13) and (14) as integers divided by 2^K. It will suffice to take $K = c\lceil \log(N) \rceil$, for a suitable choice of a constant c, although one could in fact afford to take K to be polynomial in N rather than logarithmic. As each entry of these operators has absolute value at most 1, the total number of bits needed to represent the entire collection of operators is $O(TKN^2)$, which is polynomial in N.

In addition to the above operators, the algorithm will store the scalar values $\beta_0, \ldots, \beta_{T-1}$. These values do not need to be approximated; each value β_t is computed exactly as the rational number defined by the operators ρ_t and Π_t stored by the algorithm. We will not consider that the operators $W_1, \ldots, W_T$ and $Z_1, \ldots, Z_T$ are stored by the algorithm at all, as their only purpose in the computation is to specify the density operators $\rho_1, \ldots, \rho_T$ and $\xi_1, \ldots, \xi_T$.

We will also take μ to be a small constant, say $\mu = 2^{-10}$, that will represent an error parameter for the computation. Similar to the choice of K, we could afford to take μ to be significantly smaller than this and still be able to perform the computation in NC.

Now, consider the two steps (a) and (b) that are performed within each iteration of the loop in step 2 of the algorithm. We must approximate these steps, and we demand the following accuracy requirements when doing this. For step (a), we will require that the projection operator Π_t computed by the algorithm satisfies the condition

$$\Pi_t(\Phi(\rho_t) - \gamma \mathbb{1}_{\mathcal{X}} \otimes \xi_t)\Pi_t \geq P_t - \frac{\mu}{M}\mathbb{1}_{\mathcal{X}\otimes\mathcal{W}}, \tag{15}$$

where P_t is defined as the positive part of $\Phi(\rho_t) - \gamma \mathbb{1}_{\mathcal{X}} \otimes \xi_t$. It is possible to perform such a computation in NC by setting the error parameter η in an approximate spectral decomposition

computation of $\Phi(\rho_t) - \gamma \mathbb{1}_{\mathcal{X}} \otimes \xi_t$ as $\eta = \mu/(2M)$, for instance. Then, Π_t is taken to be the appropriately defined projection operator rounded to K bits of accuracy. For step (b), we will require that

$$\|\rho_{t+1} - W_{t+1}/\operatorname{Tr}(W_{t+1})\| < \frac{\mu\delta}{N} \qquad \text{and} \qquad \|\xi_{t+1} - Z_{t+1}/\operatorname{Tr}(Z_{t+1})\| < \frac{\mu\delta}{M}. \tag{16}$$

In these inequalities we do not consider that W_{t+1} and Z_{t+1} are stored by the algorithm, but rather we consider that they are operators *defined* by the equations

$$W_{t+1} = \exp\left(-\epsilon\delta \sum_{j=0}^{t} \Phi^*(\Pi_j/\beta_j)\right) \qquad \text{and} \qquad Z_{t+1} = \exp\left(\varepsilon\delta \sum_{j=0}^{t} \operatorname{Tr}_{\mathcal{X}}(\Pi_j/\beta_j)\right),$$

for the particular operators $\Pi_0/\beta_0, \ldots, \Pi_t/\beta_t$ that are stored by the algorithm. The algorithm's *approximations* of W_{t+1} and Z_{t+1} determine the density operators ρ_{t+1} and ξ_{t+1}. As the matrix exponentials are to be computed for operators having norm bounded by $T = O(\log N)$, it is clear that ρ_{t+1} and ξ_{t+1} with the required properties can be computed in NC.

Finally, we have that the total number of iterations in the algorithm is $T = O(\log N)$. Given that each of the iterations of the algorithm can be performed in NC, and that the total number of bits that must be stored from one iteration to the next is polynomial in N, we have that the composition of these T iterations can be performed in NC as well.

It remains only to show that the approximations (15) and (16) are sufficient to guarantee that the algorithm accepts or rejects correctly. This analysis is done in almost exactly the same way as was presented in Section 4. Even though the operators

$$\rho_0, \ldots, \rho_{T-1}, \quad \xi_0, \ldots, \xi_{T-1}, \quad \text{and} \quad \Pi_0/\beta_0, \ldots, \Pi_{T-1}/\beta_{T-1}$$

do not necessarily satisfy the precise equations that were assumed in Section 4, they may nevertheless be used to construct primal and dual solutions to the semidefinite program that satisfy the required bounds.

In the case that the algorithm accepts, a consideration of the operators $\rho = \rho_t$, $\Pi = \Pi_t$, and $\xi = \xi_t$ as before allows for the construction of a primal feasible solution with a large objective value. In place of (7), we have

$$\Phi(\rho) \leq \mathbb{1}_{\mathcal{X}} \otimes \left(\gamma\xi + 2\operatorname{Tr}_{\mathcal{X}}(\Pi\Phi(\rho)\Pi) + \frac{\mu}{M}\mathbb{1}_{\mathcal{W}}\right),$$

which allows for a lower bound of $1/(\gamma + 2\varepsilon + \mu)$ for the primal objective function. For our choice $\mu = 2^{-10}$ of an error bound, this quantity is still lower-bounded by $5/8$, which implies that the algorithm has operated correctly in this case.

A similar analysis to the one before holds for the case of rejection as well. We consider the operators

$$\Pi_0/\beta_0, \ldots, \Pi_{T-1}/\beta_{T-1}$$

produced by the algorithm, and take

$$Y = \frac{(1+2\varepsilon)(1+2\mu)}{T} \sum_{t=0}^{T-1} \Pi_t/\beta_t.$$

When proving the dual feasibility of Y we are no longer free to substitute $\rho_t = W_t/\operatorname{Tr}(W_t)$, but instead we must introduce a small error term due to the fact that ρ_t is just an approximation to

$W_t / \operatorname{Tr}(W_t)$. By the first inequality of (16) above we may conclude that

$$\left\langle \frac{W_t}{\operatorname{Tr}(W_t)}, \Phi^*(\Pi_t/\beta_t) \right\rangle \geq 1 - \mu;$$

and by substituting this into (11) and following a similar argument to the one from before we obtain

$$\lambda_N(\Phi^*(Y)) \geq (1+2\varepsilon)(1+2\mu)\left((1-\mu)\exp(-\varepsilon) - \frac{\varepsilon^2}{4}\right) > 1.$$

Thus, dual feasibility holds for Y. Along similar lines, by using (15) and (16), one finds again that the dual objective value achieved by Y less than $7/8$, and therefore the algorithm operates correctly in this case as well.

Acknowledgments

We thank Xiaodi Wu for helpful discussions. Rahul Jain's research is supported by the internal grants of the Centre for Quantum Technologies, which is funded by the Singapore Ministry of Education and the Singapore National Research Foundation. Zhengfeng Ji's research at the Perimeter Institute is supported by the Government of Canada through Industry Canada and by the Province of Ontario through the Ministry of Research & Innovation. Sarvagya Upadhyay's research is supported in part by Canada's NSERC, CIFAR, MITACS, QuantumWorks, Industry Canada, Ontario's Ministry of Research and Innovation, and the U.S. ARO. John Watrous's research is supported by Canada's NSERC and CIFAR.

References

[AB09] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[AHK05] S. Arora, E. Hazan, and S. Kale. Fast algorithms for approximate semidefinite programming using the multiplicative weights update method. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 339–348, 2005.

[AK07] S. Arora and S. Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, pages 227–236, 2007.

[Bab85] L. Babai. Trading group theory for randomness. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 421–429, 1985.

[BCP83] A. Borodin, S. Cook, and N. Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58:113–136, 1983.

[BGH82] A. Borodin, J. von zur Gathen, and J. Hopcroft. Fast parallel matrix and GCD computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 65–71, 1982.

[Bha97] R. Bhatia. *Matrix Analysis*. Springer, 1997.

[BM88] L. Babai and S. Moran. Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36(2):254–276, 1988.

[BOFKT86] M. Ben-Or, E. Feig, D. Kozen, and P. Tiwari. A fast parallel algorithm for determining all roots of a polynomial with real roots. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 340–349, 1986.

[BOGKW88] M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson. Multi-prover interactive proofs: how to remove intractability assumptions. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 113–131, 1988.

[Bor77] A. Borodin. On relating time and space to size and depth. *SIAM Journal on Computing*, 6:733–744, 1977.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[Csa76] L. Csanky. Fast parallel matrix inversion algorithms. *SIAM Journal on Computing*, 5(4):618–623, 1976.

[ESY84] S. Even, A. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, 1984.

[Fel86] P. Feldman. The optimum prover lies in PSPACE. Manuscript, 1986.

[FK97] U. Feige and J. Kilian. Making games short. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 506–516, 1997.

[Gat93] J. von zur Gathen. Parallel linear algebra. In J. Reif, editor, *Synthesis of Parallel Algorithms*, chapter 13. Morgan Kaufmann Publishers, Inc., 1993.

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 291–304, 1985.

[GMR89] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GMW91] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(1):691–729, 1991.

[Gol05] O. Goldreich. On promise problems (a survey in memory of Shimon Even [1935–2004]). Electronic Colloquium on Computational Complexity, Report TR05-018, 2005.

[GS89] S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. In S. Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 73–90. JAI Press, 1989.

[HKSZ08] S. Hallgren, A. Kolla, P. Sen, and S. Zhang. Making classical honest verifier zero knowledge protocols secure against quantum attacks. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*, volume 5126 of *Lecture Notes in Computer Science*, pages 592–603. Springer, 2008.

[HO02] L. Hemaspaandra and M. Ogihara. *The Complexity Theory Companion*. Springer, 2002.

[JUW09] R. Jain, S. Upadhyay, and J. Watrous. Two-message quantum interactive proofs are in PSPACE. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009. To appear.

[JW09] R. Jain and J. Watrous. Parallel approximation of non-interactive zero-sum quantum games. In *Proceedings of the 24th IEEE Conference on Computational Complexity*, pages 243–253, 2009.

[Kal07] S. Kale. *Efficient algorithms using the multiplicative weights update method*. PhD thesis, Princeton University, 2007.

[KKMV09] J. Kempe, H. Kobayashi, K. Matsumoto, and T. Vidick. Using entanglement in quantum multi-prover interactive proofs. *Computational Complexity*, 18(2):273–307, 2009.

[KM03] H. Kobayashi and K. Matsumoto. Quantum multi-prover interactive proof systems with limited prior entanglement. *Journal of Computer and System Sciences*, 66(3):429–450, 2003.

[Kob08] H. Kobayashi. General properties of quantum zero-knowledge proofs. In *Proceedings of the Fifth IACR Theory of Cryptography Conference*, volume 4948 of *Lecture Notes in Computer Science*, pages 107–124. Springer, 2008.

[KW00] A. Kitaev and J. Watrous. Parallelization, amplification, and exponential time simulation of quantum interactive proof system. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 608–617, 2000.

[LFKN92] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.

[ML03] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[MW05] C. Marriott and J. Watrous. Quantum Arthur-Merlin games. *Computational Complexity*, 14(2):122–152, 2005.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[Nef94] C. A. Neff. Specified precision polynomial root isolation is in NC. *Journal of Computer and System Sciences*, 48(3):429–463, 1994.

[Sha92] A. Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.

[She92] A. Shen. IP = PSPACE: simplified proof. *Journal of the ACM*, 39(4):878–880, 1992.

[Wat99] J. Watrous. PSPACE has constant-round quantum interactive proof systems. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 112–119, 1999.

[Wat09] J. Watrous. Zero-knowledge against quantum attacks. *SIAM Journal on Computing*, 39(1):25–58, 2009.

[WK06] M. Warmuth and D. Kuzmin. Online variance minimization. In *Proceedings of the 19th Annual Conference on Learning Theory*, volume 4005 of *Lecture Notes in Computer Science*, pages 514–528. Springer, 2006.